Lecture 9

Lecturer: Sofya Raskhodnikova Scribe(s): Sandeep Narayanaswamy, Kashyap Dixit, Nai-Hui Chia, Nithin M. Varma

1 Max-cut additive approximation scheme

In the previous lectures, we have seen a sublinear algorithm to test if a graph is bipartite or is ϵ-far from bipartite. Here we study an algorithm to approximate the size of maximum-cut in dense graphs with additive error. The algorithm is due to Goldreich, Goldwasser and Ron [GGR98]. We will start with some basic definitions.

Definition 1. *Let $G = (V, E)$ be an undirected unweighted graph and (V_1, V_2) be a partition of V. The* edge density *of a cut (V_1, V_2), denoted by $\mu(V_1, V_2)$, is*

$$\mu(V_1, V_2) = \frac{|e(V_1, V_2)|}{n^2},$$

where $e(V_1, V_2)$ is the set of edges with one endpoint in V_1 and the other in V_2.

The edge density μ is normalized to lie in $[0, 1]$, but in fact never exceeds $\frac{1}{4}$: the cut with the largest edge density results from a complete bipartite graph with $n/2$ vertices in each part of the partition. It may also result from a bipartition of a clique K_n on n vertices, with $n/2$ vertices in each part.

Definition 2. *The edge density of the largest cut in G, denoted by $\mu(G)$, is*

$$\mu(G) = \max_{partitions\ (V_1,V_2)} \mu(V_1, V_2).$$

The problem that we will consider today and in the upcoming two lectures is the following. Given a graph $G = (V, E)$, output a value $\tilde{\mu} \in (0, 1)$ and a cut (V_1, V_2) such that $\mu(V_1, V_2) = \tilde{\mu}$ such that $\tilde{\mu} \geq \mu(G) - \epsilon$ w.p. at least $5/6$ and $\tilde{\mu} \leq \mu(G)$ w.p. 1.

1.1 The Approximation Algorithm

We will see the approximation algorithm that has oracle access to the graph, approximates $\mu(G)$ with an additive error ϵ, i.e., outputs an approximate value $\tilde{\mu}$ such that $\tilde{\mu} \geq \mu - \epsilon$ with probability $\geq \frac{5}{6}$, runs in time $O(n \cdot 2^{poly(\frac{1}{\epsilon})}))$ and with $O(n\frac{1}{\epsilon^2} \log \frac{1}{\epsilon})$ queries. Note that $\tilde{\mu} \leq \mu$ is always true since μ is the maximum density of the cut.

Theorem 3. *For every ϵ, the algorithm returns a partition (V_1, V_2) of V such that $\mu(V_1, V_2) \geq \mu(G) - \epsilon$ with probability at least $5/6$.*

1.1.1 Basic Idea

Consider two subsets L_1 and L_2 of V. Let v be a vertex that is neither in L_1 nor L_2. If we wish to decide whether to place v in L_1 or L_2, with the goal of achieving a large edge density, a greedy way to do this would be to place v in the set where it has fewer neighbors. To make this formal, we introduce some notation: we denote by $N(v, U)$ the set of neighbors a vertex v has in set U and let $\Gamma(v, U) = |N(v, U)|$. Now the greedy idea is: if $\Gamma(v, L_1) > \Gamma(v, L_2)$, we add v to L_2; otherwise, we add it to L_1.

1.1.2 A preliminary algorithm that approximates Max-Cut

1. Partition V into $\ell = \frac{1}{\epsilon}$ sets of (almost) equal size, $V^1, V^2, \ldots, V^\ell$, i.e., $|V_i| = \epsilon n$ for all $i \in [n]$. (To make things simpler, we assume these sets are of equal size.) We will consider each V^i separately.

2. For each $i \in [\ell]$, select a set L^i of size $\frac{1}{\epsilon^2} \cdot \log \frac{1}{\epsilon}$ uniformly at random from $V \backslash V_i$, the set of vertices not in V_i. We denote the ℓ-tuple of these sets by L, that is, $L = (L^1, L^2, ..., L^\ell)$.

3. Note that there are $2^{|L^j|} = 2^{\frac{1}{\epsilon^2} \cdot \log \frac{1}{\epsilon}}$ possible partitions of the set L^j into (L_1^j, L_2^j). Also, there are $O((2^{|L^i|})^\ell) = O(2^{\frac{1}{\epsilon^3} \cdot \log \frac{1}{\epsilon}})$ partition sequences $\pi(L) = ((L_1^1, L_2^1), \ldots (L_1^\ell, L_2^\ell))$. For each partition sequence π, partition the whole vertex set V greedily, as follows:

   - For each $i \in [\ell]$, partition V^i into (V_1^i, V_2^i) using the greedy rule: for every vertex $v \in V^i$, if $\Gamma(v, L_1^i) > \Gamma(v, L_2^i)$, put v in V_2^i, and otherwise in V_1^i. See Figure 1.
   - Define $V_1^\pi = \bigcup_i V_1^i$ and $V_2^\pi = \bigcup_i V_2^i$. (Thus, every partition π of L induces a partition (V_1^π, V_2^π) of V.) Calculate the edge density for this cut, $\mu(V_1^\pi, V_2^\pi)$.

4. Output the partition with the largest density, which we denote as $\tilde{\pi}$.

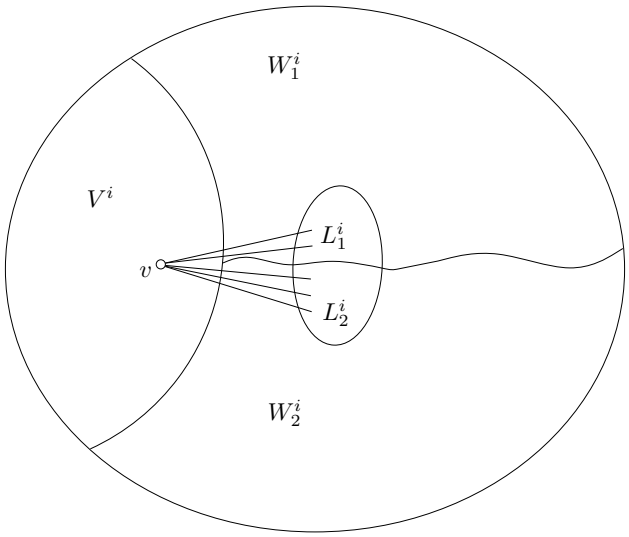

Figure 1: Greedy strategy will place v in L_1^i. Note that $W^i = V \backslash V^i$.

This algorithm samples the sets $L^1, L^2, \ldots, L^\ell$ only once, but considers all possible partitions of L. Since the sets L^i are fixed, we make the queries (v, u) for all $v \in V$ and $u \in V_i$ and store the answers in our local memory. That is, to make all our greedy choices, for each vertex $v \in V_i$, we use $|L_i|$ queries. Thus, the total number of queries is bounded by $\sum_i |L_i||V_i| = O((\frac{1}{\epsilon^2} \cdot \log \frac{1}{\epsilon})n)$. The time complexity is bounded by $O(2^{\ell|L_1|} \frac{1}{\epsilon^2} \cdot \log \frac{1}{\epsilon} \cdot n) = O((2^{\frac{1}{\epsilon^3} \cdot \log \frac{1}{\epsilon}} \frac{1}{\epsilon^2} \cdot \log \frac{1}{\epsilon})n)$.

1.2 Correctness

Lemma 4. *Let (H_1, H_2) be a cut of V. Then, with probability $\geq 5/6$ over the choice of L, some partition sequence $\pi(L)$ is **good**, i.e., it induces a partition (V_1^π, V_2^π) of V such that $\mu(V_1^\pi, V_2^\pi) \geq \mu(H_1, H_2) - \frac{3\epsilon}{4}$.*

Note that Lemma 4 holds for every cut (H_1, H_2), so in particular, it holds for the max-cut as well.

Proof. The proof goes via a hybrid argument. We define the i^{th} hybrid partition (H_1^i, H_2^i) as follows. Let $(H_1^0, H_2^0) = (H_1, H_2)$ and (W_1^1, W_2^1) be the partition of $W^1 = V \backslash V^1$ induced by (H_1, H_2). Similarly, we define (W_1^i, W_2^i) to be the partition of $V \backslash V^i$ such that the sets $V^1, V^2, .., V^{i-1}$ are partitioned according to our algorithm and $V^{i+1}, V^{i+2}, ..., V^\ell$ are partitioned according to (H_1, H_2). The partition (H_1^{i-1}, H_2^{i-1}) is depicted in Figure 2. We define $H_1^i = W_1^i \cup V_1^i$ and $H_2^i = W_2^i \cup V_2^i$.

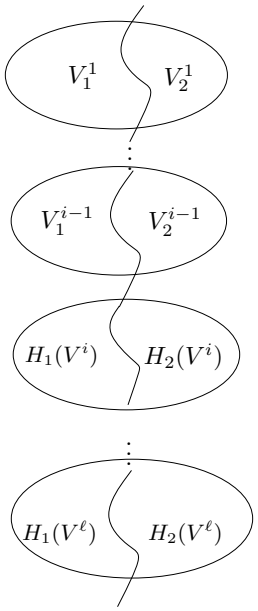

Figure 2: Partition (H_1^{i-1}, H_2^{i-1}): the vertex set V^{i-1} is being re-partitioned according to the algorithm.

Simply stated, in partition (H_1^i, H_2^i), the nodes in $V^1, ..., V^i$ are partitioned according to our algorithm and nodes in $V^{i+1}, ..., V^\ell$ are partitioned according to (H_1, H_2). We will argue that the density drop between successive hybrid partitions is small. More precisely, suppose we already picked $L_1, \ldots, L_{i-1}$; we want to argue that with probability at least $1 - \frac{1}{6\ell}$ over the choice of L_i, we get $\mu(H_1^i, H_2^i) \geq \mu(H_1^{i-1}, H_2^{i-1}) - \frac{3\epsilon}{4\ell}$. Note that if this statement holds *for every* $i \in [\ell]$, it implies the lemma, since the inequality in the lemma can be obtained by summing up the inequalities for all $i \in [\ell]$.

Claim 5. *Fix $i \in [n]$ and $L_1, \ldots, L_{i-1}$. Let A_i be the event that $\mu(H_1^i, H_2^i) \geq \mu(H_1^{i-1}, H_2^{i-1}) - \frac{3\epsilon}{4\ell}$. Then* $\Pr[\overline{A_i}] \leq \frac{1}{6\ell}$.

Here, we want to prove that all events $\overline{A_i}$ happen with probability $\leq \frac{1}{6\ell}$. This implies that $\Pr[\bigcup_i \overline{A_i}] \leq 1/6$, or say $\Pr[\bigcap_i A_i] \geq 5/6$.

Proof. We prove this by the following accounting argument. We say that a vertex is *bad* w.r.t. (L_1^i, L_2^i) if $\left|\frac{\Gamma(v,L_j^i)}{|L^i|} - \frac{\Gamma(v,W_j^i)}{|W^i|}\right| > \frac{\epsilon}{32}$ for some $j \in [2]$; otherwise, the vertex is *good.* A sample L^i is *good* with respect to (W_1^i, W_2^i) if at most $\frac{\epsilon}{8}$ fraction of vertices in V^i are bad. Assume in fact that for each i, the set L^i is good for (W_1^i, W_2^i). We will later see in Claim 7 that this happens with high probability. We call a vertex *unbalanced* with respect to (W_1^i, W_2^i) if $\frac{\Gamma(v,W_j^i)}{|W^i|} - \frac{\Gamma(v,W_{j'}^i)}{|W^i|} \geq \frac{\epsilon}{8}$ for $j \neq j'$ and $j, j' \in \{1, 2\}$. Suppose a vertex $v \in V^i$ is unbalanced and good:

$$\begin{aligned}\frac{\epsilon}{8} &\leq \frac{\Gamma(v,W^i_j)}{|W^i|} - \frac{\Gamma(v,W^i_{j'})}{|W^i|} \\ &\leq \frac{\Gamma(v,L^i_j)}{|L|} + \frac{\epsilon}{32} - \left(\frac{\Gamma(v,L^i_{j'})}{|L|} - \frac{\epsilon}{32}\right) \\ &= \frac{\Gamma(v,L^i_j)}{|L|} - \frac{\Gamma(v,L^i_{j'})}{|L|} + \frac{\epsilon}{16}.\end{aligned}$$

Thus, $\Gamma(v,L^i_j)) \geq \Gamma(v,L^i_{j'})) + \frac{1}{16}\epsilon|L|$, which means that an unbalanced good vertex always has one partition L^i_j with surplus neighbors as against the other partition $L^i_{j'}$. From our assumption, we also have that $\Gamma(v,W^i_j)) \geq \Gamma(v,W^i_{j'})) + \frac{\epsilon|W^i|}{8}$ (noting that the order of j and j' in the inequalities is the same). This fact implies that the unbalanced and good vertices are in the same side with respect to (L^i_1, L^i_2) after repartitioning V^i.

Thus, we have the following useful observation.

Observation 6. *When the partition (L^i_1, L^i_2) is used, v is put opposite to the majority of neighbors in W^i.*

Note that in each step i, only the partition of V^i is changed. Thus, the potential reduction of cut-edges may be caused only between the partitions V^i and W^i.

Now we upper bound the number of cut-edges that we loose by re-partitioning the vertices of V^i, i.e. going from hybrid partition (H^{i-1}_1, H^{i-1}_2) to (H^i_1, H^i_2).

1. The number of cut-edges due to *unbalanced* vertices that are *good* remains unchanged. This follows from observation 6.
2. The number of cut-edges due to *unbalanced* vertices that are *bad* decreases by at most $\frac{\epsilon}{8}|V_i|n \leq \frac{\epsilon}{8\ell}n^2$
3. The number of cut-edges due to *balanced* vertices decreases by at most $\frac{\epsilon}{8}|W^i||V^i| \leq \frac{\epsilon}{8\ell}n^2$.
4. Edges lost inside V^i are at most $|V^i|^2/4 \leq \frac{n^2}{4\ell^2} \leq \frac{\epsilon}{4\ell}n^2$.

Hence, the total number of cut-edges lost following the re-partitioning of V^i is $\frac{3\epsilon}{4\ell}n^2$. □

Hence, the loss in density at each step is at most $\frac{3\epsilon}{4}$. □

It remains to show that each set L^i is good with high probability.

Claim 7. *Probability that there exists $i \in [\ell]$ such that the set L^i is bad is at most $\frac{1}{6}$.*

Proof. Let $L^i = \{u_1, u_2, ..., u_{|L|}\}$. First fix a vertex $v \in V^i$. Recall that L^i is chosen uniformly at random from $W^i = V \backslash V^i$. Define the indicator random variable

$$X^k_j = \begin{cases} 1 & \text{if } u_k \in N(v, W^i_j); \\ 0 & \text{otherwise.} \end{cases}$$

.

Note that $X_j = \sum_k X^k_j$ is the number of neighbors of v in L^i_j and the probability that $X^k_j = 1$ is $\frac{1}{n}\Gamma(v,W^i_j)$ $(j \in \{1,2\})$. Thus, by additive Chernoff bound,

$\Pr\left[\left|\frac{\Gamma(v,L^i_j)}{|L|} - \frac{\Gamma(v,W^i_j)}{|W^i|}\right| > \frac{\epsilon}{32}\right] = exp(-\Omega(\epsilon^2|L|)) \leq \epsilon/96\ell$. By Markov's inequality, the probability of getting more than $\frac{\epsilon}{8}$ fraction of bad vertices is upper bounded by $\frac{1}{6\ell}$. □

1.3 Improved Algorithm for Approximating Max-cut Density

1. Partition V into $\ell = \frac{4}{\epsilon}$ sets of (almost) equal size, $V^1, V^2, \ldots, V^\ell$. (To make things simpler, we assume these sets are of equal size.) We will consider each V^i separately.

2. For each i in $\{1, 2, \ldots, \ell\}$, select a set L^i of size $t = \frac{1}{\epsilon^2} \cdot \log \frac{1}{\epsilon}$ uniformly at random from $V \backslash V_i$, the set of vertices not in V_i. We denote the ℓ-tuple of these sets by L: $L = (L^1, L^2, ..., L^\ell)$.

3. Select uniformly $S = \{s_1, ..., s_m\}$ of size $m = \Theta(\frac{\ell t}{\epsilon^2})$

   - For each partition sequence $\pi(L) = ((L_1^1, L_2^1), \ldots (L_1^\ell, L_2^\ell))$ partition S^i into two disjoint sets S_1^i and S_2^i and let $S_j^{\pi(L)} = \bigcup_{i=1}^{\ell} S_j^i$ (for $j = 1, 2$)
   - For each partition $(S_1^{(\pi(L))}, S_2^{(\pi(L))})$ compute the fraction of the cut-edges between pairs of vertices (S_{2k-1}, s_{2k}). More precisely, define $\hat{\mu}(s_1^\pi, \mu(s_2^\pi)) = \frac{(s_{2k-1}, s_{2k}) \in e(S_1^\pi, S_2^\pi)}{m/2}$

4. Output $\hat{\mu}_{max} = max_\pi \hat{\mu}(S_1^\pi, S_2^\pi)$.

References

[GGR98] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998. Preliminary version in 37th FOCS, 1996.