Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
Philipp Bamberger

Distributed Systems, Summer Term 2019
Exercise Sheet 7

1 3^Δ-Coloring

a) Describe an algorithm that colors a directed graph with out-degree at most 1 with 6 colors in $O(\log^* n)$ rounds.

Hint: Use the ring coloring algorithm from the lecture.

b) Given a directed graph with out-degree at most 1 which is colored with $m > 3$ colors, describe a method to recolor the graph in one round using $m - 1$ colors.

c) Use part a) and b) to develop an algorithm that colors an (undirected) graph with at most 3^Δ colors in time $O(\log^* n)$. You can assume that Δ is known to all nodes.

Sample Solution

a) Assume that the nodes have IDs with $\log n$ bits. Initially, each node sets $c_v = \mathrm{ID}(v)$.

Each node without out-neighbor assigns itself color 0 (assume that all IDs are > 0).
Each other node v executes the following code:
Send own color c_v to all in-neighbors
repeat
  receive color c_p from out-neighbor
  interpret c_v and c_p as bit-strings
  let i be the index of the smallest bit where c_v and c_p differ
  the new label is i (as bitstring) followed by the ith bit of c_v
  send c_v to all in-neighbors

The analysis (correctness and runtime) is the same as for the ring coloring algorithm.

b) Let $\{1, \ldots, m\}$ be the set of colors. To eliminate color m, each node recolors itself with the color of its out-neighbor, where nodes without out-neighbors choose a new (different) color from $\{1, \ldots, m-1\}$. This yields a valid coloring where siblings (in-neighbor of the same node) are monochromatic. This means that each node has only two different colors in its neighborhood, so each node with color m can choose a new color from $\{1, 2, 3\}$.

c) Given a graph $G = (V, E)$, orient the edges arbitrarily which requires one communication round (e.g., orient each edge towards the node with higher ID). Then each node v labels its outgoing edges with numbers $1, \ldots, \deg_{\text{out}}(v)$ where $\deg_{\text{out}}(v) \leq \Delta$ is the out-degree of v. Now consider the oriented graph with V as node set and all edges with label i. This is a graph with out-degree at most 1 which we can color with three colors in time $O(\log^* n)$. Do this in parallel for each $i \in \{1, \ldots, \Delta\}$. Let c_i^v be the color that node v gets for component i. v takes as its final color the vector $(c_1^v, \ldots, c_\Delta^v)$. As $c_i^v \in \{1, 2, 3\}$, at most 3^Δ colors are used. The coloring is valid because if two nodes are connected by an edge with label i, the ith components in their color vectors are different. Each node v needs to know Δ because otherwise, if ℓ is the largest label that some incoming/outgoing edge of v has, v can not know whether or not it has to choose a value c_i^v for all $i > \ell$.

2 Color Reduction

a) Given a graph which is colored with $m > \Delta + 1$ colors, describe a method to recolor the graph in one round using $m - \lfloor \frac{m}{\Delta+2} \rfloor$ colors.

Hint: Partition the set of colors into sets of size $\Delta + 2$ and recall the color reduction method from the lecture.

b) Show that after $O(\Delta \log(m/\Delta))$ iterations of step a), one obtains a $O(\Delta)$ coloring.

Sample Solution

a) Partition the set of colors into $\lfloor \frac{m}{\Delta+2} \rfloor$ disjoint sets of size $\Delta + 2$ and one set of size at most $\Delta + 1$. From each set C of size $\Delta + 2$, take the largest color and let each node v with this color choose a new color from C that is not among the colors of its neighbors. If a neighbor u of v concurrently chooses a new color, it will not cause a conflict as u chooses from a disjoint color set. So we obtain a new coloring with $m - \lfloor \frac{m}{\Delta+2} \rfloor$ colors.

b) We calculate the number of iterations needed to obtain at most $2(\Delta + 2)$ colors. In one iteration m is reduced to

$$m - \left\lfloor \frac{m}{\Delta + 2} \right\rfloor \leq m - \frac{m}{\Delta + 2} + 1 = m\left(1 - \left(\frac{1}{\Delta + 2} - \frac{1}{m}\right)\right) \overset{m \geq 2(\Delta+2)}{\leq} m\left(1 - \frac{1}{2(\Delta + 2)}\right)$$

So we are looking for the minimum t such that

$$m\left(1 - \frac{1}{2(\Delta + 2)}\right)^t \leq 2(\Delta + 2)$$

For all $x \in \mathbb{R}$ it holds $1 + x \leq e^x$. It follows

$$m\left(1 - \frac{1}{2(\Delta + 2)}\right)^t \leq m \cdot e^{-\frac{t}{2(\Delta+2)}} \leq 2(\Delta + 2) \ ,$$

so we choose $t = \left\lceil 2(\Delta + 2) \ln\left(\frac{m}{2(\Delta+2)}\right)\right\rceil$.

Once we obtained $O(\Delta)$ colors, we can use a) another $O(\Delta)$ times until $\Delta + 1$ colors are left (as long as $m > \Delta + 1$, at least one color is eliminated in each step). This yields an overall runtime of $O(\Delta \log(m/\Delta))$.