CS 7880: Algorithms for Big Data (Fall'22)	**Northeastern University**

Lecture 8

October 7, 2022

Instructor: Soheil Behnezhad *Scribe: Omer Wasim*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications.*

1 Sublinear Algorithms for Maximum Matching (Part I)

In the last lecture, we gave a sublinear time algorithm to determine whether a random vertex v is contained in a maximal independent set constructed using a greedy procedure under a random permutation of the vertex set V. In this lecture, we give an algorithm to (approximately) compute the size of a maximum matching in $\widetilde{O}(\Delta^3)$ time, by using the algorithm from the last lecture as a subroutine.

Our goal will be to prove the following theorem that appeared in a seminal paper by Yoshida, Yamamoto and Ito [1].

Theorem 1. *Let $G = (V, E)$ be a graph of maximum degree Δ and let $\mu(G)$ denote the size of the maximum matching of G. For any $\epsilon > 0$, there is an $\tilde{O}(\Delta^3)$ time algorithm to compute a $(\frac{1}{2}, \epsilon)$-approximation of $\mu(G)$ w.h.p in the adjacency list model. More precisely, the output $\tilde{u}$ of the algorithm satisfies,*

$$\frac{1}{2}\mu(G) - \epsilon n \le \tilde{\mu} \le \mu(G)$$

> **Remark.** We note that the actual bound claimed in [1] is $\widetilde{O}(\Delta^4)$, which is slightly slower than the claimed bound of $\widetilde{O}(\Delta^3)$ in Theorem 1. To achieve a bound of $\widetilde{O}(\Delta^3)$ we incorporate one additional idea due to [2] which is to sample vertices and test whether they belong to matching instead of sampling edges. In the next lecture, we will discuss how we can further improve this to $\widetilde{O}(\bar{d})$ where $\bar{d}$ is the average degree of the graph.

1.1 Recap

Recall the local query process from last lecture that we used to determine whether a vertex v is contained in the MIS $\mathcal{I}$ computed via the Greedy MIS algorithm under a permutation π of V:

`IsInMIS`(v, G, π)):

1. Let $u_1, u_2, ..., u_d$ be neighbors of v s.t. $\pi(u_1) < \pi(u_2) < .... < \pi(u_d) < \pi(v)$.
2. **For** $i = 1$ to d:
   - **If** `IsInMIS` $(u_i, G, \pi) =$ True:
     return False.
3. **return** True.

We defined $Q(v, \pi)$ as the number of recursive calls to the `IsInMIS` procedure when calling `IsInMIS`(v, G, π), and sketched a proof of the following theorem.

Theorem 2. *[1] For a random vertex v and a random permutation π over V, the expected value of $Q(v,\pi)$ satisfies,*

$$\mathop{\mathbb{E}}_{v,\pi}[Q(v,\pi)] \leq 1 + \frac{m}{n}$$

or equivalently,

$$\sum_{v} \mathop{\mathbb{E}}_{\pi}[Q(v,\pi)] \leq n + m.$$

2 Matchings and Line Graphs

We begin by recalling the definition of a matching.

Definition 3. (Matching) A matching $\mathcal{M} \subseteq E$ of a graph $G = (V,E)$ is a collection of vertex disjoint edges of G, i.e. for any $e = (u,v) \in \mathcal{M}$, u and v are not in the set $\cup_{e'=(s,t)\in\mathcal{M}\setminus\{e\}}\{s,t\}$.

Definition 4. (Maximal Matching) A matching $\mathcal{M}$ is a maximal matching of graph $G = (V,E)$ if $\mathcal{M} \not\subset \mathcal{M}'$ for any matching $\mathcal{M}'$ of G.

Claim 5. *The size of a maximal matching $\mathcal{M}$ is a $\frac{1}{2}$-approximation of the size of a maximum matching $\mu(G)$, i.e. $|\mathcal{M}| \geq \frac{1}{2}\mu(G)$.*

Proof. Fix a maximum matching $\mathcal{M}'$. For any edge $e = (u,v) \in \mathcal{M}'$, at least one of u and v must be matched in $\mathcal{M}$ (since otherwise $\mathcal{M}$ would fail to be maximal). Thus, the number of matched vertices in $\mathcal{M}$ is at least $\mu(G)$, which implies that $|\mathcal{M}| \geq \frac{1}{2}\mu(G)$. □

We now define the line graph $L(G)$ of a graph $G = (V,E)$.

Definition 6. (Line Graph) Given a graph $G = (V,E)$, such that $E = \{e_1, e_2, ..., e_{|E|}\}$, the line graph $L(G) = (V', E')$ of G is the graph on vertex set $V' = v_1, v_2, ..., v_{|E|}$ where each vertex v_i corresponds to an edge e_i, and edge set $E' = \{(v_i, v_j) | e_i, e_j \text{ share an endpoint in } G\}$.

The key property of line graphs that we exploit in order to give a sublinear time algorithm for computing the size of a maximum matching is that any MIS on the line graph of G corresponds to a maximal matching of G.

Claim 7. *For any graph G, any MIS $\mathcal{I}$ of $L(G)$ corresponds to a maximal matching of G.*

Proof. Since $\mathcal{I}$ is an independent set in $L(G)$, then for any $v_i, v_j \in \mathcal{I}$ we have that e_i, e_j *do not* share an endpoint in G. Thus, $\mathcal{I}$ corresponds to a matching in G. Indeed, it is a maximal matching since $\mathcal{I}$ is a maximal independent set of $L(G)$ (note that if $\mathcal{I}$ does not correspond to a maximal matching in G, then one can add another edge-say e_k to the matching. This would imply that v_k can be added to $\mathcal{I}$, contradicting that $\mathcal{I}$ is a maximal independent set). □

Remark. The big picture here is that to estimate the size of a maximum matching of G, we can estimate the size of a maximal independent set in $L(G)$ by going over the vertices of $L(G)$ (or edges of G) in some order. If `IsInMIS`$(v_i, L(G), \pi)$ returns True, we can add the corresponding edge e_i to the maximal matching $\mathcal{M}$. By sampling random vertices and repeating this process, we get an estimate for the size of the maximal matching which is sufficient for a $(\frac{1}{2}, \epsilon)$-approximation to $\mu(G)$.

3 The Algorithm

Before giving the full algorithm, we begin by considering an analogous algorithm to the greedy maximal independent set adapted for computing the maximal matching. The greedy algorithm simply works as follows: beginning with an empty matching $\mathcal{M}$, if an edge e can be added to $\mathcal{M}$, add e to $\mathcal{M}$. If no edges can be added to $\mathcal{M}$, then by definition $\mathcal{M}$ is maximal.

We give the following procedure which utilizes the subroutine `IsInMIS` to determine whether a vertex v is matched by the greedy maximal matching algorithm under an input permutation π on *edges* of G:

We revise the IsInMIS procedure to draw ranks of vertices (of the line graph) on the fly.
`IsInMIS`(v, G)**):**

1. Let $u_1, u_2, ..., u_d$ be neighbors of v. For all $i \in [d]$, draw $\pi(u_d) \in [0, 1]$ uniformly in random (if $\pi(u_i)$ has not been drawn before).
2. **For** $i = 1$ to d:
   - **If** `IsInMIS` $(u_i, G, \pi) =$ True:
     return False.
3. **return** True.

`IsInMM(`v, G, π`))`:

1. **For** all edges e incident to v:
   - **If** `IsInMIS` $(e, L(G)) =$ True:
     return True.
2. **return** False.

Lemma 8. *For a random permutation π over E and a random vertex v, the procedure `IsInMM` (v, G, π) takes $O(\Delta^3)$ expected time.*

Proof. Let $e_1, ..., e_{|N(v)|}$ denote the set of edges incident to v. Then, the total running time of the calls to the `IsInMIS` subroutine is bounded by $2\Delta \sum_{i=1}^{|N(v)|} Q(e_i, \pi)$, where the multiplicative factor of 2Δ comes from generating the ranks on the line graph for a single call to the `IsInMIS` procedure. However, note that to bound each $Q(e_i, \pi)$, we cannot simply use the guaranteed of Theorem 2 since it holds for a randomly chosen vertex. Instead we will bound the sum itself. To this end, let $T(v, \pi) = \sum_{i=1}^{|N(v)|} Q(e_i, \pi)$. Observe that,

$$\sum_{v} \mathop{\mathbb{E}}_{\pi}[T(v, \pi)] = 2 \sum_{e \in E} \mathop{\mathbb{E}}_{\pi}[(Q(e, \pi)] \leq 2m(1 + \Delta) = O(m\Delta)$$

where the second to last inequality follows from Theorem 2 and the fact that the average degree of the line graph is bounded by its maximum degree (which is $2\Delta = O(\Delta)$). Then, for a randomly chosen vertex v, we have that

$$\mathop{\mathbb{E}}_{v,\pi}[T(v, \pi)] = \frac{1}{n} \sum_{v} \mathop{\mathbb{E}}_{\pi}[T(v, \pi)] = O(\frac{m\Delta}{n}) = O(\Delta^2)$$

Thus, the total running time due to the calls to the `IsInMIS` subroutine bounded by $O(\Delta^3)$, completing the proof. □

As noted earlier, the idea will be to sample a sufficiently large number of vertices independently and uniformly, and check whether the sampled vertices are matched in a greedy matching by calling the `IsInMM` procedure on each sampled vertex. The fraction of matched vertices in our sample is then used as a proxy for estimating the size of a maximal matching- which by Claim 5 is a $\frac{1}{2}$-approximation of the size of a maximum matching. The size of the sample controls the final error in our approximation by a standard Chernoff bound.

`Approximate-MM`(G)**:**

1. Sample $k = \frac{12 \ln n}{\epsilon^2}$ vertices $v_1, ..., v_k$ independently and uniformly.
2. For $i = 1$ to k:
   - Let X_i=1 if `IsInMM`(v_i, G) returns True.
3. Let $X = \sum_{i=1}^{k} X_i$ and $f = \frac{X}{k}$.
4. **return** $\frac{fn}{2}$.

Lemma 9. *The output of algorithm* `Approximate-MM` *is within* $\frac{\epsilon n}{2}$ *of* $\mathbb{E}_\pi[|GMM(G,\pi)|]$*, the expected size of a greedy maximal matching under a random permutation* π*. More precisely,*

$$\frac{fn}{2} \in (\mathbb{E}_{\pi}[|GMM(G,\pi)|] \pm \frac{\epsilon}{2})$$

Proof. Note that $E[X_i] = E[X_1] = \frac{2\,\mathbb{E}_\pi[|GMM(G,\pi)|}{n}$. This is because the number of vertices incident to a maximal matching are twice the size of the matching and the fact that each X_i is chosen uniformly at random. From this, it follows that $E[X] = \frac{2k\,\mathbb{E}_\pi[|GMM(G,\pi)|}{n}$. Applying the Chernoff bound (additive version), we get that,

$$\Pr[|X - \mathbb{E}[X]| \geq \sqrt{12\,\mathbb{E}[X]\ln n}] \leq 2exp(-\frac{12\,\mathbb{E}[X]\ln n}{3\,\mathbb{E}[X]}) = \frac{2}{n^4}.$$

From this we have that $fn \in \frac{n}{k}(\mathbb{E}[X] \pm \sqrt{12\,\mathbb{E}[X]\ln n})$. Plugging in the value of $\mathbb{E}[X] = \mathbb{E}_\pi[|GMM(G,\pi)|]$, we get that $fn \in (2\,\mathbb{E}_\pi[|GMM(G,\pi)|] \pm n\sqrt{\frac{12 \ln n}{k}})$. Setting $k = \frac{12 \ln n}{\epsilon^2}$, we get that with high probability, $\frac{fn}{2} \in \mathbb{E}_\pi[|GMM(G,\pi)|] \pm \frac{\epsilon n}{2}$. □

Note that our estimate $\frac{fn}{2}$ could possibly be greater than the size of the maximum matching by $\frac{\epsilon n}{2}$ violating the guarantee of Theorem 1. The simple fix is to output $\tilde{\mu} = \frac{fn}{2} - \frac{\epsilon n}{2}$. This gives us an $\tilde{O}(\Delta^3)$ algorithm to compute a $(\frac{1}{2}, \epsilon)$-approximation of $\mu(G)$ w.h.p.

References

[1] Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. An improved constant-time approximation algorithm for maximum matchings. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 225–234, New York, NY, USA, 2009. Association for Computing Machinery. 1, 2

[2] Huy N. Nguyen and Krzysztof Onak. Constant-time approximation algorithms via local improvements. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 327–336. IEEE Computer Society, 2008. 1