CSCI 1951-W Sublinear Algorithms for Big Data **Fall 2020**

Lecture 7: Dense Graph

Lecturer: Jasper Lee *Scribe: Alessio Mazzetto*

1 Dense Graph - Bipartiteness

In this lecture, we study the problem of testing whether a given dense graph $G = (V, E)$ is bipartite, in the dense graph model. As a reminder, a bipartition of V is a pair (V_1, V_2) such that $V_1, V_2 \subseteq V$, $V_1 \cap V_2 = \emptyset$, and $V_1 \cup V_2 = V$. A graph G is bipartite if there exists a bipartition of (V_1, V_2) of V such that $E \cap (V_1 \times V_2) = \emptyset$. The following definition will prove useful.

Definition 7.1 Given a bipartition (T_1, T_2) of T, edge (u, v) is said to *violate* (T_1, T_2) if both u and v are neighbor in T_1 or T_2, i.e. $(u, v) \in T_1 \times T_1$ or $(u, v) \in T_2 \times T_2$

Proposition 7.2 *Suppose $G = (V, E)$ is ϵ-far from bipartite. Then, for any bipartition of V, there are at least $O(\epsilon n^2)$ edges that violate that bipartition.*

Proof. The proposition immediately follows from the definition of ϵ-farness. In fact, if there exists a bipartition (V_1, V_2) with only $\epsilon n^2/4$ edges that violate that bipartition, we could remove them and obtain a bipartite graph. This is not possible, as we assume that G is ϵ-far from bipartite. □

The previous proposition suggests a simple ϵ-tester algorithm to check whether a graph is bipartite. The algorithm has query complexity $O(n/\epsilon)$, which is sublinear in the size of the input, as a dense graph has size $O(n^2)$.

Algorithm 7.3 *The algorithm operates as follows:*

- *Query $O(n/\epsilon)$ random pairs (i, j)*
- *Accept if and only if the queried subgraph is bipartite*

Proposition 7.4 *Algorithm 7.3 is a one-sided ϵ-tester for bipartiteness of dense graphs with query complexity $O(n/\epsilon)$ and runtime $O(n/\epsilon)$.*

Proof. Completeness: The algorithm trivially accepts all the graphs which are bipartite, as every subgraph is still bipartite.

Soundness: Let m be the number of sampled edges in the first step of the algorithm. Assume that G is ϵ-far from bipartite. We reject if and only if the queried subgraph is not bipartite, which implies that for any bipartition (V_1, V_2), we sampled an edge e that violates such bipartition, i.e. $e \in V_1 \times V_1$ or $e \in V_2 \times V_2$. Fix a bipartition (V_1, V_2). By Proposition 7.1, we know that there are at least $O(\epsilon n^2)$ edges that violate the bipartition. Hence, the probability that we do not sample any of those edges is $\leq \left(1 - \frac{O(\epsilon n^2)}{O(n^2)}\right)^m$. By an union bound over all possible 2^n bipartitions, we have that:

$$\begin{aligned}
&\mathbb{P}(\text{Algorithm wrongly accepts} \,|\, G \text{ is } \epsilon - \text{far from bipartite}) \\
=&\mathbb{P}(\forall \text{ bipartition } (V_1, V_2), \text{ no sampled edges violate that bipartition} \,|\, G \text{ is } \epsilon - \text{far from bipartite}) \\
\leq&(2^n)(1 - O(\epsilon))^m \leq 1/3 \Longleftarrow m = O(n/\epsilon)
\end{aligned}$$

where the first inequality is due to an union bound over all possible bipartitions, and in the second inequality we used the standard inequality that $(1-\epsilon) \leq e^{-\epsilon}$. Hence, for a suitable $m = O(n/\epsilon)$, the algorithm correcly rejects any graph G which is ϵ-far from bipartite with probability at least 2/3.

The query complexity is $O(n/\epsilon)$ as we sampled that many edges. The time complexity is $O(n/\epsilon)$, which is the time required to check whether the queried subgraph is bipartite. □

It is possible to significantly improve upon the previous algorithm. In particular, it is possible to obtain a one-sided ϵ-tester for bipartiteness of dense graphs with query complexity $\tilde{O}(1/\epsilon^4)$. While the algorithm is very simple and easy to implement, its analysis requires some effort.

Algorithm 7.5 *The algorithm operates as follows:*

- *Sample* $r = O(\frac{1}{\epsilon^2} \log \frac{1}{\epsilon})$ *random vertices.*
- *Query all* $\binom{r}{2}$ *possible edges for the subgraph induced by the sample vertices.*
- *Accept if and only if the resulting subgraph is bipartite*

Theorem 7.6 *Algorithm 7.5 is a one-sided ϵ-tester for bipartiteness of dense graphs with query complexity and time complexity equal to $\tilde{O}(1/\epsilon^4)$.*

The main idea of the proof of the previous theorem is the following (we focus on the soundness, as completeness is trivial). For the sake of the analysis, we can view the first step of the algorithm as the sampling of two disjoint sets of vertices:

- T of size $t = O(\frac{1}{\epsilon} \log \frac{1}{\epsilon})$
- S of size $s = O(t/\epsilon)$

In the analysis, we will try to use the set T to enforce a bipartition on V, while the set S is used to check whether there are edges violating such bipartition.

A first issue is that, unlike the biclique testing problem we saw last class, it is very unclear how to obtain a canonical bipartition from the set T. Moreover, in the analysis, we would like to get not only a bipartition of T but a bipartition of all the vertices V. As a solution, we will consider all possible 2^t bipartitions (T_1, T_2) of T (and then perform an union bound over that), and for each of them we will associate a bipartition on V based on the neighbors of T_1 and T_2.

The second issue with this approach is that we do not know how to handle vertices that are not neighbor to T, when we generalise from a bipartition of T to one of V. To mitigate the problem, we will show that, with high probability, the set T is neighbor to most of the "high degree" vertices (in a sense formalised later). Since "most of the high degree vertices" cover most of the edges on the graph (low degree vertices involve few edges by definition, a small number of high degree vertices also involve few edges total), this implies that any bipartition of V is "not far" from one that is generated from a bipartition of T based on the neighbor information. Therefore it suffices to exploit Proposition 7.2 on these 2^t bipartitions generated from T.

As a digression, the 2^t bipartitions generated from T essentially form an ϵ-net over the entire set of bipartitions on V, but we will not go into any further detail on this perspective.

The following definitions formally describes what is a high-degree vertex and when a set T is neighbor to most of those vertices.

Definition 7.7 A vertex is said to be high degree if and only if its degree is $\geq \epsilon n/100$. T is said to be a *good* set if and only if it neighbours all but $\leq \epsilon n/100$ of the high degree vertices.

The sketch of the soundness analysis is the following. We fix a graph G that is ϵ-far from bipartiteness, then we show:

1. the set T is good with very high probability (99%);
2. for any good T and for any bipartition of T, there are at least $\epsilon n^2/100$ edges violating that bipartition (this implies that the set S is likely to pick up on them);
3. conditioned on T being good, every bipartition of T is accepted with small probability (we then union bound over all possible bipartitions 2^t);
4. the analysis done with the two disjoint sets S and T still hold for Algorithm 7.5. In particular, we will show that the algorithm does a stronger check than the one done in the analysis.

The following proposition captures the intuition that T is likely to be good, that is, T is neighbor to most of the high degree vertices with high probability. This is the step 1 of the sketch of the analysis.

Proposition 7.8 *T is good with probability $\geq 99\%$.*

Proof. Fix a particular high degree vertex $v \in V$. Then, we have that:

$$\mathbb{P}(v \text{ does not neighbour } T) \leq \left(1 - \frac{n\epsilon/100}{n}\right)^t = (1 - \epsilon/100)^t \leq \epsilon/10000$$

The last inequality is due to the fact that $t = O\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$. Let X be the number of vertices that are not neighbours of T. By linearity of expectation, we have that $\mathbb{E}\,X = n \cdot \epsilon/10000$. By using Markov's inequality, we have that:

$$\mathbb{P}\left(X > \frac{\epsilon n}{100}\right) \leq \frac{\mathbb{E}\,X}{\epsilon n/100} = \frac{\epsilon n/10000}{\epsilon n/100} = 1\%$$

□

As stated previously, we will use the set S to check whether there are any edges that violate the bipartition induced by T. This is possible if for each bipartition induced by T, there are a significant amount of edges that violate such a bipartition. We will show that if G is ϵ-far from bipartite and T is good, then this is indeed the case. This will allow us to claim that each bipartition of T is accepted with small probability (and then union bound all over 2^t bipartitions). This is the step 2 of the sketch of the analysis.

Proposition 7.9 *Let G be ϵ-far from bipartite. Then, for any good T, and any bipartition (T_1, T_2), there are at least $\geq \varepsilon n^2/100$ edges that violate such bipartition.*

Proof. Based on T, we construct a bipartition (V_1, V_2) of V as follows: we set $V_1 = \texttt{neighbor}(T_2)$ and $V_2 = V \setminus V_1$ (note that we could obtain an incomplete bipartition if we set $V_2 = \texttt{neighbor}(T_1)$). As G is ϵ-far, by Proposition 7.2 there are at least $\geq \epsilon n^2/2$ edges violating (V_1, V_2). We now want to upper bound the number of edges that violate

(V_1, V_2) but not (T_1, T_2). Those edges are always going to be edges that do not neighbor T. We count those edges by dividing them in two categories: edges that have high degree endpoints, and edges that have low degree endpoints. In the first case, we have that the number of edges that have high degree endpoints and not neighbor T is at most $\leq n \cdot \epsilon n/100 = \epsilon n^2/100$, as n is the maximum degree of a high degree vertex and $\epsilon n/100$ and is an upper bound on the number of high degree vertices not neighboring T given that T is good. In the second case, the total number of edges that have low degree endpoints and not neighbor T is at most $n \cdot \epsilon n/100 = \epsilon n^2/100$, as n is an upper bound to the total number of vertices that can be low degree, and $\epsilon n/100$ is an upper bound to the degree of a low degree vertex. Hence, we have that the number of edges that violate (T_1, T_2) is $\geq \epsilon n^2/2 - 2 \cdot \epsilon n^2/100 > \epsilon n^2/100$. □

We are now ready to prove Theorem 7.6. The following proof contain steps 3 and 4 of the sketch.

Proof of Theorem 7.6. The completeness property is trivial. We will now prove the soundness of the algorithm. Assume that G is ϵ-far from bipartite. The algorithm fails in two cases: (1) T is not good, (2) T is good but there exists a bipartition of T without violating edges in T or S. Note that this is a pessimistic analysis, as we are saying that the algorithm fails every time T is not good. By Proposition 7.8, we know that event (1) happens with very small probability, therefore we focus on event (2). We fix a bipartition (T_1, T_2) for a good T. By Proposition 7.9, we know there are at least $\epsilon n^2/100$ edges that violate this bipartition. This implies that there are at least $\epsilon n^2/100 - O(t^2) - O(tn) = \epsilon n^2/100 - O(\frac{1}{\epsilon^2}\log^2\frac{1}{\epsilon}) - O(tn) \geq \epsilon n^2/200$ edges e (last inequality is true for n big enough), such that e violates the bipartition (T_1, T_2), but e has neither endpoint in T. In particular, those are the edges that the set S is checking on (by checking all edges within $S \times S$), as S and T are disjoint.

If we sample s' edges outside of T, the probability not finding an edge violating the partition (T_1, T_2) is upper bounded by $(1 - O(\epsilon n^2)/O(n^2))^{s'} = O(1 - O(\epsilon))^{s'}$. By union bound over all the partition of T, we have that:

$$\begin{aligned}&\mathbb{P}(\text{no edge violate any partition of } T \text{ by sampling } s' \text{ edges outside of } T)\\ \leq& O(2^t) \cdot O(1 - O(\epsilon))^{s'} \leq 1/4 \Longleftarrow s' = O(t/\epsilon)\end{aligned}$$

The first inequality is due to a union bound, whereas the implication is due to the same computations done for the analysis of Algorithm 7.5. Therefore, if $S = O(s')$, with probability at least $3/4$ we will find a violating edge. By a union bound with the case of when T is not good (1% probability), we have that with probability at least $2/3$, the algorithm correctly rejects an ϵ-far graph G.

Note that our analysis is done with two disjoint sets T of size t and S of size $O(s') = O(t/\epsilon)$. We observe that the actual algorithm pick $O(s')$ vertices and query all pairs among those vertices. Clearly this is a stronger testing than the one done in the proof, hence the analysis still holds. □

This algorithm does not achieve the best query complexity for this problem, in fact there exists a one-sided ϵ-tester for bipartiteness of dense graphs with query complexity equal to $\tilde{O}(1/\epsilon^3)$, which follows from the proof above.

As a side note, the concept of bipartite is strictly related with the concept of 2-colorability of a graph. In fact, it's straightforward to see that a graph is 2-colorable if

and only if it is bipartite. The problem is a special case of the more general k-colorability problem.

Fact 7.10 *The decision problem of determining whether a graph is k-colorable for $k \geq 3$ is NP-hard.*

It is interesting to point out that based on the idea presented in the previous algorithm, it is possible to build an ϵ-tester to determine whether a graph is k-colorable (we won't go into the details of the analysis)

Algorithm 7.11 *The algorithm operates as follows:*

1. *Sample $O(\frac{k^2}{\epsilon^3} \log k)$ vertices uniformly at random, and query all the edges within these vertices.*

2. *Check if induced subgraph is k-colorable.*

As stated earlier, checking whether the induced subgraph is k-colorable is an NP-hard problem, therefore Step 2 of this algorithm requires exponential time on the size of the queried subgraph. However, since the queried subgraph has a small constant size (size independent of n), the tester's runtime is also independent of n.