6.5240 Sublinear Time Algorithms Oct 7, 2024

Lecture 10

Lecturer: Ronitt Rubinfeld *Scribe: Dingding Dong*

Today's goal is to finish the construction of LCA for the Maximal Independent Set (MIS) problem. We first recall the definition of LCA.

Definition 1 *$\mathcal{A}$ is a "local computation algorithm" (LCA) for Π if:*

- *it is given probe access to input X, random bits r and local memory;*
- *it answers queries to bits/words of $y \in \Pi(x)$, where local memory (not random bits) is "wiped clean" between queries*
- *for all queries, LCA answers concistently with some $y \in \Pi(x)$ with high probability over r.*

We now recall the definition of Maximal Independent Set (MIS).

Definition 2 *Let $G = (V, E)$ be a graph. We say that a vertex subset $U \subseteq V$ is a maximal independent set (MIS) if:*

1. *(U is independent) for all $u_1, u_2 \in U$, we have $\{u_1, u_2\} \notin E$;*
2. *(maximality) there is no $w \in V \setminus U$ such that $U \cup \{w\}$ is independent.*

We assume that G has maximum degree Δ and is represented by its adjacency list.

Today we'll look at an LCA that does not have ideal time, but it will demonstrate important techniques that are useful for building LCA. One of our ways to build good algorithm is to have distributed algrithms that have few rounds. We recall the well-known "Luby's" distributed algorithm for MIS (which is not limited to constant degree graphs).

Algorithm 1 Luby's distributed algorithm for MIS

1: MIS $\leftarrow \emptyset$ # all nodes are set to "active"
2: **for** k rounds **do**
3: in parallel, all nodes (both active and inactive) v color itself red with probability $\frac{1}{2\Delta}$ and blue with probability $1 - \frac{1}{2\Delta}$, and send its color to all neighbors
4: if active node v is red and all (both active and inactive) neighbors are blue, add v to MIS and remove v and its active neighbors from graph (i.e, set them to "inactive")

We hope that the above algorithm gives an MIS after few rounds. To this end we will define what we call Luby's status. Suppose we sequentially simulate Luby's algorithm for $k = \Theta(\Delta \log \Delta)$ rounds. Luby's status is an indicator that whether each vertex has been decided yet.

Definition 3 *Suppose we sequentially simulate Luby's algorithm for $k = \Theta(\Delta \log \Delta)$ rounds. At the end each of v us is one of (1) live, (2) in MIS, and (3) not in MIS (meaning that some neighbor of v is in MIS).*

To show that we will be done after k rounds, we will need the following main lemma/observation.

Lemma 4 *Observe that in each round, the probability that v is added to MIS is at least*

$$\frac{1}{2\Delta}\left(1 - \frac{1}{2\Delta}\right)^{\Delta} \geq \frac{1}{4\Delta}.$$

Thus for all live v, $\mathbb{P}[v \text{ still lives after } k := 4k'\Delta \text{ rounds}] \geq (1 - \frac{1}{4\Delta})^{4k'\Delta} \leq e^{-k'} \leq e^{-\Theta(\log \Delta)} \leq \frac{1}{\Delta^c}$. Thus if $k = \Theta(\Delta \log \Delta)$, then we have $k' = \Theta(\log \Delta)$. We can pick $c > 0$ arbitrarily large so that the above probability is small.

We now introduce the LCA for MIS that is based on the above.

Algorithm 2 LCA for MIS

Input: variable $v \in V$
Output: "in" or "out" indicating whether v lies in MIS

1: Run sequential version of LubyStatus(v)
2: **if** LubyStatus(v) is "in" or "out" **then**
3: Output the answer and halt
4: do BFS to find the connected component of live nodes that contains v — size of connected component times $\Delta^{O(\Delta \log \Delta)}$
5: compute the lexicographically first MIS M' for that connected component — O(size of component)
6: output whether v is in/out of M'

We analyze the efficiency of the above algorithm.

- Line 1 takes $\Delta^{O(\Delta \log \Delta)}$ time,
- Line 4 takes $\Delta^{O(\Delta \log \Delta)} \cdot$ (size of connected component) time,
- Line 5 takes (size of connected component) time.

To compute the efficiency, we will show that live vertices happen to lie in small connected components of size $O(\log n)$ in the graph. Note that boundary of connected component are blue nodes, because if they are red, they will pull out their neighbors to be inactive.

Theorem 5 *(Bounding size of connected component) After $O(\Delta \log \Delta)$ rounds, all connected component of survivors are of size $O(\mathrm{poly}(\Delta) \log n)$.*

Proof We first define (and compare) two variables. We define

$$A_v = \begin{cases} 1 & v \text{ survives all rounds} \\ 0 & \text{otherwise} \end{cases}.$$

We also define

$$B_v = \begin{cases} 1 & \text{if there is no round such that } v \text{ is red and no } w \in N(v) \text{ is red} \\ 0 & \text{otherwise} \end{cases}.$$

Observe that $B_v = 0$ implies $A_v = 0$ (or equivalently, $A_v = 1$ implies $B_v = 1$). The good thing is that B_v is independent between rounds. So we can compute

$$\mathbb{P}[A_v = 1] \leq \mathbb{P}[B_v = 1] \leq \left(1 - \frac{1}{4\Delta}\right)^{c\Delta \log \Delta} \leq \frac{1}{8\Delta^3}.$$

Moreover, each B_v depends on Δ other B_w's. We will show that these nodes are unlikely to simultaneously survive. To do so we will use the so called graph shattering, which is an importantic technique in the algorithmic version of Lovász local lemma and distributed algorihtms.

We will construct the following graph $G^{(3)}$, defined by

$$V = \{B_v : v \in G\}, \qquad E = \{B_u B_v : \mathrm{dist}(u, v) = 3\}.$$

Observe that the maximum degree of $(G^{(3)})$ is at most Δ^3.

Observation 6 *The number of connected components in $G^{(3)}$ of size w is at most the number of size w subtrees in $G^{(3)}$. This is because we map can each connected component C to an arbitrary spanning tree of C, and this map is 1-1.*

Claim 7 *For every live connected component S in G, $G^{(3)}$ contains a tree with vertex set T as a subgraph, where*

1. $|T| \geq \frac{|S|}{\Delta^2+1}$,
2. $dist_G(u,v) \geq 3$ *for all* $u, v \in T$ *(so* B_u, B_v *are independent!).*

Proof We can pick T greedily as follows:

- Pick $v \in S$
- Repeat the following until S is empty:
  - Move v from S to T
  - remove u with $\text{dist}(u,v) \leq 2$ from S
  - pick new $v \in S$ such that $\text{dist}(u,v) = 3$ for some $u \in T$

Observe that by adding every $v \in T$, we remove $\leq \Delta^2$ nodes from S plus v itself, so totally at most $\leq \Delta^2 + 1$ nodes. This guarantees that $|T| \geq \frac{|S|^2}{\Delta+1}$. ■

We also utilize the following theorem in graph theory.

Theorem 8 *The number of size w subtrees in n-node graph of degree $\leq D$ is $\leq N \cdot (4D)^w$.*

We can now prove the theorem. Let $s = \log(3n)$ and

$$\mathcal{T}_s = \{T \subseteq V : |T| = s,\ T \text{ connected in } G^{(3)}, \text{ all } u, v \text{ have dist} \geq 3 \text{ in } G\}.$$

We have

$$\begin{aligned}
&\mathbb{P}[\text{there exists a connected component } S \text{ of size } (\Delta^2+1)s \text{ that survives}] \\
&\leq \mathbb{P}[\text{there exists } T \in \mathcal{T}_s \text{ such that } A_v = 1 \text{ for all } v \in T] \\
&\leq |\mathcal{T}_s| \left(\frac{1}{8\Delta^3}\right)^s \leq n(4\Delta^3)^s \cdot \frac{1}{(8\Delta^3)^s} = \frac{n}{2^s} = \frac{1}{3}.
\end{aligned}$$

Thus, with probability $\geq 2/3$, all surviving connected components have size $(\Delta^2+1)\log(3n) = O(\Delta^2 \log n)$. ■