Advanced Algorithms 2019 Lecturer: Mohsen Ghaffari

11

Student: Sebastian Brandt *12/06, 2019*

1 Online Edge Coloring [Recommended]

Given a fixed set of vertices V, a set of edges $E \subseteq V \times V$ arrives over time and upon arrival of each edge, we should color it with one of colors $\{1, 2, \ldots, q\}$. This is the permanent color of that edge and cannot be changed later. The coloring should be a proper coloring at all times, i.e., no two edges that share an endpoint should receive the same color. Suppose that we are given the guarantee that at all times, the maximum degree of any node is at most Δ. Notice that by Vizing's theorem, the offline algorithm can color the edges using just $\Delta + 1$ colors.

(A) Devise an online algorithm that computes a $(2\Delta - 1)$-edge-coloring.

Hint:

What is the simplest thing you can think of?

(B) More interestingly, prove that any deterministic online edge-coloring algorithm requires at least $2\Delta - 1$ colors, i.e., no deterministic online algorithm can get a competitive ratio better than 2.

Hint:

What would you do as the adversary playing against the coloring algorithm that has to color everything your give it? Fix a small Δ (say, $\Delta = 4$) and play! Don't forget you can use arbitrarily many vertices.

2 Hungry Cow [Recommended]

Consider the following hungry cow problem—a cow stands on the x-axis at the origin, and is looking for a nice patch of yummy green grass, which it knows exists somewhere on the x-axis at some integer distance $d \geq 1$ either to the left or to the right of the origin. Neither the distance nor the side are known to the cow. Devise a 9-competitive algorithm for the cow with respect to the distance it needs to travel to get to the food.

Hint:

Cows like exponential zigzag strategies!

3 Optimal Offline Algorithm for Paging

Devise a polynomial-time algorithm for computing the optimal offline solution for the paging problem. Prove its correctness and analyze its time complexity.

Hint:

For the correctness proof: Let's say you want to compare an arbitrary algorithm A to your optimal offline algorithm OPT. Can you show that by replacing the next decision of A (and perhaps some other decisions of A) by the decision of OPT, algorithm A doesn't become worse?