6.5240 Sublinear Time Algorithms November 21, 2022

Lecture 20

Lecturer: Ronitt Rubinfeld *Scribe: Yuchong Pan*

1 Greedy List Coloring

Today, we discuss a sublinear time algorithm for graph coloring.

Definition 1. A *proper c-coloring* of a graph $G = (V, E)$ assigns a color c_v from a *palette* (e.g., $\{1, \dots, C\}$) to each $v \in V$ such that $c_u \neq c_v$ for all $(u, v) \in E$.

In this lecture, we assume that the maximum degree of a vertex in G is Δ, and $C = \Delta + 1$ (or maybe $C = 2\Delta$). Note that $(\Delta + 1)$-coloring is easy via greedy. We call this greedy algorithm GreedyListColoring and give it in Algorithm 1. Note that GreedyListColoring takes $O(|E|)$ time.

foreach $v \in V$ **do**
  $L(v) \leftarrow \{1, \dots, \Delta + 1\}$
foreach $v \in V$ *(in an arbitrary order)* **do**
  if $L(v) = \emptyset$ **then**
    return *fail*
  else
    $c_v \leftarrow$ any color in $L(v)$
    remove c_v from $L(u)$ for all neighbors u of v

Algorithm 1: A greedy algorithm, called GreedyListColoring, for finding a $(\Delta + 1)$-coloring in a graph $G = (V, E)$.

The model we consider supports the following three types of queries on a graph $G = (V, E)$:

- *Degree queries*: Given $u \in V$, what is $\deg(u)$?
- *Pair queries*: Given $u, v \in V$, is it true that $(u, v) \in E$?
- *Neighbor queries*: Given $u \in V$ and $k \in \mathbb{N}$, what is the k^{th} neighbor of u?

2 Palette Sparsification

We introduce a technique called *palette sparsification* to improve the running time of graph coloring. The goal is to prove the following theorem.

Theorem 2. *One can find a $(\Delta + 1)$-coloring of an n-vertex graph in $\tilde{O}(n\sqrt{n})$ time.*

The idea of *palette sparsification* is the following:

For each vertex $v \in V$ in an n-vertex graph $G = (V, E)$, sample $k = \Theta(\log n)$ colors from $\{1, \dots, \Delta + 1\}$ to get $L(v)$. (1)

The following lemma is the main observation, which we state without giving a proof. In Section 4, we prove a weakened version which relaxes $(\Delta + 1)$-coloring to 2Δ-coloring.

Lemma 3. *With high probability, a graph $G = (V, E)$ can be colored by (1) such that $c_v \in L(v)$ for all $v \in V$ via* GreedyListColoring.

In what follows, we denote $G_{\text{sparse}} = (V, E_{\text{sparse}})$, where $E_{\text{sparse}} = \{(u, v) \in E : L(u) \cap L(v) \neq \emptyset\}$. Figure 1 gives an example of G_{sparse}.

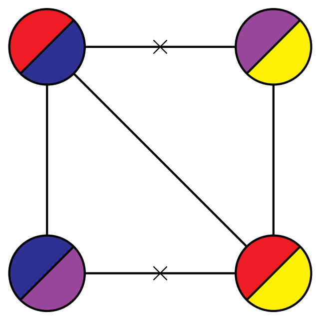

Figure 1: An example of palette sparsification, where edges absent from G_{sparse} are crossed off, and each vertex is colored by the colors in its palette.

It turns out that G_{sparse} does not contain many edges.

Lemma 4. *With high probability, $|E_{sparse}| = O(n \log^2 n)$.*

Proof. Fix $u \in V$. Without loss of generality, suppose that $L(u) = \{1, \ldots, k\}$. For all $v \in N(u)$ and $i \in \{1, \ldots, k\}$, set

$$X_{v,i} = \begin{cases} 1, & \text{if } i \in L(v), \\ 0, & \text{otherwise.} \end{cases}$$

Let

$$X = \sum_{i=1}^{k} \sum_{v \in N(u)} X_{v,i}.$$

Since $\sum_{v \in N(u)} X_{v,i}$ is the number of edges due to color i, then X is an upper bound on $\deg(u)$ in G_{sparse}. Note that $\mathbb{E}[X_{v,i}] = k/(\Delta + 1)$ for all $v \in N(u)$ and $i \in \{1, \ldots, k\}$. Hence,

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{k} \sum_{v \in N(u)} X_{v,i}\right] = \sum_{i=1}^{k} \sum_{v \in N(u)} \mathbb{E}[X_{v,i}] \leq k \cdot \Delta \cdot \frac{k}{\Delta + 1} < k^2.$$

Since $k = \Theta(\log n)$, then $\mathbb{E}[\deg(u)] \leq O(\log^2 n)$. (One can show "with high probability" with additional work.) $\square$

3 A Sublinear Time Algorithm for Graph Coloring

Assuming Lemma 3, we give a sublinear time algorithm for finding a $(\Delta + 1)$-coloring of a graph:

1. Construct the palette of each vertex using (1).
2. Construct G_{sparse}: For each color $c \in \{1, \ldots, \Delta + 1\}$, find $X_c = \{v \in V : c \in L(v)\}$ (do this while doing step 1). Query all pairs of vertices in each X_c to find E_{sparse}: suppose that $X_c = \{v_{i_1}, \ldots, v_{i_\ell}\}$; for distinct $j, k \in [\ell]$, query if $(v_{i_j}, v_{i_k}) \in E$, and if so, add it to E_{sparse}.
3. Perform GreedyListColoring on G_{sparse}.

Step 1 takes $O(n \log n)$ time. Step 3 takes $O(|E_{\text{sparse}}|) = O(n \log^2 n)$ time. For each color c,

$$\mathbb{E}\left[|X_c|^2\right] = \sum_{\substack{u,v \in V \\ u \neq v}} \mathbb{E}\left[\mathbb{1}_{u,\, v \text{ both choose color } c}\right] = \binom{n}{2}\left(\frac{k}{\Delta+1}\right)^2 = O\left(\frac{n^2 \log^2 n}{\Delta^2}\right).$$

Hence, the running time to query all pairs in each X_c is at most

$$(\Delta+1) \cdot O\left(\frac{n^2 \log^2 n}{\Delta^2}\right) = \tilde{O}\left(\frac{n^2}{\Delta}\right).$$

It follows that the total running time is $\tilde{O}(n^2/\Delta)$. This proves Theorem 2.

Proof of Theorem 2. If $\Delta \leq \sqrt{n}$, then we use GreedyListColoring with $O(|E|) \leq O(n\Delta) \leq O(n\sqrt{n})$ time. If $\Delta > \sqrt{n}$, then we use palette sparsification with $\tilde{O}(n^2/\Delta) \leq \tilde{O}(n^2/\sqrt{n}) = \tilde{O}(n^{3/2})$ time. □

4 Relaxing Lemma 3 to 2Δ-Coloring

In this section, we prove a weakened version of Lemma 3 which relaxes $(\Delta+1)$-coloring to 2Δ-coloring. Other parts in the proof of Theorem 2 remain the same, hence giving an $\tilde{O}(n^{3/2})$ time algorithm for finding a 2Δ-coloring of an n-vertex graph.

Lemma 5. *With high probability, a graph $G = (V, E)$ can be colored by* (1) *with $\Delta+1$ replaced by 2Δ such that $c_v \in L(v)$ for all $v \in V$ via* GreedyListColoring*.*

Proof. When attempting to color a vertex v, we say that a color $c \in \{1, \ldots, 2\Delta\}$ is *good* if $c \in L(v)$ initially and c is not used to color any *previous* neighbor of v. If $L(v)$ contains any good color c, then we can color v successfully. Since $L(v)$ is chosen independently of its neighbors, we can think of choosing $L(v)$ "now." Since v has at most Δ neighbors, then

$$\Pr[L(v) \text{ contains no good color}] \leq \frac{\binom{\Delta}{k}}{\binom{2\Delta}{k}} = \frac{\frac{\Delta(\Delta-1)\cdots(\Delta-k+1)}{k!}}{\frac{(2\Delta)(2\Delta-1)\cdots(2\Delta-k+1)}{k!}} < \frac{1}{2^k} = \left(\frac{1}{2}\right)^{\Theta(\log n)} = \frac{1}{n^\alpha},$$

for some constant α. By the union bound,

$$\Pr[\text{there exists a vertex } v \text{ such that } L(v) \text{ has no good color}] \leq \frac{1}{n^{\alpha'}},$$

for some constant α' (e.g., $\alpha' = 3$). Hence, with high probability, the algorithm never fails. It follows that G has a legal list coloring with high probability. □

We note that the proof of Lemma 5 indeed generalizes to $(1+\delta)\Delta$-coloring for any constant $\delta > 0$ which does not depend on Δ.