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

Lecture 16

November 4th, 2020

Instructor: Soheil Behnezhad *Scribe: Dongyue Li*

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

1 Massive Parallel Computation Algorithms (Part II)

In the last lecture, we give the massive parallel computation (MPC) algorithms for maximum matching and minimum spanning trees. In this lecture, we present a improved MPC algorithm for maximum spanning tree and vertex coloring problems.

1.1 Recap

We first recap the definitions of massive parallel computation algorithms. Suppose there are M machines, each with a space S. The input of a size N is divided among the M machines arbitrarily. The computation happens in synchronized rounds. Within each round, each machine performas any computation on its local data. In addition, each machine can send messages to other machines, as long as the total messages (sent and received) by a single machines is less than S, where S is generally defined as $N^{2-\Omega(1)}$. M is set to be large enough to contain the input, i.e., $M = N/S$.

For graph problems, three space regimes are common:

- Super-linear regime: $S = O\left(n^{1+\epsilon}\right)$, where $\epsilon \in (0,1)$.
- Near-linear regime: $S = \tilde{O}(n)$.
- Sublinera space regime: $S = O(n^{\epsilon})$.

In the last lecture, we show the following results.

Theorem 1. *There is a randomized MPC algorithm that finds the maximal matching in $O(\frac{1}{\epsilon})$ rounds with space $O\left(n^{1+\epsilon}\right)$.*

Theorem 2. *There is a randomized MPC algorithm that finds the minimum spanning tree in $O(\frac{1}{\epsilon})$ rounds with space $O\left(n^{1+\epsilon}\right)$.*

Based on the assumption that all edge weights are distinct, we use the following Lemma to obtain the statement from Theorem 2.

Lemma 3. *Let $E' \subseteq E$. Then for any $e \in E' \setminus MST(E')$, we have $e \notin MST(E)$.*

Based on Lemma 3, one MPC algorithm is to find the minimum spanning tree (MST) by recursivly finding MSTs on subsets of edges.

```
MST(E):
    1. Partition the edge into subsets $E_1, \dots, E_l$ such that $|E_i| = \Theta\left(n^{1+\epsilon}\right)$.
    2. Send each $E_i$ to different machine.
    3. Return MST(MST($E_1$) $\cup \cdots \cup$ MST($E_l$)).
```

Running time. In every round, edges are pruned by a factor n^ϵ. After $O(\frac{1}{\epsilon})$ rounds, we are done.

1.2 Improved MPC Algorithm for Maximum Spanning Trees

In this lecture, we present an improved MPC algorithm that finds the MST in $O(\log \frac{1}{\epsilon})$ rounds.

```
MST(E):
    1. Randomly divide the vertices into $k$ subsets $V_1, \dots, V_k$.
    2. For any $i, j \in [k]$, send $G_{i,j} := G[V_i \cup V_j]$ to a different machine.
    3. Return MST$\left(\bigcup_{i \neq j \in [k]} \text{MST}(G_{i,j})\right)$.
```

First, we analyze the space complexity of Algorithm 1.2. We first analyze the number of edges in $G_{i,j}$. Each vertex v belongs to V_1 independently with probability $\frac{1}{k}$. Moreover, for any $v \in V_1$, $\mathbb{E}\left[\deg_{G_{1,2}}(v)\right] \leq \frac{\deg_G(v)}{k}$. By Chernoff bound, we have $\deg_{G_{i,j}}(v) \leq \tilde{O}\left(\frac{\deg_G(v)}{k}\right)$ with high probability.

Thus, in any $G_{i,j}$, the number of edges is

$$\sum_{v \in V} \Pr\left[v \in V_i\right] \cdot \deg_{G_{i,j}}(v) = \sum_{v \in V} \frac{1}{k} \cdot \frac{\deg_G(v)}{k} = O\left(\frac{m}{k^2}\right).$$

with high probability.

Therefore, to get $\frac{m}{k^2} \leq n^{1+\epsilon}$, it suffices to set $k = \sqrt{\frac{m}{n^{1+\epsilon}}}$. Suppose $m = n^{1+c}$, then $k = n^{\frac{c-\epsilon}{2}}$.

Theorem 4. *In the super-linear regime there is an MST algorithm that solves the problem in $O\left(\log \frac{1}{\epsilon}\right)$ rounds.*

Proof. For any $G_{i,j}$, we return $O\left(\frac{n}{k}\right)$ edges. Overall, we have $\binom{k}{2} \cdot O\left(\frac{n}{k}\right) = O\left(nk\right) = O\left(n^{1+c/2-\epsilon/2}\right)$ edges after recursion. After $O\left(\log \frac{1}{\epsilon}\right)$ rounds, we will have at most $O\left(n^{1+\epsilon}\right)$ edges. Therefore, Algorithm 1.2 solves the problem in $O\left(\log \frac{1}{\epsilon}\right)$ rounds. □

Corollary 5. *In the linear regime, there is an MST algorithm that takes $O\left(\log \log n\right)$ rounds.*

This results directly follows the runtime for super-linear regime. Since $n = n^{1+\frac{1}{\log n}}$, the MST algorithm in the linear regime takes $O(\log \varepsilon) = O(\log \log n)$ rounds.

1.3 MPC Algorithm for Vertex Coloring

Next, we present MPC algorithms for vertex coloring.

Definition 6. Given a graph $G = (V, E)$, a k-vertex coloring is an assignment C of $\{1, \dots, k\}$ to V such that $C(v) \neq C(u)$ if $(v, u) \in E$.

One observation is that any graph of max deg Δ, can be vertex colored with $\Delta + 1$ colors. Parameterized by Δ, this is the best bound.

Theorem 7 (AKC'19)**.** *There is an $O\left(n \log^2 n\right)$ space MPC algorithm that finds a $(\Delta+1)$-coloring in $O(1)$ rounds.*

Instead of proving Theorem 7, we prove a simplified Theorem that finds a $(1+\epsilon)\Delta$ -olorings for any constant $\epsilon > 0$:

Theorem 8. *There is an $\tilde{O}\left(\frac{n}{\epsilon}\right)$ space MPC algorithm that finds a $(1+\epsilon)\Delta$-colorings for any constant $\epsilon > 0$ in $O(1)$ rounds.*

The key idea behind the MPC algorithm to find vertex coloring is pallete sparsification. We define a pallete in the following.

Definition 9. For any vertex v, let $P(v)$ include any color $c \in \{1, \dots, (1+\epsilon)\Delta\}$ independently w.p. $\frac{\log n}{\epsilon\Delta}$.

We present a useful lemma in the following to prove Theorem .

Lemma 10. *There is a linear in space $(1+\epsilon)\,\Delta$-coloring C of G s.t. for any vertex v, $C(v) \in P(v)$ w.h.p.*

Next, we first show the proof of Theorem 1.3 by using Lemma 10.

Proof of Theorem 1.3. In one round, every vertex v samples colors for pallete $P(v)$ (where each color from $\{1, \dots, (1+\epsilon)\Delta\}$ has probability of $\frac{\log n}{\epsilon\Delta}$ of being included), and we communicate the colors to all machines.

For any edge (u, v) if $P(u) \cap P(v) \neq \emptyset$, we send (u, v) to machine 1 to form subgraph G' of G

Then find the coloring guaranteed by Lemma 10 in machine 1 on the subgraph $G'$of G sent to machine 1. $\square$

Claim 11. *G' has $O\left(\frac{n\log^2 n}{\epsilon^2}\right)$ edges w.h.p.*

Proof. Take a vertex v and condition on $P(v)$ and also that $P(v)$ has size $O\left(\frac{\log n}{\epsilon}\right)$ with high probability. For any neighbor u of v in G, we have

$$\begin{aligned}\Pr\left[P(u) \cap P(v) \neq \emptyset \mid P(v)\right] &\leq \sum_{c \in P(v)} \frac{10\log n}{\epsilon\Delta} \\ &= |P(v)|\,\frac{10\log n}{\epsilon\Delta} \\ &= O\left(\frac{\log n}{\epsilon}\right)\frac{10\log n}{\epsilon\Delta}.\end{aligned}$$

Therefore, $(u, v) \in G'$ with probability $O\left(\frac{\log^2 n}{\epsilon^2\Delta}\right)$. Then, we can have the following

$$\begin{aligned}\mathbb{E}\left[\deg_{G'}(v)\right] &= \deg_G(v) O\left(\frac{\log^2 n}{\epsilon^2\Delta}\right) \\ &\leq \Delta O\left(\frac{\log^2 n}{\epsilon^2\Delta}\right) \\ &= O\left(\frac{\log^2 n}{\epsilon^2}\right).\end{aligned}$$

By a Chernoff bound, we get $\deg_{G'}(v) = O\left(\frac{\log^2 n}{\epsilon^2}\right)$ w.p. $1 - \frac{1}{n^2}$.

Then, by a union bound over all v, we have that all vertices have degree $O\left(\frac{\log^2 n}{\epsilon^2}\right)$ in G' w.p. $1 - \frac{1}{n}$. Therefore, G' has at most $n \cdot \Delta(G') = O\left(\frac{n \log^2 n}{\epsilon^2}\right)$ edges. $\square$

Finally, we show the proof of Lemma 10.

Proof of Lemma 10. Consider the process that goes over the vertices one by one in an arbitrary order. Upon visiting vertex v, we reveal the colors of v in $P(v)$ one by one. The first color that is not assigned to any neighbor of v is assigned to v. We claim that this finds a valid coloring w.h.p.

Since we have $(1+\epsilon)\Delta$ colors, for any vertex v, there are at least $\epsilon\Delta$ colors not used by any neighbor so far. Every time that we reveal the next color of $P(v)$, we hit one of these $\epsilon\Delta$ colors w.p. $\geq \frac{\epsilon\Delta}{(1+\epsilon)\Delta} \geq \epsilon/2$.

Therefore, the probability that we do not sample any "good" color after $\frac{10 \log n}{\epsilon}$ trials is $(1-\epsilon/2)^{\frac{10 \log n}{\epsilon}} \leq e^{-5 \log n} \leq \frac{1}{n^5}$.

By a union bound over n vertices, any vertex has a "good" color in its pallete w.p. $1 - 1/n^4$. $\square$