Principles of Distributed Computing	05/13, 2020
Solution 12	
Lecturer: Mohsen Ghaffari	

While it is possible to solve the exercises without the following theorem, the calculations required get significantly longer. So for simplicity, we will use the following:

Theorem 1 (Chernoff Bound, upper tail) *Let X be the sum of independent random indicator variables $X_1, \ldots, X_n$, and let $\mu = \mathbb{E}[X]$. Then for any $\delta \geq 0$,*

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\frac{\delta^2 \mu}{2+\delta}}.$$

1 Vertex Coloring using All-to-All Communication

Our goal will be to partition the graph into different parts, such that we can (1) solve the coloring problem on each part separately, and (2) each part has size (i.e. number of edges) at most $O(n/\log n)$. We can then use the routing result seen in the lecture, which allows us to send all information about one part to some vertex, which can then perform the coloring locally, and inform all vertices about their color. Note, that the problem is not about vertices having enough capacity to send the necessary information, they can always send most of their incident edges, but about one vertex having enough capacity to receive all information required to make a decision.

To be precise, we randomly divide the graph into $k = \Delta/\log n$ parts, by each vertex choosing one part at random. This will result in graphs $G_1, \ldots, G_\ell$, where each G_i contains all vertices of part i, as well as all edges between them. First, let us analyze the number of vertices in G_i. Letting X denote the number of vertices in G_i, we can see that $\mathbb{E}[X] = n/\ell$, and that X can be written as the sum of independent random indicator variables, where each indicator is one if a vertex is in part i, and zero otherwise. Further, note that by our choice of ℓ, we have that $\mathbb{E}[X] \geq \log n$. Thus, we can use a Chernoff Bound to get:

$$\Pr[X \geq 10n/\ell] \leq e^{-\frac{100n/\ell}{12}} \leq e^{-8\log n} = n^{-8}.$$

By a union bound over all graphs G_i, we can conclude that each graph contains at most $10n/\ell$ vertices with probability at least $1 - n^{-7}$.

Next, we will analyze the degree of a vertex in G_i. Letting Y denote said degree, we observe that $\mathbb{E}[Y] \leq \Delta/\ell = \log n$. Again, we can also observe that Y can be written as a sum of independent random variables, one for each neighbor. Therefore, by a Chernoff bound we have:

$$\Pr[Y \geq 10\log n] \leq e^{-8\log n} = n^{-8}.$$

In the same way as for the number of vertices, by a union bound over all vertices, the probability that there is some vertex which has degree more than $10\log n$, is at most $1 - n^{-7}$.

Combining these results, we get that each graph G_i contains at most $\frac{100n\log^2 n}{\Delta}$ edges. As $\Delta = \Omega(\log^3 n)$, this is at most $O(n/\log n)$.

Thus, we can use the routing result from the lecture to conclude that for each part G_i there is a node that can learn the topology of G_i. By using different palettes (i.e. colors $1, \ldots, 10\log n$ for $i = 1$, $10\log n + 1, \ldots, 20\log n$ for $i = 2$, etc.) for each part, each part can be colored independently, using at most $10\log n \cdot \Delta/\log n = O(\Delta)$ colors.

2 Edge Coloring

(2a) Let us set $q = 40\Delta \log n$. We create a set $C(e)$ for each edge e by randomly sampling $r = 10 \log n$ colors from $\{1, \ldots, q\}$ with replacement[1]. We can also think of this as $C(e)$ having r slots, and for each slot we pick a color at random.

In the following, we will show that with high probability, for each edge e, the set $C(e)$ contains a color that is contained by none of the edges adjacent to e. Let us look at one particular edge e, and let us assume that all other edges e' have already chosen their set $C(e')$. First, notice that there are at most $2\Delta - 2 \leq 2\Delta$ adjacent edges. Each of these edges e' has a set $C(e')$ of cardinality at most $r = 10 \log n$. Thus, there are at least

$$40\Delta \log n - 2\Delta \cdot 10 \log n = 20\Delta \log n$$

colors that are not used by edge adjacent to e, i.e. they are good for e. Picking one such color in one sampling step has probability at least $\frac{20\Delta \log n}{40\Delta \log n} = \frac{1}{2}$. Thus, the probability that within $r = 10 \log n$ sampling steps, we never pick a good color is at most $2^{-r} = n^{-10}$. Finally, we can union bound over all at most n^2 many edges, to get that each edge has a color that is good with probability at least n^{-8}. Note, that by multiplying both q and r by some constant c, this probability can be made smaller than n^{-c} (in fact n^{2-10c}).

(2b) For this exercise, the main idea is the following: In a first step, we choose colors such that the probability of one edge finding a good color is only $1 - 1/\text{poly}(\log n)$. While this probability is smaller than we would like, it does allow us to argue that the number of adjacent edges drops by a factor of roughly $\log n$ for each edge. And for this event we can get the guarantee that it holds with high probability, which also allows us to union bound over all edges. Finally we observe that this reduction in the number of adjacent edges (which can also be thought of as a reduction in the degree of each node) leaves us in a setting where the relationship between Δ and the number of allowed colors is the same as in part (a).

To be precise, we will perform two steps of sampling, where we use different colors in each step. For the first step, we will use $q_1 = 40\Delta \log \log n$ colors, and create each set $C(e)$ by sampling $r_1 = 10 \log \log n$ of them, with replacement. Using the same analysis as before, it can be shown that the probability that one edge finds a good color is now at least $1 - \log^{-10} n$. All these edges that found a good color are now ignored in the second round. Thus, for a given edge e, we expect its number of adjacent edges to drop by at least a factor of $\log^{10} n$. While the events that two edges are removed are not independent[2], we can observe that the events that we analyzed are independent, as we calculated the probability that an edge finds a good color independently of what color its adjacent edges choose. Thus, the number of remaining adjacent edges is stochastically dominated by a sum of at most 2Δ independent random variables that are 1 with probability at most $\log^{-10} n$, and 0 otherwise. Using a Chernoff bound, this allows us to conclude that the number of adjacent edges being larger than $\Delta / \log n$ is at most n^{-10} (for larger enough n).

For the second step of sampling, we will sample in the same way as in the first step, just using a different palette of colors. We can notice that this setup is similar to part (a), in fact even a bit stronger, as we have a factor of $\log \log n$ colors more, since effectively the value of Δ has changed.

Overall, we used $80\Delta \log \log n = O(\Delta \log \log n)$ colors, and by a union bound over both steps, we succeed with high probability.

[1] While this might not seem like a smart thing to do, as we potentially waste some sampling steps by picking a previously chosen color, it helps with the analysis, as it means that the color choices in each step are independent

[2] For part (a), this was not an issue, as we were using a union bound for different edges.

(2c) From the lecture notes, we know that there is a bipartite graph H, such that an edge coloring of H implies a routing scheme. Thus, we only need to argue that we can implement the color checker on H. Recall that H is bipartite with nodes $\{a_1, \ldots, a_n\}$ on one, and $\{b_1, \ldots, b_n\}$ on the other side. For every message from i to j, we have an edge from a_i to b_j.

We will let all nodes i be responsible for the incident edges of a_i, so they will perform both the sampling of colors and the following checking. Let e be one edge incident to node i, and suppose we have chosen the colors $C(e)$ for e. A node i will find a good color for all edges incident to a_i as follows: if there are two edges that chose the same color we know that this color is not good. For a color in $C(e)$, for which we do not that it is not good yet, we will spend $O(1)$ rounds to find out whether it is good or not. In the first 2 rounds, we will check the colors $1, \ldots, n$, by sending the endpoints of every edge of color c to the node with ID c. Now every node c can check if there are two edges e and e' that share an endpoint and chose the same color. This works, as there is at most one edge of a given color c that needs to be sent from any node i. For the next two rounds, we check the colors $n + 1, \ldots, 2n$, and so on.

This works in $O(1)$ rounds, as every node needs to send $\Delta = O(n/\log n)$ (resp. $O(n/\log\log n)$) messages, and we use $O(\Delta \log n) = O(n)$ (resp. $O(\Delta \log\log n) = O(n)$) colors. Having this color checker, we can now color the graph H and use the coloring to route the messages as discussed in the lecture.