Advanced Distributed Algorithms **Spring 2019**

Lecture 3: April 18

Lecturer: Merav Parter

Randomized Vertex Coloring

In the distributed setting, we distinguish between three variants of the problem, with increasing level of complexity:

- $(\Delta + 1)$**-Coloring.** Given is a graph with maximum degree Δ, where each vertex v is given a palette $Pal(v) = \{1, \ldots, \Delta + 1\}$ of $\Delta + 1$ colors. In the output of the algorithm every vertex is colored with some color from its palette such that no two neighbors are colored with the same color.

- $(\Delta + 1)$**-List Coloring.** Same as above with the only distinction is that every vertex is given a palette of arbitrary $\Delta + 1$ colors. Thus vertices are allowed be given distinct color palettes.

- $(deg + 1)$**-List Coloring.** Same as $(\Delta + 1)$-List Coloring only that every vertex v is given a palette of $\deg(v) + 1$ colors rater than $\Delta + 1$.

Clearly, any algorithm for the $(deg+1)$-List Coloring also solves the weaker variants of $(\Delta+1)$-List Coloring and $(\Delta + 1)$-Coloring but not (necessarily) vice versa.

A Basic $(\Delta + 1)$ Coloring Algorithm. The algorithm has $O(\log n)$ iterations, in each iteration i, every vertex v that is still uncolored runs the following procedure:

BasicColorStep (for an uncolored vertex v)

- Let F_v be the set of colors taken by the colored neighbors of v (initially $F_v = \emptyset$).
- With probability $1/2$ set $c_v = 0$, and o.w pick c_v u.a.r from $\{1, \ldots, \Delta + 1\} \setminus F_v$.
- Send c_v to neighbors, if no other neighbor picked this color, set c_v as a permanent color.

Claim 3.1 *W.h.p. all vertices are colored after $O(\log n)$ application of Basic-Rand-Step.*

Proof: Fix a vertex v. We bound the probability that v gets colored in iteration i given that it was not colored in the previous iterations. Assume first that $c_v > 0$, and fix a neighbor u of v that is also not yet colored at the beginning of iteration i. We have that $\Pr[c_v = c_u] = 1/2 \cdot 1/(\Delta + 1 - |F_v|)$. This holds since w.p $1/2$ the color of c_u is at least 1, and the probability that v picked the color of $c_u \geq 1$ is $1/(\Delta + 1 - |F_v|)$. Since v has at most $\Delta + 1 - |F_v|$ uncolored neighbors at the beginning of iteration i, by the union bound, the probability that $c_v = c_u$ for *some* uncolored neighbor u is at most[1] $1/2$. Overall, the probability that v is colored is the probability that $c_v > 0$ and that $c_v \neq c_u$ for any u, which happens w.p. at least $1/4$.

The probability that a vertex v remains uncolored for $c \cdot \log n$ iterations is at most $(3/4)^{c \cdot \log n} \leq 1/n^{c'}$. The claim follows by applying the union bound over all the vertices. ■

[1]This is the point where having v picking $c_v = 0$ w.p. $1/2$ is crucial, otherwise we get that v is uncolored with probability of 1.

Sublogarithmic Randomized Coloring: We next turn to consider randomized coloring algorithms with sublogarithmic number of rounds. This calls for a totally new approach due to the *union bounded barrier* (see [BEPSv3]). Specifically, due to dependencies between vertices it is not so clear how to avoid applying the union bound to obtain a success guarantee for the entire graph. In particular, given a $O(1)$ round coloring procedure that colors a vertex with probability p, one needs to apply this procedure for $\Omega(\log_{1/p} n)$ rounds to guarantee that a vertex is colored with a polynomially large probability, i.e., probability of at least $1 - 1/n^c$. Such a strong guarantee per vertex is needed if one wants to apply the union bound over all n vertices. To overcome this barrier there are two ways out: (1) resort to deterministic algorithms and (2) require a smaller success guarantee per vertex (e.g., letting v being colored with probability of $1 - 1/\Delta^c$ rather than $1 - 1/n^c$). Indeed the sublogarithmic solutions that we have these days mix these two approaches and generally have the following two step procedure: a randomized phase which applies some randomized coloring rule for $o(\log n)$ rounds, and a deterministic phase that applies a deterministic coloring procedure on the remaining uncolored parts of the graph. Recall that in the first class we saw that network decompositions can be used to solve all classical local problems, and $(\Delta + 1)$ coloring in particular:

Lemma 3.2 (Deterministic Coloring) *For any n-vertex graph there is a deterministic algorithm that computes a $(\Delta + 1)$ coloring within $2^{O(\sqrt{\log n})}$ rounds.*

Proof: Compute a (d, c) network decomposition within r rounds for $c, d, r = 2^{O(\sqrt{\log n})}$. Then iterate over all c color classes of clusters. In step i, color all subgraphs of i-colored clusters by letting each vertex collecting the entire topology of its cluster along with the colors assigned to the neighbors of this cluster. Since all clusters of the same color have no neighbors in G, we can safely color those clusters in parallel. Overall, we have c iterations, each takes $O(d)$ rounds, which leads to a total round complexity of $O(c \cdot d) = 2^{O(\sqrt{\log n})}$. ■

This two step procedure is made efficient due to a miraculously phenomenon that occurs during the randomized part commonly referred to by *shattering*: at the end of the randomized procedure, the unsolved pieces are considerably "smaller"[2], thus the originally large graph is shattered (broken) into smaller pieces that can be solved faster deterministically. The reason for why we handle those pieces deterministically (i.e., instead of using a randomized algorithm) is due to the fact that we might have many small pieces, since the randomized algorithms succeed with probability which is polynomial in the size of the piece, that probability would be rather small. Moreover, we will also need to do a union bound over all pieces which might end up with a failure probability very close to 1.

$(1 + \epsilon)\Delta$ Coloring in $O(\sqrt{\log n}/\epsilon)$ Rounds. We first present an algorithm under the assumption that $\Delta \leq 2^{\sqrt{\log n}}$. The algorithm has $k = O(\sqrt{\log n}/\epsilon)$ iterations. In each iteration, a currently uncolored vertex v applies the following procedure:

RandColorStep (for an uncolored vertex v)

- Pick a color c_v u.a.r from the set of currently free colors in $Pal(v)$.
- Exchange c_v with neighbors, and if the color is free (i.e., not chosen by neighbors of v), assign the color to v.

We now claim that after the first phase the graph is shattered into components of diameter $O(\sqrt{\log n}/\epsilon)$.

Lemma 3.3 *W.h.p., after $O(\sqrt{\log n}/\epsilon)$ applications of Proc.* RandColorStep*, the diameter of any uncolored subgraph is $O(\sqrt{\log n}/\epsilon)$.*

Proof: The key observation is that in each iteration, a currently uncolored vertex v gets colored with probability of at least $\epsilon/2$ *regardless* of the color choices made by its neighbors. To see this observe that v has an extra subset of $\epsilon \cdot \Delta$ colors throughout the entire simulation. Thus regardless of the colors picked by its neighbors, there is always a subset of $\epsilon \cdot \Delta$ colors that are not chosen by any of its neighbors. The probability that v picks a color in this extra subset is at least $\epsilon \cdot \Delta/(1 + \epsilon)\Delta \geq \epsilon/2$.

[2]By smaller we mean that the structure of the unsolved part is easier, either because each unsolved component is small or with a small diameter etc., which makes it much easier to be solved deterministically.

We next consider a fixed path $P = [v_1, \ldots, v_\ell]$ for $\ell = \sqrt{\log n}/\epsilon$. We say that the path P is *uncolored* if all its vertices remain uncolored at the end of the first phase. The probability p that P is uncolored is at most:

$$(1-\epsilon/2)^{\ell \cdot k} \leq exp(-\ell \cdot k \cdot \epsilon/2) = exp(-c \cdot \log n/\epsilon).$$

Note that in the above we use the fact that each vertex v is colored w.p. at least $\epsilon/2$ independently to the choices of its neighbors. In addition, note that in each iteration we use fresh coins. To complete the proof we need to show that w.h.p. there is no *uncolored* path. For that purpose, we bound the total number of ℓ-length paths and apply the union bound. Clearly, there are at most $n \cdot \Delta^{\ell-1} = n \cdot 2^{\log n/\epsilon}$ paths. Thus the probability that there exists an uncolored path is at most $p \cdot n \cdot 2^{\log n/\epsilon} \leq 1/n^{c'}$ upon setting the constant c large enough. ■

In the second phase of the algorithm, every uncolored vertex collects its $O(\sqrt{\log n}/\epsilon)$ ball in G, i.e., it collects its entire component in the remaining uncolored graph. It then applies *locally* a deterministic coloring algorithm for its uncolored component. We therefore conclude:

Lemma 3.4 *For every graph G with maximum degree $\Delta \leq 2^{\sqrt{\log n}}$ there is a randomized algorithm that computes a $(1+\epsilon)\Delta$ coloring within $O(\sqrt{\log n}/\epsilon)$ w.h.p.*

We now extend this algorithm for graphs with large maximum degree where $\Delta \geq 2^{\sqrt{\log n}}$.

- Partition G into $k = \Theta(\epsilon^2 \cdot \Delta/\log n)$ vertex-disjoint subgraphs, by letting each vertex pick a subgraph u.a.r in $[1, k]$.
- Allocate a disjoint subset of $\Delta' = \Delta/k \cdot (1+\epsilon)$ colors to each subgraph.
- Apply the algorithm of Lemma 3.4 on each subgraph G_i in parallel (using the colors allocated to that subgraph).

Observation 3.5 *(1) W.h.p. the maximum degree of each subgraph G_i is $\Delta_i \leq \Delta/k(1+\epsilon/3)$.*
(2) $\Delta' \geq (1+\epsilon/2) \cdot \Delta_i = O(\log n)$.

Proof: (1) follows by a direct application of the Chernoff bound by taking $k = c \cdot \epsilon^2 \cdot \Delta/\log n$ for a large enough constant c. To see (2), observe that $\Delta' \geq (1+\epsilon/2) \cdot (1+\epsilon/3) \cdot \Delta/k$. ■

By Obs. 3.5(2), we get that indeed the $(1+\epsilon')\Delta$ algorithm of Lemma 3.4 can be applied on each G_i by taking $\epsilon' = \epsilon/2$.

Deg +1 Coloring in $\log \Delta + 2^{O(\log(\Delta \log n))}$ Rounds. Finally, we turn to consider an even more drastic illustration of the shattering phenomenon. The algorithm that we presented in class is a weaker version of the following seminal result:

Theorem 3.6 *[BEPSv3] There is a randomized algorithm that computes a $(deg+1)$ list coloring within $O(\log \Delta) + 2^{\sqrt{\log \log n}}$ rounds w.h.p.*

The state of the art for the weaker variant of $(\Delta+1)$ list coloring problem is $\log^* n + 2^{\sqrt{\log \log n}}$ by Chan and Pettie [CLP18]. As we will see in the next class, improving upon this result boils down to improving the deterministic algorithm for network decomposition. In the last class, we presented the less ambitious bound of $\log \Delta + 2^{O(\log(\Delta \log n))}$ that already captures the key aspects of the shattering effect.

The Algorithm. The algorithm has two phases. A randomized part that consists of $O(\log \Delta)$ rounds, and a deterministic part that consists of $2^{O(\log(\Delta \log n))}$ rounds. See also Sec. 5.1 of [BEPSv3].

The first part contains $O(\log \Delta)$ applications of Proc. RandColorStep that is run by any vertex v that is not yet colored[3]. In the analysis, we later show that at the end of the first phase, the size of each uncolored component is $\ell = O(\Delta^2 \cdot \log_\Delta n)$. In the second phase, the deterministic algorithm of Lemma 3.2 is applied on each connected component in the remaining unsolved graph, which leads to a round complexity of $2^{\sqrt{\log \ell}}$.

[3]In class we showed another procedure which breaks tie to the favor of vertex of largest ID, as used in [BEPSv3]. However, to obtain the weaker bound of $\log \Delta + 2^{O(\log(\Delta \log n))}$, it is sufficient to use Proc. RandColorStep.

Lemma 3.7 *The probability that a vertex gets colored after a single application of Procedure* RandColorStep *is at least* $1/4$.

Proof: Consider a vertex v that is not yet colored at the beginning of iteration i. We will show that v gets colored after applying Procedure RandColorStep with probability of at least $1/4$. Let $U \subset V$ be the set of uncolored vertices at the beginning of iteration i. We denote by $N_U(v) = N(v) \cap U$ to be all neighbors of v in U. Note that whether v gets colored or not, depends only at the decision made by the vertices in $N_U(v)$. Finally, for every $u \in U$, let $Pal(u)$ denote the current palette of free colors of vertex u at the beginning of iteration i (i.e., $Pal(u)$ contains all colors in the original palette of u excluding the colors of u's neighbors that got colored up to iteration i).

To show that v gets colored with probability of at least $1/4$, we will show that in expectation over all random choices made by the neighbors of v in $N_U(v)$, v has at least $|Pal(v)|/4$ free colors (i.e., that are not selected by its neighbors in $N_U(v)$). Fix a color q in $Pal(v)$ and let $Comp(q) = \{u \in N_U(v) \mid q \in Pal(u)\}$ be the neighbors of v in $N_U(v)$ that compete with v on the color q. Since each vertex $u \in Comp(q)$ picks the color q with probability of $1/|Pal(u)|$, the *weight* of the competition over the color q is defined by:

$$w(q) = \sum_{u \in Comp(q)} 1/|Pal(u)| \ .$$

Let X_q be the indicator random variable that the color q is free after exposing the colors chosen by $N_U(v)$. The probability that q is free is then given by:

$$\begin{aligned}\Pr[X_q = 1] &= \prod_{u \in Comp(q)} (1 - 1/|Pal(u)) \geq \prod_{u \in Comp(q)} (1/4)^{1/|Pal(u)} \\ &= (1/4)^{\sum_{u \in Comp(q)} 1/|Pal(u)} = (1/4)^{w(q)} \ ,\end{aligned}$$

where the first inequality follows by noting that $|Pal(u)| \geq 2$, and using the inequality of $(1-x) \geq (1/4)^x$ for $x \in [0, 1/2]$.

Let $X = \sum_{q \in Pal(v)} X_q$. We then that the expected number of free colors is at least

$$\begin{aligned}\mathsf{Exp}[X] &\geq \sum_{q \in Pal(v)} (1/4)^{w(q)} \geq |Pal(v)| \cdot (1/4)^{\sum_{q \in Pal(v)} w(q)/|Pal(v)|} \\ &\geq |Pal(v)| \cdot (1/4)^{\deg_U(v)/|Pal(v)|} > |Pal(v)|/4 \ .\end{aligned}$$

The second inequality follows by the convexity of the exponential function where the value is minimized when all the colors have the same weight, and the third inequality where $\sum_{q \in Pal(v)} w(q) \leq \deg_U(v)$ is due to the fact that each $u \in \deg_U(v)$ contributes at most one unit to $\sum_{q \in Pal(v)} w(q)$. We have that v has at least $|Pal(v)|/4$ free colors in expectation, and thus it is colored with probability of at least $1/4$. The lemma follows. ■

We now turn to show the key shattering property of this algorithm which allows the efficient determenstic coloring of the remaining uncolored parts.

Lemma 3.8 *At the end of the first phase, w.h.p. the size of each uncolored component is at most* $t \cdot \Delta^2$ *for* $t = c \cdot \log_\Delta n$.

Proof: First observe that the probability of a vertex v to get colored in a single application of the randomized coloring step depends only on the random coins of its immediate neighbors. Therefore for two currently uncolored nodes at distance at least 3 in G, it holds that their probabilities to get colored in that particular coloring step are independent. Since in each application of the coloring step, we use fresh set of coins, it holds that the probability of a vertex to get colored at some point during the algorithm also depends only on the coins of its immediate neighbors. We consider a set of nodes T with the following properties: (i) T is a 3-independent set (i.e., $\texttt{dist}(u, v, G) \geq 3$ for every $u, v \in T$), (ii) $|T| = t$, and (iii) T span a tree in the uncolored part[4] of G^3. The proof will go as follows. First, we will bound the probability that such a fixed set T exists at the end of the first phase. Then, we will bound the total number of such sets T that satisfy the above mentioned three properties. This number will be sufficiently small, so that we will be able to apply the union bound and deduce that w.h.p. no such set T exists. Finally, we will use this fact to conclude that the size of each uncolored component is small.

We start by bounding the probability that there exists a fixed set T of t vertices that are uncolored at the end of the first phase. By Lemma 3.7, the probability that a single vertex $v \in T$ is uncolored after $c \cdot \log \Delta$ applications of the coloring step is $(3/4)^{c \log \Delta}$. Since every two vertices in T are at distance at least 3, the probability that all these vertices are uncolored after $c \cdot \log \Delta$ iterations is $p = (3/4)^{c \log \Delta \cdot t} \leq 1/n^{c'}$ for large enough $c' > 5$.

We next bound the total number of such T sets. To do that, we first bound the total number of rooted *unlabeled* trees of t nodes by 4^t. To see this bound, observe that any rooted unlabeled tree can be uniquely encoded by a binary vector of length $2t$ that describes the Euler tour of this tree (i.e., start the Euler traversal from the root, a step up is indicated by putting 0 and a step down is indicted by putting 1 in the corresponding position in the $2t$-length vector). Overall there are $2^{2t} = 4^t$ distinct binary vectors. We then have n possible choices for picking the root, and there are Δ^3 options for choosing the i^{th} vertex (i.e., since its parent in the tree, which has already been selected, has at most Δ^3 neighbors in G^3). Overall there are $4^t \cdot n \cdot (\Delta)^{3t} \leq n^5$. By applying the union bound, we get that the probability that such a set T exists is at most $p \cdot n^5 \leq 1/n^{c'-5}$.

We are now ready to complete the proof of the lemma. Assume towards contradiction that there exists an uncolored component C of size $t \cdot \Delta^2$. We will use this set to generate a set $T \subset C$ that satisfies the three properties. Since we showed that w.h.p. no such set exists, we will end up with a contradiction. Set $T \leftarrow \emptyset$, pick an arbitrary node $v_0 \in C$, and add it to T. Eliminate all its 2-hop neighbors from C and add to T some vertex v_1 which is at distance exactly 3 from v_0 (and thus $(v_0, v_1) \in G^3$). Continue this way until C gets empty. Observe that each time that we add a vertex to T, we eliminate at most Δ^2 vertices from C, therefore when C is empty, the size of T is at least t. One can verify that indeed T satisfies all the properties (i-iii). ■

References

[BEPSv3] Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. In *Foundations of Computer Science (FOCS) 2012*, pages 321–330. IEEE, 2012, also coRR abs/1202.1983v3.

[CLP18] Yi-Jun Chang, Wenzheng Li, and Seth Pettie. An optimal distributed (δ+ 1)-coloring algorithm? In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 445–456. ACM, 2018.

[4]The power graph G^i is the graph obtained by drawing an edge between u and v iff $\texttt{dist}(u, v, G) \leq i$.