6.5240 Sublinear Time Algorithms	September 25, 2024
Lecture 7	
Lecturer: Ronitt A. Rubinfeld	*Scribe: Kenny Zhang*

Today, we went over testing a graph for planarity in sublinear time.

1 Problem statement and definitions

Given a collection of graphs $\mathcal{P}$, a property testing algorithm for $\mathcal{P}$ must satisfy the following:

- If $G \in \mathcal{P}$, output PASS with probability at least 3/4.
- If G ε-far from $\mathcal{P}$, output FAIL with probability at least 3/4.

There are no restrictions on output for any graphs not in $\mathcal{P}$ that are ε-close to $\mathcal{P}$.

Our query model for today is on bounded-degree graphs, allowing queries for vertex degree of v and for the ith neighbor of the adjacency list of v.

Note. Distinction in property testing: one sided vs two-sided error. For today, we have two-sided error, where both PASS and FAIL outputs are allowed some error (like in the definition above). This is in contrast with one-sided error, i.e. when the algorithm is given an input that is expected to output PASS, the algorithm will always output PASS.

The definition for ε-close can vary, but for today's discussion on planarity testing, we use the following:

Definition 1 (ε-close (for downwards-closed properties))**.** A max-degree Δ graph G is "ε-*close* to $\mathcal{P}$" if it is possible to remove at most $\varepsilon\Delta n$ edges to turn G into some $G' \in \mathcal{P}$.

Note. In the pset, an "upwards-closed" defintion of ε-close is used for connectedness and diameter testing, where edges are added instead of removed.

For today, Δ is constant.

To do sublinear-time planarity testing, we will use the notion of "hyperfinite" graphs, and an important theorem linking planarity and hyperfiniteness:

Definition 2 ($(\widehat{\varepsilon}, k)$-hyperfinite)**.** Graph G is "$(\widehat{\varepsilon}, k)$-*hyperfinite*" if it is possible to remove at most $\widehat{\varepsilon}n$ edges such that all remaining connected components are of size at most k.

Remark. In some sense, hyperfinite graphs are not very well connected, and are the opposite of expander graphs.

Theorem 1. *For all ε, Δ there exists c such that every planar graph of max degree Δ is $(\varepsilon\Delta, c/\varepsilon^2)$-hyperfinite.*

Note. While planarity implies hyperfiniteness, hyperfiniteness does not imply planarity. Our algorithm must explicitly check for planarity.

Remark. By Kuratowski's theorem, a graph is planar if and only if it does not contain K_5 or $K_{3,3}$ as minors.

Remark. All graph properties that forbid classes of subgraphs have a related theorem!

Remark. Due to the planar separator theorem, it is possible to find $\sqrt{n}$ nodes that separate the graph into two subgraphs, each of up to $(2/3)n$ nodes.

Remark. There was a question asked during class that completely went over my head. It had something to do with smoothness/proximity oblivious/phase transition? Potentially something about wheter the algorithm's rejection probability is proportional to how far the graph is from being planar? Cool keywords for looking up interesting algorithmic ideas, but not important for this lecture.

2 Sublinear planarity testing algorithm overview

For any planar G, use hyperfiniteness with the parameters ($\widehat{\varepsilon} = (1/2)\varepsilon\Delta, k = c/\varepsilon^2$) to take a partition of G, where the partitions are the components of size at most $k = c/\varepsilon^2$.

Let G' be G without edges between partitions. Then, G' satisfies the following properties:

- Since we've only deleted edges, G being planar implies that G' is planar.
- G' differs from G by at most $(1/2)\varepsilon\Delta n$ edges (due to hyperfiniteness).
- Thus, if G is ε-far from planar, then G' is $(1/2)\varepsilon$-far from planar.
- All connected components in G' are size $k = O(1/\varepsilon^2)$, so we can check if a component is planar in $\mathsf{poly}(1/\varepsilon)$.

If such a partition does not exist, then G is not $((1/2)\varepsilon\Delta, c/\varepsilon^2)$-hyperfinite, and thus not planar.

How do we get this partition? Wait for later in the lecture... (specifically section 6)

3 Partition oracle definition

Input: a node v.

Output: a label $P[v]$, which is consistent between different invocations of the oracle.

Require: With $P(v)$ being the set of all nodes with label $P[v]$, the oracle outputs must satisfy that for all $v \in V$, $|P(v)| \leq k$ and $P(v)$ is connected.

Definition 3 (crossing edge)**.** Let (u, v) where $P[u] \neq P[v]$ be a *crossing edge*, and $C := \{(u, v) \in E \mid P[u] \neq P[v]\}$ be all the crossing edges of a partition.

Require: If G is planar, then with probability at least 9/10, the number of crossing edges $|C|$ is at most $(1/4)\varepsilon\Delta n$.

Note. The 9/10 is arbitrary, but it is chosen to be high enough so that a union bound over all potential sources of error in the algorithm still comes out to at least a 3/4 success probability.

4 Sublinear planarity testing using a partition oracle

Assume the partition oracle almost never fails. We'll union bound the error probability later.

The algorithm proceeds as follows:

I. Does the partition "look right?"

   That is, given a non-planar G, the oracle may find a partition with too many crossing edges.

   - Let $\widehat{f}$ be an estimate of the number of crossing edges to within an additive error of $(1/8)\varepsilon\Delta n$ (see 5.3).

     Note. How do we pick uniformly random edges for this estimate? Use the fact that G has bounded degree.
   - If $\widehat{f}$ is greater than $(3/8)\varepsilon\Delta n$, the number of crossing edges is larger than allowed, so output "not planar".

II. Test random partitions for planarity.

   - Choose $|\mathcal{S}| = O(1/\varepsilon)$ random nodes
   - For $s \in \mathcal{S}$, if $P(s)$ is not planar, output "not planar".

     Note. $|P[s]| \leq k = O(1/\varepsilon^2)$ so it's easy to test planarity.
   - Otherwise, if no counterexample is found, output "planar"

5 Analysis

5.1 Runtime analysis

The runtime will come out to be Δ/ε^3 calls to the partition oracle.

In part I, for sampling $\widehat{f}$, every sampled edge needs 2 calls to the oracle. Using sampling bounds (Chernoff/Hoeffding), there are $O(\Delta/\varepsilon^2)$ call to oracle (see 5.3).

In part II, for finding each component $P(s)$, since each component is size $O(1/\varepsilon^2)$, and each vertex has max degree Δ, there are $O(\Delta/\varepsilon^2)$ calls to the oracle.

Since $|\mathcal{S}| = 1/\varepsilon$ components are searched, this results in Δ/ε^3 calls to oracle.

5.2 Error analysis

Case I. If G is planar:

There exists a good partition that should be found by the oracle with the number of crossing edges $|C|$ at most $(1/4)\varepsilon\Delta n$, so

$$\mathrm{E}\left[\widehat{f}\right] \leq (1/4)\varepsilon\Delta n.$$

Since our estimate has an additive error of $(1/8)\varepsilon\Delta n$, with high probability $\widehat{f} \leq (3/8)\varepsilon\Delta n$, and the algorithm continues to part II.

For all $s \in V$, $P(s)$ is planar, so part II always passes, and the algorithm outputs "planar".

Case II. If G is ε-far from planar:

Case 1. The number of crossing edges $|C|$ is greater than $(1/2)\varepsilon\Delta n$.

By sampling bounds, with high probability

$$\widehat{f} > (1/2)\varepsilon\Delta n - (1/8)\varepsilon\Delta n = (3/8)\varepsilon\Delta n,$$

and the algorithm outputs "not planar".

Case 2. The number of crossing edges $|C|$ is at most $(1/2)\varepsilon\Delta n$.

Then, by definition of closeness, G' is $(1/2)\varepsilon$-close to G.

Since G is ε-far from planarity, G' is $(1/2)\varepsilon$-far from planarity.

This means that in order to make G' planar, at least $(1/2)\varepsilon\Delta$ edges must be deleted.

Since G has max degree Δ, these $(1/2)\varepsilon\Delta$ planarity-violating edges touch at least $(1/2)\varepsilon n$ nodes.

This means at least $(1/2)\varepsilon n$ nodes are in nonplanar components, so picking uniformly random nodes gives at least a $(1/2)\varepsilon$ probability of finding a nonplanar component.

Note. If we had unbounded degree, picking random nodes to get some property on edges does not work in general.

5.3 Constructing the estimator $\widehat{f}$

Let (u_i, v_i) for $i = 1, \ldots, c\Delta/\varepsilon^2$ be uniformly sampled edges.
Let $\widehat{f}_i = \mathbf{1}\left[P[u_i] \neq P[v_i]\right]$ and $\widehat{f} = m \cdot \frac{\varepsilon^2}{c\Delta} \sum \widehat{f}_i$.
Since $m \leq \Delta n$,

$$\begin{aligned}
&\Pr\left[\left|\widehat{f} - \mathrm{E}\left[\widehat{f}\right]\right| \geq (1/8)\varepsilon\Delta n\right] \\
\leq\ &\Pr\left[\left|\widehat{f} - \mathrm{E}\left[\widehat{f}\right]\right| \geq (1/8)\varepsilon m\right] \\
=\ &\Pr\left[\left|\sum \widehat{f}_i - \mathrm{E}\left[\sum \widehat{f}_i\right]\right| \geq (1/8) \cdot \frac{c\Delta}{\varepsilon}\right].
\end{aligned}$$

By Hoeffding's, we get

$$\Pr\left[\left|\sum \widehat{f}_i - \mathrm{E}\left[\sum \widehat{f}_i\right]\right| \geq (1/8) \cdot \frac{c\Delta}{\varepsilon}\right] \leq 2\exp\left(-\frac{2 \cdot \left((1/8) \cdot \frac{c\Delta}{\varepsilon}\right)^2}{c\Delta/\varepsilon^2}\right) = 2\exp\left(-\frac{c\Delta}{32}\right)$$

Thus, $\widehat{f}$ is an estimator of the number of crossing edges to within an additive error of $(1/8)\varepsilon\Delta n$.

6 Implementing the partition oracle

We will first look at a non-sublinear global partitioning algorithm. Then, we will turn this global algorithm into a sublinear local partitioning by showing that the dependency chains in the global algorithm are short. This is a similar technique to last lecture's approach to matching.

Definition 4 (Isolated neighborhood)**.** For $\delta < 1$, let S be a (δ, k)-*isolated neighborhood* of v if

(1) $v \in S$

(2) S is connected

(3) $|S| \leq k$

(4) The number of edges in the cut of S (one endpoint in S and the other in $V \setminus S$) is at most $\delta \cdot |S|$.

Observe that in planar graphs, in any good partition, most nodes are in (δ, k) isolated neighborhoods for $\delta = \varepsilon\Delta$, $k = c/\varepsilon^2$.

This is because hyperfiniteness tells us that the number of crossing edges is small, implying that the average number of cut edges over all partitions is $2\widehat{\varepsilon}n/(\text{number of partitions}) \leq 2\widehat{\varepsilon}/k$. (The factor of two comes from double counting).

Even if some partitions have a large number of cut edges, Markov's inequality tells us that for most partitions, the number of cut edges is small.

6.1 Global partitioning algorithm

Let $\Pi_1, \ldots \Pi_n$ be the nodes of G in random order.
$P := \emptyset$.
for $i = 1 \to n$ **do**
  if Π_i is still in the graph **then**
    Search for a (δ, k) isolated neighborhood in the remaining graph
    if Found an isolated neighborhood S' **then**
      $S := S'$.
    else
      $S := \Pi_i$
    $P := P \cup \{S\}$

Remove S and its adjacent edges in G.

Note. If G is planar, pulling out S keeps G planar.

Note. A singleton may not satisfy the isolated neighborhood definition, since the number of edges in the cut of a singleton may be as large as Δ, which is larger than the definition's upper bound $\delta \cdot |S| = \varepsilon\Delta$. We will need to show that number of times that we get singletons is low.

The story continues in the next lecture...