

Testing Bipartiteness of Dense Graphs

Definitions

Recall that a graph $G = (V, E)$ is *bipartite* if there exists a partition (V_1, V_2) of the vertices where there are no edges (u, w) such that $u, w \in V_1$ or $u, w \in V_2$. We shall say in such a case that the partition is *bipartite*. If a partition (V_1, V_2) is not bipartite, then we shall say that the edges $(u, w) \in E$ such that $u, w \in V_1$ or $u, w \in V_2$ are *violating edges* (with respect to (V_1, V_2)). Recall that we have an algorithm for deciding whether a graph is bipartite that runs in time linear in the number of edges (by performing Breadth First Search).

For a distance parameter $\epsilon \in [0, 1]$, we say that a graph G is ϵ -far from bipartite if it is necessary to remove more than $\epsilon|E|$ edges from the graph so that the graph becomes bipartite. Observe that if a graph is ϵ -far from bipartite, then *every* partition (V_1, V_2) has more than $\epsilon|E|$ violating edges. In what follows we shall consider dense graphs, that is, $|E| = \Theta(n^2)$, so for simplicity we use the following definition:

Definition 1 *A dense graph G is ϵ -far from (being) bipartite if it is necessary to remove more than ϵn^2 edges to make it bipartite.*

The algorithms we consider are given query access to the graph, where queries are of the form: “is there an edge between vertex u and vertex w in G ?”. We refer to such queries as *vertex-pair queries*. (This is as opposed to another type of natural queries called neighbor queries in which the algorithm asks for the i 'th neighbor of a vertex v .)

Given the above we define our testing problem.

Definition 2 *An algorithm for testing bipartiteness of dense graphs is given a distance parameter ϵ and is allowed to perform vertex-pair queries of its choice. We require:*

- *If the graph is bipartite then the algorithm should accept;*
- *If the graph is ϵ -far from bipartite then the algorithm should reject with probability at least $2/3$ and provide evidence to the non-bipartiteness of the graph.*

The above definition is a bit more strict than the one we defined in class, but our algorithm will fit this definition. In particular, the “evidence” that the algorithm provides is in the form of a small subgraph of G that is not bipartite.

One way to think about the problem is that the algorithm is actually trying to “catch” graphs that are very non-bipartite; when it catches such a graph then it has “evidence” that the graph is indeed non-bipartite. We can use such an algorithm as a preliminary step for an exact decision procedure: if the algorithm rejects, then we say that the graph is not bipartite, and if the algorithm accepts then we run the exact decision procedure. Thus we save time on “bad” inputs.

The Algorithm

The algorithm is very simple:

1. Take a sample S of $\Theta((1/\epsilon^2) \log(1/\epsilon))$ vertices, selected uniformly at random;

2. Ask vertex-pair queries for all pairs in the sample, thus obtaining the induced subgraph G_S ;
3. Run BFS on G_S : if it is bipartite then **accept**, otherwise, **reject**.

A central thing to note: the number of queries performed is *independent* of the size of the graph, and only depends (polynomially) on $1/\epsilon$.

Clearly, if the graph G is bipartite then it is accepted with probability 1, and when the algorithm rejects a graph it provides evidence “against” the graph in the form of a small subgraph (G_S) that is not bipartite.. Hence, from this point on assume G is ϵ -far from being bipartite, and we will show that it is rejected with probability at least $2/3$.

The Analysis

Recall: if G is ϵ -far from bipartite then this means that *for all* partitions (V_1, V_2) of V , there are more than ϵn^2 edges between vertices in the same part. Consider the following first attempt for analyzing the algorithm: If we consider a single partition (V_1, V_2) that has more than ϵn^2 violating edges, then it is easy to see that a sample of size $m = \Theta((1/\epsilon) \log(1/\delta))$ will hit such an edge with probability at least $1 - \delta$: think of selecting the vertices in pairs, where we want to “catch” a pair of vertices u, w that belong to the same side of the partition and have an edge between them. The probability that we don’t catch such an edge is $(1 - \epsilon)^m < e^{-\epsilon m} \leq \delta$. The natural idea would be to take a union bound over all partitions. The problem is that there are 2^n possible partitions and so in order for the union bound to work would have to take $\delta < 2^{-n}$ implying that the sample should have size linear in n ...

Instead, we’ll think of the sample as consisting of two disjoint parts, U and W . The intuition is that in some sense U will reduce the number of “relevant” partitions of V to a much smaller number than 2^n , and then W will be used to “test” only them. We shall let $|U| = \Theta(\log(1/\epsilon)/\epsilon)$ and $|W| = \Theta(|U|/\epsilon) = \Theta(\log(1/\epsilon)/\epsilon^2)$.

We’ll first need a couple of additional definitions:

Definition 3 *For any fixed partition (U_1, U_2) of U , we shall say that W is not compatible with (U_1, U_2) if there is no partition (W_1, W_2) of W such that $(U_1 \cup W_1, U_2 \cup W_2)$ is bipartite.*

We would like to show that (since G is ϵ -far from bipartite), with high probability over the choice of U and W , no matter how we partition U into (U_1, U_2) , the subset W will not be compatible with (U_1, U_2) (implying that there is no bipartite partition of both U and W , which causes the algorithm to reject).

Definition 4 *Let (U_1, U_2) be a (bipartite) partition of U . We shall say that a vertex w is a witness against (U_1, U_2) if there exist $u_1 \in U_1$ and $u_2 \in U_2$ such that $(w, u_1), (w, u_2) \in E$. We shall say that a pair w_1 and w_2 are witnesses against (U_1, U_2) if $(w_1, w_2) \in E$ and there exist $u_1, u_2 \in U$ such that $u_1, u_2 \in U_1$ or $u_1, u_2 \in U_2$ and $(w_1, u_1), (w_2, u_2) \in E$.*

For an illustration of witnesses, see Figure 1.

Observation: If W contains a vertex w that is a witness against (U_2, U_2) or a pair of vertices w_1 and w_2 that are witnesses against (U_1, U_2) then W is not compatible with (U_1, U_2) . Hence, we

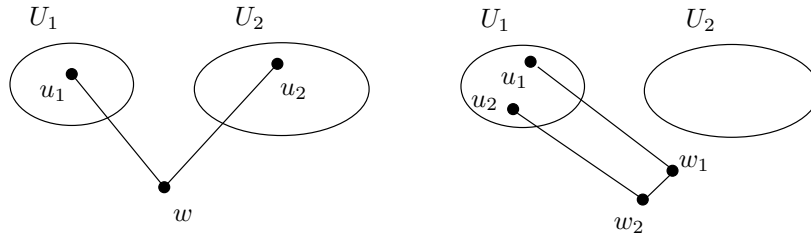


Figure 1: An illustration of a witness w , and a pair of witnesses w_1, w_2 , both with respect to the partition (U_1, U_2) of U .

would like to show that with high probability over U and W , there are witnesses in W against *every* partition of U .

Simplifying assumption: We first continue the analysis under the assumption that U is such that *every* $v \in V$ has at least one neighbor in U . (We later remove this assumption). Under this assumption, given a bipartite partition (U_1, U_2) of U , we define a partition of all of V . For $v \in U$ we put $v \in V_1$ if $v \in U_1$ and $v \in V_2$ if $v \in U_2$. For $v \in V \setminus U$ (that is, almost all vertices are considered here) if v has a neighbor in U_1 then we put v in V_2 and otherwise (it has a neighbor in U_2), then we put it in V_1 . For an illustration, see Figure 2.

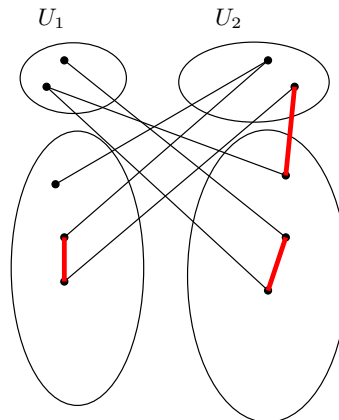


Figure 2: An illustration of the partition of V that is defined based on (U_1, U_2) when we make the simplifying assumption that every vertex in V has a neighbor in U . Violating edges (which correspond to witnesses) are marked by bold (red) lines.

Now, each one of these at most $2^{|U|}$ partitions of V contains more than en^2 violating edges. Since (U_1, U_2) is bipartite, and we put each vertex in $V \setminus U$ opposite its neighbor, these edges are of the form $(w_1, w_2) \in E$ where w_1 and w_2 both have a neighbor in U_1 or both have a neighbor in U_2 , or they are of the form (w, u_2) where $u_2 \in U_2$ and w has a neighbor $u_1 \in U_1$ (so it was put in V_2). But this exactly coincides with our definition of witnesses against (U_1, U_2) , and so if we catch such a vertex (pair) then W is not compatible with (U_1, U_2) . For simplicity of the analysis, even in the case that w is a witness because it was put in V_1 but it has a neighbor $u_2 \in U_2$, we shall think of

(u_2, w) as a pair of witnesses, and so it won't be considered sufficient that $w \in W$ but we'll require that $u_2, w \in W$.

We'll think of the uniform sample W as a sample over uniformly selected pairs of vertices. Since the probability that we catch a pair of witnesses in a single trial is more than $\frac{\epsilon n^2}{n^2} = \epsilon$, the probability that we *don't* catch any pair of witnesses in W is at most $(1 - \epsilon)^{|W|/2}$. If we take $|W| = \Theta(|U|/\epsilon)$ then this is less than $(1/6) \cdot 2^{-|U|}$. By a union bound, the probability that for some (U_1, U_2) , we have that W is compatible with (U_1, U_2) is hence at most $1/3$. In other words, with probability at least $5/6$ there is no bipartite partition of $U \cup W$.

It remains to remove the assumption that every vertex in V has a neighbor in U .

Definition 5 We say that a vertex in V has high degree if its degree is at least $(\epsilon/4)n$. Otherwise it has low degree.

Lemma 1 With probability at least $5/6$ over the choice of $(4/\epsilon) \log(24/\epsilon)$ vertices (denoted U), all but at most $(\epsilon/4)n$ of the high degree vertices in V have a neighbor in U .

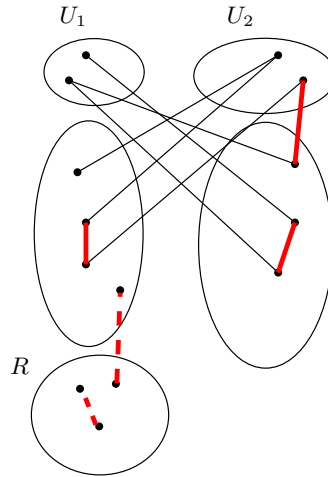


Figure 3: An illustration of the partition of V that is defined based on (U_1, U_2) when we remove the simplifying assumption that every vertex in V has a neighbor in U . Violating edges that are incident to R are marked by dashed lines while violating edges which correspond to witnesses are marked by bold lines.

We prove this lemma momentarily, but first let us see how we modify the argument based on the lemma. Assume U is as stated in the lemma (where we later take into account the probability of $1/6$ that this is not the case). Then, given a partition (U_1, U_2) of U we define a partition of all vertices similarly to what we did before. In particular, the vertices in U and their neighbors are partitioned as before. All remaining vertices, whose set is denoted R are put arbitrarily in V_1 . For an illustration, see Figure 3. Once again, for every (U_1, U_2) , the partition of V just defined contains more than ϵn^2 violating edges. Now, some of them might not correspond to witnesses. In particular, some of these violating edges might be incident to vertices in R . However, the total

number of edges that are incident to vertices in R is at most $n \cdot \frac{\epsilon}{4}n + \frac{\epsilon}{4}n \cdot n = \frac{\epsilon}{2}n^2$. Hence, there are at least $\frac{\epsilon}{2}n^2$ violating edges that correspond to witnesses, we shall catch one with probability.

More precisely, if $|W| = \Theta(|U|/\epsilon) = \Theta(\log(1/\epsilon)/\epsilon^2)$, then, conditioned on U being as in Lemma 1, with probability at least $5/6$ over the choice of W , there are a pair of witnesses in W against every partition of U . The probability that either U is not as in Lemma 1, or W does not include witnesses against some partition of U , is at most $1/3$. It follows that with probability at least $2/3$ (over the choice of $S = U \cup W$) the algorithm rejects (since there is no bipartite partition of S). It remains to prove Lemma 1.

Proof of Lemma 1: Consider any fixed high degree vertex v . The probability that U does not contain any neighbor of v is at most $(1 - (\epsilon/4))^{|U|} < \epsilon/24$. Therefore, the expected fraction of high degree vertices in V that do not have a neighbor in U is at most $\epsilon/24$. What is the probability that there are more than $\epsilon/4$ fraction such vertices in V ? By Markov's inequality, the probability that we are 6 times the expected value is at most $1/6$. ■

REDUCING THE NUMBER OF QUERIES. A more sophisticated analysis can show that actually a sample of size $\Theta(\log(1/\epsilon)/\epsilon)$ suffices (so that the number of queries is $\Theta(\log^2(1/\epsilon)/\epsilon^2)$).

ESTIMATING THE DISTANCE FROM BEING BIPARTITE. Instead of asking that the algorithm distinguish between graphs that are bipartite and those that are ϵ -far from bipartite, we might want something a bit stronger: estimate the minimum number of edges that should be removed so as to make the graph bipartite. This can be done but requires a more involved analysis.