6.5240 Sublinear Time Algorithms	October 12, 2022

Lecture 9

Lecturer: Ronitt Rubinfeld *Scribe: Akash Das*

In this lecture, we continue the discussion of finding a sublinear time algorithm for testing if a graph is triangle-free.

Recall last lecture we presented an outline of what the algorithm should like. We also stated the Triangle Removal Lemma, which will be essential in our analysis of the algorithm. Additionally, last lecture we stated the Szemerédi Regularity Lemma and proved the Triangle Counting Lemma, which we will need this lecture to prove the Triangle Removal Lemma.

1 Triangle-free Testing

1.1 Setup and Goal

Since we want a tester to see if a graph is triangle-free we should first define what exactly this property is. The definition is given below.

Definition 1 (Triangle-free) *We say a graph $G = (V, E)$ is triangle-free if there doesn't exist distinct $u, v, w \in V$ such that $(u, v), (v, w), (u, w) \in E$.*

We also want to have some notion that a graph is far from being triangle free. In the next definition, we will do just that:

Definition 2 (ϵ-far) *We say that a graph G is ϵ-far from being triangle-free if one needs to delete at least ϵn^2 edges from the graph to make it triangle-free.*

Now, suppose we are given a graph $G = (V, E)$. We want to find a sublinear time algorithm that will either:

1. **Accept** if G is triangle-free
2. **Reject** with probability greater than $\frac{2}{3}$ if G is ϵ-far from triangle-free.

The outline for the algorithm we want to use is shown below:

Algorithm 1 Outline for Triangle-free tester

```
t ← some constant that will be specified later
i ← 0
N ← n
while i < t do
    Sample distinct vertices u, v, w from G and check if it's a triangle.
    If yes, then Reject
    i ← i + 1
end while
Accept
```

Note that if we let $t = O(n^3)$, where $n = |V|$, this algorithm would clearly work the way we want it to. Our goal to choose a value of t that doesn't depend on n and only depends on ϵ. To do this, we will need the Triangle Removal Lemma, which is stated and proved in section 1.4.

1.2 Prerequisite Definitions

Before delving into the statements of the lemmas we will need, we will first recall some important definitions from last lecture. This definitions will be essential for understanding the statements of the Triangle-Counting Lemma and the Szemerédi Regularity Lemma.

Definition 3 *Given a graph $G = (V, E)$ and two subsets $A, B \subseteq V$, we define*

$$e(A, B) = |\{(a, b) \in E | a \in A, b \in B\}|.$$

Note that $e(A, B)$ is just the number of edges between the two subsets.

Definition 4 (Edge density) *Given the same setup in the previous definition, we define the edge density between the two sets to be*

$$d(A, B) = \frac{e(A, B)}{|A||B|}$$

Definition 5 (ϵ-regular) *Given two subsets of vertices A and B, we say that (A, B) is ϵ-regular if $\forall A' \subseteq A, B' \subseteq B$ such that $|A'| \geq \epsilon|A|$ and $|B'| \geq \epsilon|B|$, we have that $|d(A, B) - d(A', B')| < \epsilon$.*

Definition 6 (Equipartition) *Given a set X, we say that an equipartition of X is a partition in which every part has the same size.*

For the purposes of this lecture, we will allow parts of an equipartition to have roughly equal size (if $|X|$ isn't divisible by the number of parts we are dividing the set into, then it isn't possible to get a partition in which every part is exactly the same).

Definition 7 (Refinement) *Given a partition $\mathcal{P}$ of the set S, we say that the partition $\mathcal{Q}$ of the set S is a refinement of $\mathcal{P}$ if every part of $\mathcal{Q}$ is a subset of a part of $\mathcal{P}$. More formally, if we have partitions of S defined below*

$$\mathcal{P} = \bigsqcup_{i=1}^{m} P_i, \mathcal{Q} = \bigsqcup_{i=j}^{n} Q_j,$$

we say $\mathcal{Q}$ is a refinement of $\mathcal{P}$ if and only if for every $1 \leq i \leq n$, there exists a $1 \leq j \leq m$ such that $Q_i \subseteq P_j$.

1.3 Two Necessary Lemmas

With these definitions in mind, we will state the Triangle Counting Lemma and the Szemerédi Regularity Lemma, both of which are very important to us for proving the Triangle Removal Lemma, which will lead to a sublinear time tester for the triangle-free property. We did state these lemmas in the previous lecture, but we will present slightly different formulations of the lemmas in this lecture that will be useful for us today.

Lemma 8 (Triangle-Counting Lemma) *For all $0 < \eta < 1$, there exists $\gamma = \frac{\eta}{2} = \gamma^{\triangle}(\eta)$ and $\delta = \frac{(1-\eta)\eta^3}{8} = \delta^{\triangle}(\eta)$ such that if A, B, C are disjoint subsets of V and each pair of subsets are γ-regular with density at least η, then G has at least $\delta|A||B||C|$ distinct triangles with a node in each of A, B, and C.*

We proved an equivalent version of this in last lecture, so will omit the proof here.

Lemma 9 (Szemerédi Regularity Lemma) *For all m, $\epsilon > 0$, there exists a constant $T = T(m, \epsilon)$ such that for any graph $G = (V, E)$ and any equipartition $\mathcal{A}$ of V into m sets, there exists an equipartition V into sets $V_1, V_2, \ldots, V_k$ which refines $\mathcal{A}$, such that $m \leq k \leq T$ and at most $\epsilon\binom{k}{2}$ pairs of these sets are not ϵ-regular.*

The proof of this lemma is long and beyond the scope of this class so we omit the proof.

Remark Note that $T = T(m, \epsilon)$ is a huge number that is a function of m and ϵ, but the important thing to note is that is a constant that is independent of the graph we are looking at. The actual constant, according to [1], is an $O(\epsilon^{-5})$-level exponential of m.

Now, we are ready to state and prove the Triangle Removal Lemma.

1.4 Triangle Removal Lemma

Lemma 10 *For all $\epsilon > 0$, there exists a $\delta > 0$ such that any graph on n nodes which is ϵ-far from being triangle-free has at least $\delta\binom{n}{3}$ triangles.*

Proof Suppose we are given ϵ and G. First, we define $\epsilon' = \min(\frac{\epsilon}{5}, \gamma^{\triangle}\left(\frac{\epsilon}{5}\right)) = \frac{\epsilon}{10}$. Now, by Szemerédi Regularity Lemma, we know that we can get an equipartition $\{V_1, V_2, \ldots, V_k\}$ of V such that

$$\frac{5}{\epsilon} \leq k \leq T\left(\frac{5}{\epsilon}, \epsilon'\right)$$

and such that at most $\epsilon'\binom{k}{2}$ pairs of subsets are not ϵ'-regular. Now, we construct G' by taking G and performing the following deletions:

1. Delete edges internal to any V_i. Since each vertex is in the same set with $\frac{n}{k}$ vertices, this deletes at most $n \cdot \frac{n}{k} = \frac{n^2}{k} \leq \frac{\epsilon n^2}{5}$ edges.
2. Delete edges between pairs of sets that are not ϵ'-regular. Since there are at most $\epsilon'\binom{k}{2}$ pairs of sets that are not ϵ'-regular, this deletes at most $\epsilon'\binom{k}{2} \cdot (\frac{n}{k})^2 \leq \frac{\epsilon n^2}{20}$ edges.
3. Delete edges between low-density pairs of sets (density smaller than $\frac{\epsilon}{5}$). Since the edge density between such pairs is at most $\frac{\epsilon}{5}$, the number of edges we delete is at most $\frac{\epsilon}{5}\binom{n}{2} \leq \frac{\epsilon n^2}{10}$ edges.

Adding these up, we get the number of total edges we deleted is at most

$$\frac{\epsilon n^2}{5} + \frac{\epsilon n^2}{20} + \frac{\epsilon n^2}{10} = \frac{7\epsilon n^2}{20} < \epsilon n^2.$$

Since we assumed that G is ϵ-far from being triangle-free, we have that G' still has at least one triangle u, v, w. However, since G' contains no internal edges, we know that u, v, w must lie in distinct partitions. Suppose $u \in V_h$, $v \in V_i$, and $w \in V_j$ for distinct h, i, j. Since all pairs of sets with edges between them are ϵ'-regular and have density at least $\frac{\epsilon}{5}$, we know that V_h, V_i, V_j are all pairwise ϵ'-regular and have pairwise edge densities at least $\frac{\epsilon}{5}$. Thus, by Triangle Counting Lemma, we get that the total number of distinct triangles in G is at least $\epsilon'|V_h||V_i||V_j|$. We can lower-bound this number as shown below:

$$\epsilon'|V_h||V_i||V_j| = \frac{\epsilon}{10} \cdot \frac{n^3}{k^3} \geq \frac{\epsilon n^3}{10[T(\frac{5}{\epsilon}, \frac{\epsilon}{10})]^3} \geq \delta'\binom{n}{3}$$

for some constant $\delta' > 0$. Note that δ' is only dependent on ϵ, and thus is constant. This concludes the proof of the lemma. ■

Note that the value δ' might be very small, but the fact that it is a positive constant that only depends on ϵ will be sufficient for our purposes.

1.5 Statement and analysis of algorithm

Suppose we are given a graph $G = (V, E)$ and a parameter $\epsilon > 0$. By the Triangle Removal Lemma, we know that there exists $\delta > 0$ such that any graph G that is ϵ-far from being triangle-free has at least $\delta\binom{n}{3}$, where $n = |V|$. We use this parameter δ in the algorithm, shown below.

Algorithm 2 Triangle-free tester

$t \leftarrow \frac{2}{\delta}$
$i \leftarrow 0$
$N \leftarrow n$
while $i < t$ **do**
  Sample distinct vertices u, v, w from G and check if it's a triangle.
  If yes, then Reject
  $i \leftarrow i + 1$
end while
Accept

Theorem 11 *Algorithm 2 is sublinear and always accepts G if G is triangle free and it rejects G with probability greater than $\frac{2}{3}$ if G is ϵ-far from triangle-free.*

Proof First note that our algorithm clearly accepts whenever G is triangle-free, as we simply never will see a triangle. Now, suppose G is ϵ-far from triangle-free. Then, by Triangle Removal Lemma, we know that there exists at least $\delta'\binom{n}{3}$ distinct triangles, where $n = |V|$. Thus, the probability of three randomly sampled points being a triangle is at least δ. Thus, the probability that we never reject after t iterations of the while loop is at most $(1-\delta)^t = (1-\delta)^{\frac{2}{\delta}} \le e^{-\delta \cdot \frac{2}{\delta}} = e^{-2} < \frac{1}{3}$. Thus, the probability that we reject if G is ϵ-far from triangle-free is greater than $\frac{2}{3}$. Thus, our algorithm is correct.

As for time complexity, note that we since $t = O(\frac{1}{\delta})$, and since δ is only dependent on ϵ, we have that this algorithm has no dependence on $|V|$ or $|E|$, and thus is a valid sublinear algorithm. Thus, we are done. ■

References

[1] Szemerédi Regularity Lemma.