Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute of Theoretical Computer Science
Mohsen Ghaffari, Angelika Steger, David Steurer, Emo Welzl, Peter Widmayer

Algorithms, Probability, and Computing **Solutions KW 50** **HS17**

Solution 1: Cover free families

The solution to this exercise is obtained by bending the description of the task into the terms of Lemma 8.18 from the lecture notes. We take a k-cover free family $\mathcal{F} = \{F_1, \dots, F_n\} \subseteq 2^{[m]}$ which by Lemma 8.18 exists for $m = O(k^2 \log n)$. For each $i \in \{1, \dots, n\}$ we then define $Enc(i)$ to be the characteristic vector $\chi(F_i) \in \{0,1\}^m$ of the set $F_i \in \mathcal{F}$. We claim that this choice of encoding produces what is asked in the exercise.

Consider the two disjoint sets $S, S' \subseteq [n]$ with $|S|, |S'| \leq k$. Because of disjointness there is some $i \in S$ so that $i \notin S'$. Because of the k-cover free family property there is some element $e \in F_i$ so that for any $j \in S'$ it holds that $e \notin F_j$. In other words, $e \in F_i$ and $e \notin \cup_{j \in S'} F_j$. It remains to observe that $\vee_{j \in S} Enc(j) = \chi(\cup_{j \in S} F_j)$, i.e., the OR of the encodings corresponds to the taking the characteristic vector of the union of the respective sets and since $e \in \cup_{j \in S} F_j$ and $e \notin \cup_{j \in S'} F_j$, the claim follows.

You might see that we can actually weaken the assumption in the exercise to require that only one of the sets should have size at most k. It is also interesting to see other kinds of "corruption" of the data. Think for example what you would do if we replaced the OR with XOR?

Solution 2: More Algorithms for Color Reduction

(a) We put the colors of the given k-coloring into $\lfloor \frac{k}{\Delta+2} \rfloor$ buckets, each of size $\Delta + 2$ except the last one, which may have size between $\Delta + 2$ to $2\Delta + 3$. Within each bucket we can in one round reduce the number of colors by 1, using the method from lemma 8.21. We can do this in all buckets in parallel, so within one round we have reduced the number of colors by $\lfloor \frac{k}{\Delta+2} \rfloor$.

(b) First we run the $O(\log^* n)$ algorithm that gives us a $C\Delta^2 \log \Delta$-coloring for some constant C. Then we repeat the algorithm from (a) as often as possible, that is, until $k \leq \Delta + 1$.

In order to analyze the number of iterations of (a) that we need, observe first that $\lfloor \frac{k}{\Delta+2} \rfloor \geq \frac{k}{2\Delta+4}$ since $k \geq \Delta + 2$. Therefore the number of colors drops in each round to least a $(1 - \frac{1}{2\Delta+4})$-fraction of the previous number of colors. The number of iterations is therefore bounded by any number t that satisfies

$$\left(1 - \frac{1}{2\Delta + 4}\right)^t C\Delta^2 \log \Delta < 1 \tag{1}$$

We could replace the right hand side by $\Delta + 1$ but choosing 1 makes the calculations easier. Let us solve for t in (1) to get that

$$t \ln\left(1 - \frac{1}{2\Delta+4}\right) < -\ln(C\Delta^2 \log \Delta)$$
$$\Leftrightarrow \quad t > \frac{-\ln(C\Delta^2\log\Delta)}{\ln\left(1-\frac{1}{2\Delta+4}\right)} \geq \frac{-\ln(C\Delta^2\log\Delta)}{-\frac{1}{2\Delta+4}} = \Theta(\Delta\log\Delta)$$

where in the last inequality we used that $1 + x \leq e^x$ with $x = -\frac{1}{2\Delta+4}$. It follows from the termination criterion of our algorithm that the final coloring uses no more that $\Delta + 1$ colors.

Solution 3: More Algorithms for Color Reduction

(a) Initially we have a $(\Delta + 1)$-coloring for G_0 and a $(\Delta + 1)$-coloring for G_1. We denote the colors for G_0 as $\{1, \dots, \Delta + 1\}$ and the colors for G_1 as $\{\Delta + 2, \dots, 2\Delta + 2\}$ so that the coloring that we have attained recursively is a proper coloring (think how to decide the role of G_0 and G_1 in the LOCAL model). We want to color G_1 also with the colors $\{1, \dots, \Delta + 1\}$ so that the whole graph is properly colored.

We consider the colors $\{\Delta + 2, \dots, 2\Delta + 2\}$ as a schedule so that in the first round those vertices with color $2\Delta + 2$ choose a new color from $\{1, \dots, \Delta + 1\}$ that is not used by any of their neighbors and in the second round those with color $2\Delta + 1$ chose a new color from $\{1, \dots, \Delta + 1\}$ and so on. This process terminates in $O(\Delta)$ rounds and it remains to argue why the vertices always have a color that they can choose.

We argue inductively over the schedule and we prove that the vertices always have a choice for a new color and that the coloring resulting after the changes in one round stays proper. The initial $(2\Delta + 2)$-coloring is proper. Say that a vertex v from G_1 has originally color $i \in \{\Delta + 2, \dots, 2\Delta + 2\}$ and that it is the turn of v to choose a new color. Because the original coloring of G_1 was a proper coloring and since all vertices that have changed color have colors from $\{1, \dots, \Delta + 1\}$, v is not adjacent to any other vertex that currently has color i. Because v has degree at most Δ, there is some available color in $\{1, \dots, \Delta + 1\}$ not used by its neighbors which it can pick. Since no neighbor of v changed their color in the same round, the coloring that results after the change is a proper coloring. This completes the correctness proof.

(b) Let $t(b)$ denote the number of rounds the procedure takes if we start it with a coloring that uses b bits. Then $t(b)$ satisfies the recurrence

$$t(b) \leq O(\Delta) + t(b-1) = c\Delta + t(b-1)$$

for some fixed constant c. We want to show that $t(b) \leq C\Delta b$ for some constant C. We use induction on b. The recursion bottoms up when $b = 1$ in which case the bound is clear. Using the recurrence and the induction assumption we now get that

$$t(b) \leq c\Delta + t(b-1) \leq c\Delta + C\Delta(b-1) = C\Delta b + \Delta(c - C) \leq C\Delta b$$

as long as $C \geq c$ which we can require. Therefore $t(b) = O(\Delta b)$ and since we call the algorithm with a coloring that uses $b = O(\log\Delta)$ bits we are done.