6.5240 Sublinear Time Algorithms October 1, 2026

Lecture 7

Lecturer: Ronitt Rubinfeld *Scribe: Vasiliki Efthymiou*

Agenda

- Finish the local computation algorithm (LCA) for MIS.
- Greedy algorithm ⇒ sublinear algorithm *(if time permits)*.

1 Setting and notation

Let $G = (V, E)$ be a graph on $n = |V|$ vertices with maximum degree at most Δ (we assume $\Delta \geq 2$). $N(v)$ denotes the set of neighbors of v, $N[v] = N(v) \cup \{v\}$, and $\text{dist}(u, v)$ is the shortest-path distance in G. All logarithms are base 2. A *maximal independent set* (MIS) is an independent set that cannot be extended by any further vertex.

An LCA for MIS receives a query vertex v and must say whether v belongs to the MIS; answers to different queries must be consistent with one and the same MIS, and each query should be answered in time sublinear in n.

2 Luby's algorithm in "status" form

Throughout, every vertex is in one of two states: *alive* or *inactive*. A vertex becomes inactive when it joins the MIS or when one of its neighbors does.

Algorithm 1 LUBY-STATUS

Input: graph $G = (V, E)$ with maximum degree at most Δ
Output: for every $v \in V$ a status $\in \{$"alive", " $\notin$ MIS", "in MIS"$\}$

```
1: MIS ← ∅
2: set every v ∈ V to "alive"
3: for c Δ log Δ rounds do
4:     every v ∈ V colors itself red w.p. p = 1/(2Δ), blue otherwise        ▷ even if v is inactive
5:     for every alive vertex v do
6:         if v is red and all neighbors of v are blue then
7:             MIS ← MIS ∪ {v}
8:             set v and N(v) to "inactive"
9:         end if
10:    end for
11: end for
12: return the status of every v
```

The number of rounds is $O(\Delta \log \Delta)$; the analysis in Section 4 uses $c \geq 20$. A vertex that is inactive but not in MIS has status "$\notin$ MIS".

Remark [Color independently of being alive] Every vertex draws its color in every round *independently of whether it is still alive*. This is what later makes the colors a fixed table of random bits that does not depend on the history of the algorithm.

Time. Computing the status of a single vertex v *locally* takes time $\Delta^{O(\Delta \log \Delta)}$: the status of v after $t = c\Delta \log \Delta$ rounds depends only on the colors in the radius-$O(t)$ ball around v, which contains at most $\Delta^{O(t)}$ vertices.

3 An LCA for MIS

Algorithm 2 LCA-MIS(v)

Input: a vertex $v \in V$
Output: whether v belongs to the MIS

1: $\sigma \leftarrow$ status of v from the Parnas–Ron sequential version of Luby-Status(v)
2: **if** $\sigma \in \{$" $\notin$ MIS", " $\in$ MIS"$\}$ **then**
3: **return** σ
4: **end if**
5: run BFS(v), calling Luby-Status(w) at each newly visited vertex w
6: $U \leftarrow$ the connected component of alive nodes containing v found by the BFS
7: $M' \leftarrow$ lexicographically first MIS of U
8: **return** "in" if $v \in M'$, and "out" otherwise

Remark [Why the output is a consistent MIS] A vertex that is alive at the end of Luby-Status has no neighbor in MIS (otherwise it would have been made inactive), and two alive vertices that are adjacent lie in the same alive component. So taking the lexicographically first MIS inside each alive component and adding it to MIS yields an independent set; it is maximal because every other vertex is in MIS or adjacent to it.

3.1 A note on the "randomness" of Luby-Status

All queries must see the *same* random choices. We therefore think of the randomness as a table of random bits that is fixed once and for all: the entry in row r and column v determines the color of vertex v in round r.

	v_1	v_2	$\cdots$	v_n
round 1				
round 2				
$\vdots$				
round $c\Delta \log \Delta$				

Fix the table of random bits!

3.2 The complexity of the LCA

If the status of v was "alive", we pay

$$\underbrace{\text{size of connected component}}_{\#\text{ nodes visited in BFS}} \times \underbrace{\Delta^{O(\Delta \log \Delta)}}_{\text{cost of Luby-Status per node call}}.$$

Hence the question is:

How large is the connected component of alive nodes?

4 How large is an alive component?

Claim 1 *After $c\,\Delta \log \Delta$ rounds, with probability at least $2/3$ (over the random table), every connected component of alive nodes has size at most $(\Delta^2+1)\lceil \log 3n \rceil = O(\text{poly}(\Delta) \log n)$.*

Plugging this into the bound of the previous section, a query costs $\Delta^{O(\Delta \log \Delta)} \cdot O(\log n)$ time.

Proof of Claim 1. Fix the random table and define, for every $v \in V$,

$$A_v = \begin{cases} 1, & \text{if } v \text{ survives (stays alive) until the end,} \\ 0, & \text{otherwise,} \end{cases}$$

$$B_v = \begin{cases} 1, & \text{if there is } \textit{no} \text{ round in which } v \text{ is red and all its neighbors are blue,} \\ 0, & \text{otherwise.} \end{cases}$$

Observation 2 *$B_v = 0$ means that in some round v was red and all its neighbors were blue. Then either v itself got into the MIS, or a neighbor of it had already gotten into the MIS and made v inactive. In both cases v is inactive at the end. In particular,*

$$A_v = 1 \implies B_v = 1,$$

i.e. "v stayed alive" implies "v did not become inactive".

Observation 3 *For $c \geq 20$,* $\Pr[B_v = 1] \leq \dfrac{1}{8\Delta^3}$.

Proof In a fixed round, v is red and all (at most Δ) neighbors are blue with probability at least $\frac{1}{2\Delta}\left(1 - \frac{1}{2\Delta}\right)^{\Delta} \geq \frac{1}{2} \cdot \frac{1}{2\Delta}$. Rounds use independent bits, so

$$\Pr[B_v = 1] \;\leq\; \left(1 - \frac{1}{2} \cdot \frac{1}{2\Delta}\right)^{c\Delta \log \Delta} \;\leq\; \frac{1}{8\Delta^3} \qquad \text{for } c \geq 20.$$

(The last step uses $1 - x \leq e^{-x}$ and $\Delta \geq 2$.) ■

Observation 4 (Dependencies) *B_v is determined by the colors that the vertices of $N[v]$ choose in all rounds. Consequently B_u and B_v share no randomness, and are independent, as soon as* $\mathrm{dist}(u, v) \geq 3$. *For the path $v - w - z - u$:*



- *B_v is not independent of B_w (adjacent);*
- *B_v is not independent of B_z (they share the immediate neighbor w);*
- B_v is *independent of B_u.*

Moreover, since $\deg \leq \Delta$, *each B_u depends on at most another Δ^2 of the B_v's (those with* $\mathrm{dist}(u, v) \leq 2$*).*

4.1 Proof sketch

- Any large connected component has lots of nodes at pairwise distance $\geq 3 \Rightarrow$ independent B_v's.
- These independent nodes are unlikely to survive.

The first point is made precise in Subsection 4.2 (a tree of far-apart nodes inside every large component), the second in Subsection 4.3 (a union bound over all such trees).

4.2 The graph $G^{(3)}$

Definition 5 *$G^{(3)}$ is the graph on the same vertex set V as G, whose edge set $E^{(3)}$ is defined by*

$$(u, v) \in E^{(3)} \iff \mathrm{dist}_G(u, v) = 3.$$

The vertices u, v of such an edge are as close as possible while having independent $B_u \perp B_v$. Every vertex has at most Δ^3 vertices at distance exactly 3, so $G^{(3)}$ has maximum degree at most Δ^3.

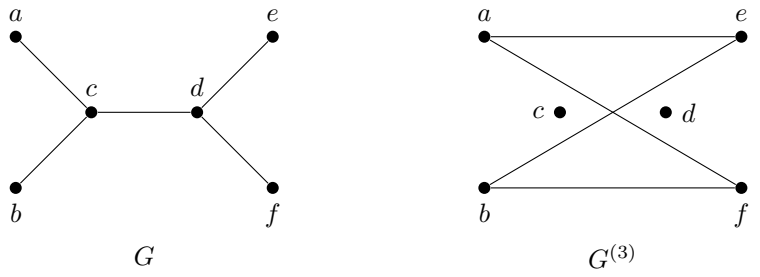

Figure 1: A graph G and the graph $G^{(3)}$ connecting vertices at distance exactly 3.

Map each connected component C of $G^{(3)}$ to a spanning tree $T(C)$; this is a one-to-one mapping.

Claim 6 *Let U be an alive connected component of G. Then $G^{(3)}$ contains a tree $T \subseteq U$ such that*
(1) $|T| \geq \dfrac{|U|}{\Delta^2+1}$;
(2) $\mathrm{dist}_G(u,v) \geq 3$ *for all distinct* $u, v \in T$.

Proof Build T greedily, starting from an arbitrary vertex of U. While some vertex of U has distance ≥ 3 from the current T, take a path inside U from T to such a vertex. Distances to T change by at most 1 along the path, so some vertex x on it has $\mathrm{dist}_G(x,T) = 3$ exactly; add x to T, attached by a $G^{(3)}$-edge to its nearest vertex of T. This keeps T a tree in $G^{(3)}$ with pairwise distances ≥ 3, giving (2). When the process stops, every vertex of U is within distance 2 of T; as at most Δ^2+1 vertices lie within distance 2 of a given vertex, $|U| \leq (\Delta^2+1)|T|$, giving (1). ■

4.3 The family of trees $\mathcal{T}_s$ and counting them

Let $s = \lceil \log 3n \rceil$ and let $\mathcal{T}_s$ be the family of possible trees of size s:

$$\mathcal{T}_s \;=\; \Big\{ T \subseteq V \;\Big|\; |T| = s,\ \mathrm{dist}_G(u,v) \geq 3 \text{ for all distinct } u, v \in T, \text{ and } T \text{ is connected in } G^{(3)} \Big\}.$$

If a tree T built as in the proof of Claim 6 has $|T| \geq s$, then its first s vertices form a member of $\mathcal{T}_s$.

Counting trees.

Fact 7 *The number of non-isomorphic trees on w nodes (ignoring the names of the nodes) is at most 4^w.*

Corollary 8 *In an N-node graph of maximum degree at most d, the number of subtrees with w nodes is at most $N \cdot 4^w \cdot d^w$.*

Proof [Why] Encode a subtree as follows.
- Choose a root node: $\leq N$ choices.
- Choose the shape of the tree: $\leq 4^w$ choices.
- Choose the "placement" by following a DFS sequence: $\leq d$ choices for the first child, $\leq d$ for the second, …, $\leq d$ for the w-th child, in total $\leq d^w$ choices.

■

Applying this to $G^{(3)}$ ($N = n$, $d = \Delta^3$, $w = s$) gives

$$|\mathcal{T}_s| \;\leq\; n\,(4\Delta^3)^s. \tag{1}$$

Putting it together. Fix $T \in \mathcal{T}_s$. By Observation 2, if all nodes of T survive then $B_v = 1$ for all $v \in T$. The nodes of T are at pairwise distance ≥ 3, so by Observation 4 the variables B_v, $v \in T$, are independent, and by Observation 3

$$\Pr[\text{all nodes in } T \text{ survive}] \;\leq\; \Big(\frac{1}{8\Delta^3}\Big)^s.$$

A union bound together with (1) gives

$$\begin{aligned}\Pr\big[\exists\, T \in \mathcal{T}_s \text{ such that all nodes in } T \text{ survive}\big] &\leq \sum_{T \in \mathcal{T}_s} \Pr[\text{all nodes in } T \text{ survive}] \\ &\leq |\mathcal{T}_s| \cdot \Big(\frac{1}{8\Delta^3}\Big)^s \\ &\leq n\,(4\Delta^3)^s \Big(\frac{1}{8\Delta^3}\Big)^s \;=\; \frac{n}{2^s} \;\leq\; \frac{1}{3}.\end{aligned}$$

Hence **with probability at least $2/3$, all surviving components have size at most $(\Delta^2+1)\,s$**: otherwise some alive component U would have $|U| > (\Delta^2+1)s$, so by Claim 6 it would contain a tree T with $|T| > s$ and pairwise distances ≥ 3, and the first s vertices of T would form a member of $\mathcal{T}_s$ all of whose nodes survive. This proves Claim 1. ■