0368.416701 Sublinear Time Algorithms	December 7, 2015
Lecture 8	
Lecturer: Ronitt Rubinfeld	*Scribe: Moab Arar, Nir Aviv, Israela Solomon*

1 Lecture Overview

1. Testing properties of dense graphs - Bipartiteness.

2 Property testing in dense graphs

When we discussed property testing in sparse graphs, the graph representation was an adjacency list. For dense graphs, the property testing algorithms will use an adjacency matrix representation.

Definition 1 (Adjacency matrix) *Given a graph G=(V,E), the adjacency matrix of G is the matrix A such that*

$$A_{i,j} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

A property testing algorithm must pass with high probability for a graph that has the property, and must fail with high probability for a graph that is ϵ-far from having the property.

Definition 2 (ϵ-far from having a property) *A graph G is ϵ-far from having property P if it takes more than ϵn^2 changes to entries in A_G in order to make G a member of P.*

Properties are closed under nodes relabeling, meaning that relabeling nodes doesn't affect whether or not a graph has a certain property.

We can see that any graph is ϵ-close to being connected, since it takes a maximum of $n-1$ changes to A_G in order to connect a graph, and we know that $n - 1 < \epsilon n^2$ regardless of ϵ.

3 Testing for bipartiteness

We will discuss testing of bipartiteness in dense graphs.

3.1 Property definition

Definition 3 (Bipartiteness) *A graph G=(V,E) is bipartite if (Both these definitions are equivalent)*

1. *It's nodes can be colored red or blue so that no edge is monochromatic (has both it's nodes colored the same color).*
2. *V can be partitioned into (V_1, V_2) s.t. $\nexists e = (u, v) \in E$ such that both $u, v \in V_1$ or both $u, v \in V_2$.*

Definition 4 (Violating edge) *For a graph G=(V,E) and some partitioning (V_1, V_2) of V, a violating edge is an edge e=(u,v) such that both $u, v \in V_1$ or both $u, v \in V_2$.*

If a given graph G is ϵ-far from bipartite:

1. It takes more than ϵn^2 changes to make it bipartite;
2. For every partitioning (V_1, V_2) of V, there are more than ϵn^2 violating edges.

Remark

1. For $\epsilon = 1/n$, testing for bipartiteness is known to be in NP-HARD.
2. For sparse graphs, testing bipartiteness to ϵdn is known to have a lower bound on query complexity of $\Omega(\sqrt{n})$.
3. For dense graphs, we will show a bipartite tester with query complexity that depends only on ϵ.

3.2 The good algorithm

First we will present a correct algorithm, which we will not prove:

Algorithm 1: The good algorithm

```
Pick a sample of nodes of size Θ(1/ε² log(1/ε))
Consider the induced graph of only the samples
if the induced graph is bipartite then
    Pass
else
    Fail
```

This algorithm has query complexity of $\Theta(\frac{1}{\epsilon^2}log(\frac{1}{\epsilon}))$ and its runtime is $poly(\frac{1}{\epsilon})$, since testing bipartiteness can be done with a simple BFS.

The algorithm is correct, but proving it is quite complicated and is done by proving other algorithm correctness. We will look at some other algorithms instead of proving this one.

3.3 Another attempt - Algorithm 0

In order to check if a graph G is ϵ-far from bipartite we can use the following algorithm:

Algorithm 2: Tester for graphs that are ϵ-far from bipartite

```
Pick some partitioning V_1, V_2 of V.
Take a sample of edges of size m = Θ(1/ε log(1/δ)). Consider the induced subgraph on the sample.
if there is a violating edge according to the partitioning then
    Fail
else
    Pass
```

Since a non-bipartite graph G has more than ϵn^2 violating edges, we will hit one with probability $\geq 1 - (1-\epsilon)^m \geq 1 - \delta$. This algorithm works only for graphs that are ϵ-far from bipartite. Graphs which are bipartite might have only one partitioning for which they don't have any violating edges. We will change the above algorithm so that it would check every partitioning of V:

Algorithm 3: Another attempt at a bipartite tester

```
Take a sample of edges of size m = Θ(1/ε log(1/δ))
for every partition V_1, V_2 of V do
    count the number of violating edges in the sample according to the partition (Let's call this number
    Violating_{V_1,V_2})
if for all partitioning, Violating_{V_1,V_2} > 0 then
    Fail
else
    Pass
```

This algorithm will pass every bipartite graph, since there is some partitioning for which there are no violating edges. If the algorithm gets a graph G that is ϵ-far from bipartite:

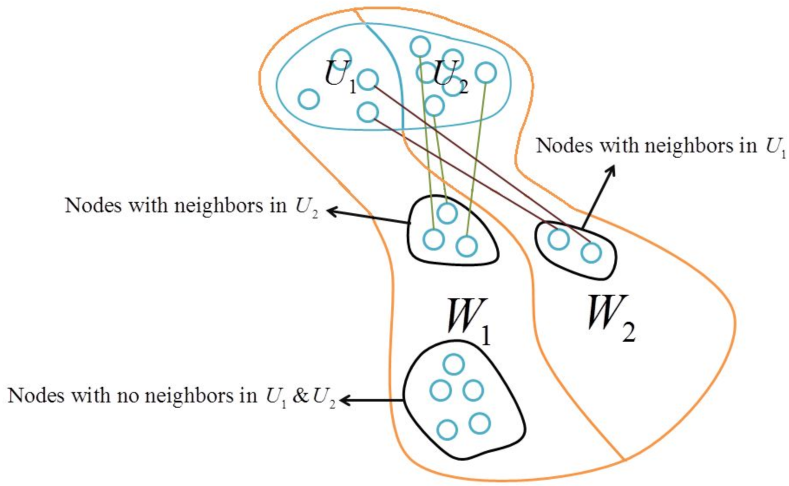

Figure 1: The induced partition (W_1^{U1U2}, W_2^{U1U2})

- For each partitioning V_1, V_2 of V, $Pr[Violating_{V_1,V_2} > 0] \geq (1-\delta)$ (We saw this in the previous algorithm);
- The probability the algorithm output fails is $Pr[\forall V_1, V_2(Violating_{V_1,V_2} > 0)] \geq (1 - 2^n\delta)$ (using union bound).

We need this probability to be high so we must pick $\delta < \frac{1}{2^n}$. This algorithm's sample complexity is $m = \Theta(\frac{1}{\epsilon}n)$ and it's runtime is $O(2^n)$.

3.4 An Improved Algorithm

We had two main problems with the previous algorithm. The first one is the running time. We had to go over every one of the 2^n partitions of V which is not feasible. The second one was the query complexity. Although $\Theta(n/\epsilon)$ is sub-linear in the input size, one can do better.

Definition 5 (partition oracle) *Let $U \subseteq V(G)$, and let (U_1, U_2) be a partition of U. We define the partition, denoted (W_1^{U1U2}, W_2^{U1U2}), on V that is induced by (U_1, U_2), in the following way:*

$$W_2^{U1U2} = U_2 \cup \{ v \in V \setminus U \mid \exists x \in U_1 \ s.t. \ (x, v) \in E \ and \ \nexists x \in U_2 \ s.t. \ (x, v) \in E\}$$

$$W_1^{U1U2} = V \setminus W_2^{U1U2}$$

Namely, W_2^{U1U2} contains the vertices in $V \setminus U$ that are neighbours of vertices in U_1 but not of vertices in U_2, and W_1^{U1U2} contains vertices in $V \setminus U$ that are neighbours of vertices in U_2, or in both U_1, U_2, or doesn't have neighbours in U. See Figure 1.

Analysis

If G Is Far From Bipartite

Suppose G is ϵ-far from bipartite. Since U is of size $\Theta(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right))$, there are $\frac{1}{\delta} = 2^{\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)}$ partitions of U. Fix some partitioning U_1,U_2, and look at the partitioning W they induce on V. Since G is ϵ-far

Algorithm 4: Algorithm1

Pick U of size $\Theta(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right))$ and U' of size $\Theta(\frac{1}{\epsilon^2}\log\left(\frac{1}{\epsilon}\right))$ nodes.
Denote $U' = \{u_1, v_1, u_2, v_2, ...\}$. Define a pair off of those vertices by $P = \{(u_1, v_1), (u_2, v_2), ...\}$.
Query the adjacency matrix A on every pair $(u, v) \in P$.
foreach (U_1, U_2) *partition of* U **do**
    Let $W = (W_1^{U1U2}, W_2^{U1U2})$ be the induced partition.
    Count $m(U_1, U_2)$ the number of edges in P that violate W.
    if $\frac{m(U_1,U_2)}{|P|} \leq \frac{3}{4}\epsilon$ **then**
        Pass
Fail

from bipartite, there are at least ϵn^2 edges which violating W. Let X be a random variable that is the number of edges in P which violate W. We can think of U' as being chosen after W was fixed. Therefore by linearity of expectation over the edges in P

$$E(X) \geq |P| \frac{\epsilon n^2}{n^2} = |P| \epsilon,$$

and since elements in P are independent, using Chernoff bounds we conclude that

$$Pr(\frac{X}{|P|} < \frac{3}{4}\epsilon) \leq \frac{1}{8}\delta.$$

So by the union bound the probability that one of the partitions of U causes the tester to pass can be upper bounded:

$$Pr(\exists (U_1, U_2) \text{ such that } \frac{X(U_1, U_2)}{|P|} < \frac{3}{4}\epsilon) \leq \sum_{(U_1,U_2)} Pr(\frac{X(U_1, U_2)}{|P|} < \frac{3}{4}\epsilon) \leq \frac{1}{8}\delta \cdot \frac{1}{\delta} = \frac{1}{8}.$$

So the probability we will fail given an ϵ-far from bipartite graph is at least $\frac{7}{8}$.

If G Is Bipartite

In this case we know there exists some partition $V = (Y_1, Y_2)$ of the vertex set such that there are no violating edges. For a fixed U, let's look at its partition $U_1 = Y_1 \cap U$, $U_2 = Y_2 \cap U$. Intuitively, we want the induced partition (W_1^{U1U2}, W_2^{U1U2}) to be (Y_1, Y_2), but it does not have to happen, because we arbitrarily add to W_1^{U1U2} the vertices that does not have any neighbours in U.

However, we will show this induced partition $W = (W_1^{U1U2}, W_2^{U1U2})$ is not that far from (Y_1, Y_2).

Notice that only vertices which do not have a neighbour in U can contribute violating edges. We divide those vertices without a neighbour in U into two groups

$$A = \left\{ v \mid \nexists x \in U \text{ s.t. } (x, v) \in E \text{ and } deg(v) < \frac{\epsilon}{4} n \right\},$$

$$B = \left\{ v \mid \nexists x \in U \text{ s.t. } (x, v) \in E \text{ and } deg(v) \geq \frac{\epsilon}{4} n \right\}.$$

Every edge that violates W is either violating for (Y_1, Y_2) or touches a vertex in A or in B. So

$$|\{W \text{ violating edges}\}| \leq |\{(Y_1, Y_2) \text{ violating edges}\}| + |\{\text{edges touching } A\}| + |\{\text{edges touching } B\}|.$$

The first summand is 0 (by the definition of (Y_1, Y_2). The second and the third summands are bounded by $\frac{\epsilon}{4} n |A|$, $n |B|$ respectively. So we conclude that

$$|\{W \text{ violating edges}\}| \leq \frac{\epsilon}{4} n |A| + n |B|.$$

We will show that with high probability over the choice of U it holds that $|B| \le \frac{1}{4}\epsilon n$ which will result that with high probability

$$|\{W \text{ violating edges}\}| \le \frac{\epsilon}{4} n\,|A| + n\,|B| \le \frac{\epsilon}{4} n^2 + n\frac{1}{4}\epsilon n = \frac{\epsilon}{2} n^2.$$

For the next lemma, let us call a vertex with degree at least $\frac{\epsilon}{4}n$ a high degree vertex.

Lemma 6 *The probability (over the choice of U) that there are at most $\frac{\epsilon}{4}n$ high degree nodes in V with no neighbour in U, is at least $\frac{7}{8}$. That is*

$$Pr(|B| \le \frac{\epsilon}{4}n) \ge \frac{7}{8}.$$

Proof Define indicator random variables σ_v for each $v \in V$: $\sigma_v = 1$ if and only if $deg(v) \ge \frac{\epsilon}{4}n$ and v has no neighbours in U.

Let v be such vertex. Let p be the probability the some specific node in U is not a neighbour of v. Because v is of high degree, $p \le 1 - \frac{\epsilon}{4}$.

The vertices of U are chosen independently so

$$E(\sigma_v) = Pr(\sigma_v = 1) = p^{|U|} \le (1 - \frac{\epsilon}{4})^{|U|} \le (1 - \frac{\epsilon}{4})^{\frac{4}{\epsilon}\log\frac{\epsilon}{32}} \le \frac{\epsilon}{32}.$$

Note that $|B| = \sum_{v \in V} \sigma_v$, so by linearity of expectation and the previous computation

$$E(|B|) \le \frac{\epsilon}{32} n.$$

And the end is due to Markov inequality

$$Pr(|B| \ge \frac{\epsilon}{4}n) \le \frac{E(|B|)}{(n\epsilon/4)} \le \frac{1}{8}.$$

■

Remark One way to prove the correctness of the "dream" algorithm that was presented at the beginning of this lecture, is to argue that it follows from the correctness of the algorithm we have just analyzed.

4 Ideas For Projects

A few ideas were suggested for final projects:

1. For dense graphs we have used adjacency matrix to represent the edges, whilst for sparse graphs we have used adjacency list. Perhaps we can obtain better results if we use both.
2. We have focused on undirected graphs. It can be interesting to study similar problems for directed graphs, weighted graphs, and other variants.