Sublinear Time and Space Algorithms 2018B – Lecture 11
Sublinear-Time Algorithms for Sparse Graphs*

Robert Krauthgamer

1 Approximating Average Degree in a Graph

Problem definition:

Input: An n-vertex graph represented (say) as the adjacency list for each vertex (or even just the degree of each vertex)

Goal: Compute the average degree (equiv. number of edges)

Concern: Seems to be impossible e.g. if all degrees ≤ 1, except possibly for a few vertices whose degree is about n.

Theorem 1 [Feige, 2004]: There is an algorithm that estimates the average degree d of a *connected* graph within factor $2+\varepsilon$ in time $O((\frac{1}{\varepsilon})^{O(1)}\sqrt{n/d_0})$, given a lower bound $d_0 \leq d$ and $\varepsilon \in (0,1)$.

We will prove the case of $d_0 = 1$ (i.e., suffices to know G is connected).

Main idea: Use the fact that it is a graph (and not just a list of degrees), although this will show up only in the analysis.

Algorithm:

1. Choose a set S by choosing at random $s = c\sqrt{n}/\varepsilon^{O(1)}$ vertices, and compute the average degree d_S of these vertices.

2. Repeat the above $8/\varepsilon$ times, and report the smallest seen d_S.

Analysis: We will need 2 claims.

Claim 1a: In each iteration, $\Pr[d_S < (\frac{1}{2}-\varepsilon)d] \leq \varepsilon/64$.

Claim 1b: In each iteration, $\Pr[d_S > (1+\varepsilon)d] \leq 1-\varepsilon/2$.

Proof of theorem: Follows easily from the two claims, as seen in class.

*These notes summarize the material covered in class, usually skipping proofs, details, examples and so forth, and possibly adding some remarks, or pointers. The exercises are for self-practice and need not be handed in. In the interest of brevity, most references and credits were omitted.

Proof of Claim 1b: Follows from Markov's inequality, as seen in class.

Proof of Claim 1a: Was seen in class. Here we really used the fact the degrees form a graph.

Exer: Explain how to extend the result to any $d_0 \geq 1$.

2 Maximum Matching

Problem definition:

Input: An n-vertex graph $G = (V, E)$ of maximum degree D, represented as the adjacency list for each vertex.

Definition: A matching is a set of edges that are incident to distinct vertices.

Goal: Compute the maximum size of a matching in G.

Note: The matching is too large to report in sublinear time, we only estimate its cost using (α, β)-approximation, i.e., $OPT \leq ALG \leq \alpha\ OPT + \beta$.

Theorem 2 [Nguyen and Onak, 2008]: There is an algorithm that gives $(2, \varepsilon n)$ approximation to the maximum matching size in time $D^{O(D)}/\varepsilon^2$.

Main idea: It is well-known that maximal matching (note: maximal means with respect to containment) is a 2-approximation for maximum matching. We will fix one such matching almost implicitly, and then estimate its size by sampling.

Algorithm GreedyMatching:

1. Start with an empty matching M.

2. Scan the edges (in arbitrary order), and add each edge to M unless it is adjacent to an edge already in M.

Lemma 2a: The size of a maximal matching is at least half that of a maximum matching.

Proof: Exercise

To be continued in the next class.

Continuation (done in the next class)

Algorithm ApproxGreedyMatching:

1. choose random edge priorities $p(e) \in [0, 1]$, implicitly defining a permutation of the edges

2. choose $s = O(D/\varepsilon^2)$ edges $e_1, \ldots, e_s$ uniformly at random from the Dn possibilities (note that each edge has two "chances" to be chosen, and some choices may lead to no edge, if the actual degree is smaller than D)

3. for each edge e_i, compute an indicator X_i for whether e_i belongs to the maximal matching

corresponding to p, by exploring the neighborhood of e_i incrementally

[stop if the algorithm took too many steps altogether]

4. report $X = \frac{Dn}{2s} \sum_i X_i$

Running time: Let M be a greedy matching constructed according to the priorities p. As seen in class, to determine whether a single $e_i \in M$, whp it suffices to explore up to radius $k = O(D)$. Moreover, the expected running time is $O(s \cdot D^k) \leq D^{O(D)}/\varepsilon^2$, and by Markov's inequality the probability to exceed it by much is small.

Correctness: As seen in class, it follows by applying Chebychev's inequality to $X = \frac{Dn}{2s} \sum_i X_i$.