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

Lecture 17

November 8th, 2022

Instructor: Soheil Behnezhad *Scribe: Dongyue Li*

Disclaimer: *These notes have not been edited by the instructor.*

1 Massive Parallel Computation Algorithms (Part III)

In the last two lectures, we give massive parallel computation (MPC) algorithms for finding maximum matching, minimum spanning trees, and vertex coloring. In this lecture, we will present an MPC algorithm for finding the maximum independent set (MIS). We will prove the following theorem of finding an MIS in $O(\log \log n)$ rounds.

Theorem 1. *With $O(n)$ space per machine, there is an MPC algorithm that w.h.p. finds an MIS in $O(\log \log n)$ rounds.*

1.1 Non-Parallel Algorithm

We will first discuss a non-parallel algorithm that finds an MIS in $O(\log n)$ rounds with high probability and then migrate the idea to design an MPC algorithm for finding an MIS.

Luby's algorithm for MIS

Until the graph becomes empty

1. On every remaining vertex v, draw a real $x_v \sim \text{Uniform}[0, 1]$ independently.
2. Any vertex v whose rank x_v is smaller than all of its remaining neighbors joins the independent set. We then remove v and all of its neighbors from the graph.

The result from Luby is that the algorithm terminates in $O(\log n)$ rounds with high probability as shown in the following.

Theorem 2. *The above algorithm terminates in $O(\log n)$ rounds with probability $1 - \frac{1}{n^{100}}$.*

Remark. There are graphs on which Luby's algorithm does require $\Omega(\log n)$ rounds with high probability.

1.2 MPC algorithm for finding MIS

We will see next that there is a $\log \log n$ round MPC algorithm. Consider the following two procedures.

`GreedyMIS`(G, π): Iterate over the vertices in the order of π and greedily add any vertex possible to the independent set.

`RandomGreedyMIS`(G):

1. Draw a random permutation π from the set of all permutation.
2. Return GreedyMIS(G, π).

With the above procedures, we present the MPC implementation of `RandomGreedyMIS`.

`MPC-RandomGreedyMIS`(G):

1. Draw a random permutation π from the set of all permutation.
2. Let $l = O(\log \log n)$
3. For $i = 1, \ldots, l$ do:
   - Let V_i be the subset of V including the first $\frac{\sqrt{n}}{\Delta}$ vertices.
   - Find a MIS on the subset V_i and remove the vertices in V_i and the neighbors of V_i.
4. Return the MIS which is $\text{MIS}(V_1) \cup \cdots \cup \text{MIS}(V_l)$.

Next, we will prove that the above MPC algorithm finds an MIS with high probability. We will base the proof on the following two lemmas.

Lemma 3. *Let G be a graph of maximum degree $\Delta >> \log n$ and let V' be a random subset of V including ever vertex independently with probability $\frac{1}{\sqrt{\Delta}}$. Then the number of edges in $G[V']$ is at most $O(n)$ w.h.p.*

Proof. Take a vertex v and let d'_v denotes the number of neighbors of v in V'. Then

$$\mathbb{E}\left[d'_v\right] = d_G(v)\frac{1}{\sqrt{\Delta}} \leq \sqrt{\Delta}.$$

Applying the Chernoff bound

$$\begin{aligned}\Pr\left[d'_v \geq \sqrt{\Delta} + \sqrt{6\sqrt{\Delta}\log n}\right] &\leq \exp\left(-\frac{6\sqrt{\Delta}\log n}{3\sqrt{\Delta}}\right) \\ &\leq \frac{1}{n^2}.\end{aligned}$$

With high probability, for all $v \in V$,

$$d'_v \leq \sqrt{\Delta} + O\left(\Delta^{1/4}\sqrt{\log n}\right) \leq O\left(\sqrt{\Delta}\right).$$

Moreover, w.h.p., we sample at most $O\left(\frac{n}{\sqrt{\Delta}}\right)$ vertices in V'. Thus the graph $G[V']$ has $O\left(\sqrt{\Delta}\right) \times O\left(\frac{n}{\sqrt{\Delta}}\right) = O\left(n\right)$ edges w.h.p. $\square$

This lemma indicates that we can put the subgraph $G[V']$ into one machine. The following lemma states that we can decay the maximum degree by using the procedure.

Lemma 4 (Sparsification Lemma)**.** *Let π be any arbitrary permutation over the vertex set V of a graph G. Let V' includes every vertex in V independently with probability p. Let I be the output of GreedyMIS$(G[V'], \pi)$. Let H be the graph obtained by removing I and every vertex adjacent to I from G. Then H has maximimum degree at most $O\left(\frac{\log n}{P}\right)$ wit probability $1 - \frac{1}{n^2}$.*

We first give an intuition to prove the Sparsification Lemma with a wrong proof.

Wrong proof. If $d_G(v) < 100\frac{\log n}{p}$ then we are done. However, if $d_G(v) \geq 100\frac{\log n}{p}$, the probability of seeing at least one neighbor of v in V' is $1-(1-p)^{100\log(n)/p} \geq 1-e^{-100\log n} = 1-\frac{1}{n^{100}}$. If v has less than $\frac{100\log n}{p}$ remaining neighbors, we are done. Otherwise, one of these remaining neighbors must have been sampled, which in that case it would join the independent set.

The above proof has not decoupled the probability of joining V' and I. Next, we present the formal proof that computes the probability by interpreting the procedure in another way.

Proof. Consider the following equivalent construction of I that reveals the subsample V' "on the fly".

1. Set $I = \emptyset$.
2. Iterate over V in the order of π. Upon visiting a vertex v,
   - If v has any neighbor in I, continue to the next vertex.
   - Else, we call v a potential vertex. Reveal whether $v \in V'$, and if so add v to I.

We have an observation from the above procedure: Every vertex in H is a potential vertex.

Therefore, if a vertex v has degree d'_v in H, then it must have at least d'_v potential neighbors.

Take any vertex v. If we see more than $10\frac{\log n}{p}$ potential neighbors of v throughout the construction of I, then w.h.p. $v \notin H$. This holds because every time that we see a potential neighbor of v, it joins the independent set independently with probability p.

Thus $\Pr[v \text{ has no neighbor in } I] \leq (1-p)^{100\frac{\log n}{p}} < \frac{1}{n^{100}}$, which means that H has max degree $O\left(\frac{\log n}{p}\right)$ w.h.p. $\square$

Following the above two Lemmas, we can prove Theorem 1 that the MPC-RandomGreedyMIS algorithm finds an MIS with high probability.

Consider the first iteration where we choose a vertex with a probability $\frac{1}{\sqrt{\Delta}}$. Then, suppose $\Delta \geq \log^{10} n$, applying Lemma 4 yields that the remaining graph has maximum degree $\sqrt{\Delta}\log n \leq \Delta^{\frac{1}{2}+\frac{1}{10}} \leq \Delta^{0.9}$. Therefore, each iteration will reduce the maximum degree to its power of 0.9. After $100\log\log\Delta$ steps, the process will terminate since the maximum degree will be $O(1)$.