Principles of Distributed Computing 03/14, 2017

Lecture 4

Lecturer: Mohsen Ghaffari *Scribe:*

In this lecture, we discuss *distributed algorithms for coloring general graphs.* In particular, we will see the following two results:

1. An algorithm for $(\Delta+1)$-vertex-coloring with round complexity $O(\Delta \log \Delta + \log^* n)$.
2. An algorithm for $(2\Delta-1)$-edge-coloring with round complexity $O(\Delta + \log^* n)$.

1 Vertex-Coloring

In this section, we start the study of LOCAL vertex-coloring algorithms for general graphs. The eventual goal would be to obtain $(\Delta+1)$-coloring of the graphs — that is, an assignment of colors $\{1, 2, \ldots, \Delta+1\}$ to vertices such that no two adjacent vertices receive the same color — where Δ denotes the maximum degree. Notice that by a simple greedy argument, each graph with maximum degree at most Δ has a $(\Delta+1)$-coloring: color vertices one by one, each time picking a color which is not chosen by the already-colored neighbors. However, this greedy argument does not lead to an efficient LOCAL procedure for finding such a coloring[1].

1.1 Take 1: Linial's Coloring Algorithm

We start with presenting an $O(\log^* n)$-round algorithm that computes a $O(\Delta^2 \log \Delta)$ coloring. This algorithm is known as Linial's coloring algorithm [Lin87, Lin92]. In the next subsection, we see how to transform this coloring into a $(\Delta+1)$-coloring.

Theorem 1. *There is a deterministic distributed algorithm in the* LOCAL *model that colors vertices of any n-node graph G with maximum degree Δ using $O(\Delta^2 \log \Delta)$ colors, in $O(\log^* n)$ rounds.*

Conceptually, the algorithm can be viewed as a more general variant of the Colve-Vishkin coloring algorithm for oriented trees, which we discussed in the first lecture. In particular, the main ingredient in Theorem 1 is a single-round color reduction method, conceptually similar to the single-round color reduction of Cole-Vishkin for trees. However, here, each node has to ensure that the color it picks is different than all of its neighbors, and not just its parents. For that purpose, we will work with a concept called *cover free families*, which we introduce next.

Definition 2. *Given a ground set $\{1, 2, \ldots, k'\}$, a family of sets S_1, S_2, ..., $S_k \subseteq \{1, 2, \ldots, k'\}$ is called a Δ-cover free family if for each set of indices $i_0, i_1, i_2, \ldots, i_\Delta \in \{1, 2, \ldots, k\}$, we have $S_{i_0} \setminus (\cup_{j=1}^{\Delta} S_{i_j}) \neq \emptyset$. That is, if no set in the family is a subset of the union of Δ other sets.*

Notice that in particular, a Sperner family —where no set S_i is a subset of another set S_j— is simply a 1-cover free family.

We use cover-free families to obtain a single-round color reduction algorithm that allow us to transform any k-coloring to a k'-coloring, for $k' \ll k$. We would like to have k' be as small as possible, as a function of k and Δ. In the following, we prove the existence of a Δ-cover free families with a ground set size $k' = O(\Delta^2 \log k)$.

[1] The straightforward transformation of this greedy approach to the LOCAL model would be an algorithm that may need $\Omega(n)$ rounds.

Lemma 3. *For any k and Δ, there exists a Δ-cover free family of size k on a ground set of size $k' = O(\Delta^2 \log k)$.*

Proof. We use the probabilistic method [AS04] to argue that there exists a Δ-cover free family of size k on a ground set of size $k' = O(\Delta^2 \log k)$. Let $k' = C\Delta^2 \log k$ for a sufficiently large constant $C \geq 2$. For each $i \in \{1, 2, \ldots, k\}$, define each set $S_i \subset \{1, 2, \ldots, k'\}$ randomly by including each element $q \in \{1, 2, \ldots, k'\}$ in S_i with probability $p = 1/\Delta$. We argue that this random construction is indeed a Δ-cover free family, with high probability, and therefore, such a cover free family exists.

First, consider an arbitrary set of indices $i_0, i_1, i_2, \ldots, i_\Delta \in \{1, 2, \ldots, k\}$. We would like to argue that $S_{i_0} \setminus (\cup_{j=1}^{\Delta} S_{i_j}) \neq \emptyset$. For each element $q \in \{1, 2, \ldots, k'\}$, the probability that $q \in S_{i_0} \setminus (\cup_{j=1}^{\Delta} S_{i_j})$ is at exactly $\frac{1}{\Delta}(1 - \frac{1}{\Delta})^{\Delta} \geq \frac{1}{4\Delta}$. Hence, the probability that there is no such element q that is in $S_{i_0} \setminus (\cup_{j=1}^{\Delta} S_{i_j})$ is at most $(1 - \frac{1}{4\Delta})^{k'} \leq exp(-C\Delta \log k/4)$. This is an upper bound on the probability that for a given set of indices set of indices $i_0, i_1, i_2, \ldots, i_\Delta \in \{1, 2, \ldots, k\}$, the respective sets violate the cover-freeness property that $S_{i_0} \setminus (\cup_{j=1}^{\Delta} S_{i_j}) \neq \emptyset$.

There are $k\binom{k-1}{\Delta}$ way to choose such a set of indices $i_0, i_1, i_2, \ldots, i_\Delta \in \{1, 2, \ldots, k\}$, k ways for choosing the central index i_0 and $\binom{k-1}{\Delta}$ ways for choosing the indices $i_1, i_2, \ldots, i_\Delta$. Hence, by a union bound over all these choices, the probability that the construction fails is at most

$$\begin{aligned} k\binom{k-1}{\Delta+1} \cdot exp(-C\Delta \log k/4) &\leq k(\frac{e(k-1)}{\Delta+1})^{\Delta+1} \cdot exp(-C\Delta \log k/4) \\ &\leq exp(\log k + (\Delta+1)(\log k + 1) - C\Delta \log k/4) \\ &\leq exp(-C\Delta \log k/8) \ll 1, \end{aligned}$$

for a sufficiently large constant C. That is, the random construction succeeds to provide us with a valid Δ-cover free family with a positive probability, and in fact with a probability close to 1. Hence, such a Δ-cover free family exists. □

Now we discuss how to use cover-free families to perform a single-round color reduction.

Lemma 4. *Given a k-coloring ϕ_{old} of a graph with maximum degree Δ, in a single round, we can compute a k'-coloring ϕ_{new}, for $k' = O(\Delta^2 \log k)$.*

Proof Sketch. Follows from the existence of cover free families as proven in Lemma 3. Namely, each node v of old color $\phi_{old}(v) = q$ for $q \in \{1, dots, k\}$ will use the set $S_q \subseteq \{1, \ldots, k'\}$ in the cover free family as its *color-set*. Then, it sets its new color $\phi_{new}(v) = q'$ for a $q' \in S_q$ such that q' is not in the color-set of any of the neighbors. □

Finally, we see how a repeated applicatin of the single-round color reduction of Lemma 4 allows us to get to an $O(\Delta^2 \log \Delta)$-vertex-coloring in $O(\log^* n)$ rounds, hence proving Theorem 1.

Proof of Theorem 1. The proof will be via iterative applications of Lemma 4. We start with the initial numbering of the vertices as a straightforward n-coloring. With one application of Lemma 4, we transform this into a $O(\Delta^2 \log n)$ coloring. With another application, we get a coloring with $O(\Delta^2(\log \Delta + \log \log n))$ colors. With another application, we get a coloring with $O(\Delta^2(\log \Delta + \log \log \log n))$ colors. After $O(\log^* n)$ applications, we get a coloring with $O(\Delta^2 \log \Delta)$ colors. □

1.2 Take 2: Kuhn-Wattenhofer Coloring Algorithm

In the previous section, we saw an $O(\log^* n)$-round algorithm for computing a $O(\Delta^2 \log \Delta)$-coloring. In this section, we explain how to transform this into a $(\Delta + 1)$-coloring. We will first see a very basic algorithm that performs this transformation in $O(\Delta^2 \log \Delta)$ rounds, by

removing essentially one color in each round. Then, we see how with the addition of a small but clever idea of [KW06], this transformation can be performed in $O(\Delta \log \Delta)$ rounds. As the end result, we get an $O(\Delta \log \Delta + \log^* n)$-round algorithm for computing a $(\Delta+1)$-coloring.

1.2.1 Warm up: One-By-One color Reduction

Lemma 5. *Given a k-coloring ϕ_{old} of a graph with maximum degree Δ where $k \geq \Delta + 2$, in a single round, we can compute a $(k-1)$-coloring ϕ_{new}.*

Proof. For each node v such that $\phi_{old}(v) \neq k$, set $\phi_{new}(v) = \phi_{old}(v)$. For each node v such that $\phi_{old}(v) = k$, let node v set its new color $\phi_{new}(v)$ to be a color $q \in \{1, 2, \ldots, \Delta+1\}$ such that q is not taken by any of the neighbors of u. Such a color q exists, because $v$has at most Δ neighbors. The resulting new coloring ϕ_{new} is a proper coloring. □

Theorem 6. *There is a deterministic distributed algorithm in the* LOCAL *model that colors any n-node graph G with maximum degree Δ using $\Delta+1$ colors, in $O(\Delta^2 \log \Delta + \log^* n)$ rounds.*

Proof. First, compute an $O(\Delta^2 \log \Delta)$-coloring in $O(\log^* n)$ rounds using the algorithm of Theorem 1. Then, apply the one-by-one color reduction of Lemma 5 for $O(\Delta^2 \log \Delta)$ rounds, until getting to a $(\Delta+1)$-coloring. □

1.2.2 Parallelized Color Reduction

Lemma 7. *Given a k-coloring ϕ_{old} of a graph with maximum degree Δ where $k \geq \Delta + 2$, in $O(\Delta \lceil \log(\frac{k}{\Delta+1}) \rceil)$ rounds, we can compute a $(\Delta+1)$-coloring ϕ_{new}.*

Proof. If $k \leq 2\Delta + 1$, the lemma follows immediately from applying the one-by-one color reduction of Lemma 5 for $k - (\Delta+1)$ iterations. Suppose that $k \geq 2\Delta+2$. Bucketize the colors $\{1, 2, \ldots, k\}$ into $\lfloor \frac{k}{2\Delta+2} \rfloor$ buckets, each of size exactly $2\Delta+2$, except for one last bucket which may have size between $2\Delta+2$ to $4\Delta+3$. We can perform color reductions in all buckets in parallel (why?). In particular, using at most $3\Delta+2$ iterations of one-by-one color reduction of Lemma 5, we can recolor nodes of each bucket using at most $\Delta+1$ colors. Considering all buckets, we now have at most $(\Delta+1)\lfloor \frac{k}{2\Delta+2} \rfloor \leq k/2$ colors. Hence, we managed to reduce the number of colors by a 2 factor, in just $O(\Delta)$ rounds. Repeating this procedure for $\lceil \log(\frac{k}{\Delta+1}) \rceil$ iterations gets us to a coloring with $\Delta+1$ colors. The round complexity of this method is $O(\Delta \lceil \log(\frac{k}{\Delta+1}) \rceil)$, because we have $\lceil \log(\frac{k}{\Delta+1}) \rceil$ iterations and each iteration takes $O(\Delta)$ rounds. □

Theorem 8. *There is a deterministic distributed algorithm in the* LOCAL *model that colors vertices of any n-node graph G with maximum degree Δ using $\Delta+1$ colors, in $O(\Delta \log \Delta + \log^* n)$ rounds.*

Proof. First, compute an $O(\Delta^2 \log \Delta)$-coloring in $O(\log^* n)$ rounds using the algorithm of Theorem 1. Then, apply the parallelized color reduction of Lemma 7 to transform this into a $(\Delta+1)$-coloring, in $O(\Delta \log \Delta)$ additional rounds. □

2 Edge-Coloring

In this section, we present an algorithm that computes a $(2\Delta-1)$-edge-coloring — that is, an assignment of colors $\{1, 2, \ldots, 2\Delta-1\}$ to edges such that no two edges that share an endpoint receive the same color — in $O(\Delta + \log^* n)$ rounds.

Theorem 9. *There is a deterministic distributed algorithm in the* LOCAL *model that colors edges of any n-node graph G with maximum degree Δ using $2\Delta-1$ colors, in $O(\Delta + \log^* n)$ rounds.*

The algorithm that proves this theorem is made of three parts, which we explain next.

Part I First we decompose the graph G into Δ spanning edge-disjoint graphs $F_1, F_2, \ldots, F_\Delta$, such that each F_i is an oriented pseudo-forest. That is, in the edge-set of $F_i = (V, E_i)$, each node v has at most one outgoing edge to a neighboring node u.

To compute this decomposition, first we orient the graph G arbitrarily, say by oriented each edge from the lower-ID endpoint to the higher-ID endpoint. Then, each node v numbers its outgoing edges 1, 2, ... Then, the pseudo-forest F_i is defined by including in it all the vertices, as well as all the edges which are numbered i^{th} by their starting point. Notice that F_i is an oriented pseudo-forest, i.e., each node v has at most one outgoing edge in F_i.

Part II Then, we compute a 3-vertex-coloring for each F_i, using the algorithm we saw in the first lecture, in $O(\log^* n)$ rounds. The coloring is computed for all pseudo-forests $F_1, F_2, \ldots, F_\Delta$ in parallel. We refer to these colors as *schedule-colors* of F_i.

Part III Finally, we process the forests one by one, spending $O(1)$ rounds on each. Consider an $i \in \{1, 2, \ldots, \Delta\}$, and suppose that we already have a $(2\Delta - 1)$-edge-coloring of the graph $H_{i-1} = \cup_{j=1}^{i-1} F_j$. For the first step, where $i = 1$, we use the convention that the graph H_{i-1} is the empty graph and thus no edge-coloring of it is needed.

We now process the edges of F_i in $O(1)$ rounds. In particular, we add the edges of F_i to the already edge-colored graph $H_{i-1} = \cup_{j=1}^{i-1} F_j$. We spend $O(1)$ rounds to compute a coloring for these edges of F_i in a way that is consistent with the already colored edges in H_{i-1}. We have three steps, corresponding to the three colors of the schedule-color of F_i computed in part II. We next discuss each of these steps.

In the k^{th} step, for $k \in \{1, 2, 3\}$, let E_k^i be the set of F_i-edges whose parent endpoint (the endpoint where the arrow ends) is colored with color k in the schedule-color we compute in Part II. Notice that these edges form vertex-disjoint stars. Consider one star centered at a node v and connecting it to neighboring nodes $u_1, u_2, \ldots, u_\ell$. We make the center v learn the colors used by edges adjacent to $u_1, \ldots, u_\ell$. Then, node v computes edge-colors for edges $\{v, u_1\}$, $\{v, u_2\}, \ldots, \{v, u_\ell\}$ in local manner. Each time, when trying to find a color for edge $\{v, u_i\}$, there will be one color available from colors $\{1, 2, \ldots, 2\Delta - 1\}$. This is because the edge $\{v, u_i\}$ has at most $2\Delta - 2$ incident edges and each of them can block at most one color. Node v can do this for all of its edges in the star. Moreover, the centers of different stars can work in parallel as they as the edges they want to color are non-adjacent. Hence, in $O(1)$ rounds, all edges of E_k^i get colored, and thus we can move to the next step.

References

[AS04] Noga Alon and Joel H Spencer. *The probabilistic method*. John Wiley & Sons, 2004.

[KW06] Fabian Kuhn and Rogert Wattenhofer. On the complexity of distributed graph coloring. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, pages 7–15. ACM, 2006.

[Lin87] Nathan Linial. Distributive graph algorithms global solutions from local data. In *Proc. of the Symp. on Found. of Comp. Sci. (FOCS)*, pages 331–335. IEEE, 1987.

[Lin92] Nathan Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992.