COMS E6998-9: Algorithms for Massive Data (Fall'23) Nov 29, 2023

Lecture Lecture 23: MPC for Estimating MST (cont'd)

Instructor: *Alex Andoni* Scribes: *Ruowang Zhang*

1 Recap: Algorithm for computing MST of Geometric Graphs

1.1 Algorithm Overview

Recall in the previous class, we described an algorithm for the problem of computing MST of $P \subset \mathbb{R}^d$:

1. Assume n distinct points in an integer grid of size $[n^2] \times [n^2]$.
2. Split the space into a randomly-shifted quad-tree with partition π, which has cells by size $\sigma \times \sigma$ (or more precisely $(\sigma + 1) \times (\sigma + 1)$ with spilled margin).
3. Set $\sigma = s^{1/4}$, and define $L = O(\log_s n)$ where L is the number of levels for the algorithm runs.

The high level algorithm is to run $cell - algo$ on each cell where $cell - algo$ is:

1. From the lowest level.
2. Run the Kruskal algorithm on the cell of V nodes until the edge length $> \varepsilon\triangle$ where $\triangle$ is the current cell's side length.
3. Represent V' as a $\varepsilon\triangle$-net of V.
4. Return V' with connectivity info, i.e. the representative set of V.
5. Propagate to the upper next level.

Theorem 1. *The algorithm runs in* $O(\log_S n)^{O(1)}$ *p-time, if* $s > (\frac{1}{\varepsilon})^{O(d)}$*, assuming* $d = 2$*.* s *is the space of the machine.*

1.2 Correctness of the algorithm

Definition 2. $\rho(u, v)$ *as the original distance between* u *and* v *in the graph.*
$\rho_\pi(u, v) = \rho(N_{l-1}(u), N_{l-1}(v)) + 2\varepsilon\triangle_{l-1}$ *where* l *is the level* u *and* v *belong to in the same cell,* $N_l(u)$ *as the representative set of* u *in level* l*, and* $\triangle_l$ *is the side length at level* l*.*

Fact 3. $\rho_\pi(u, v) \leq \rho(u, v) + 4\varepsilon\triangle_l$

We proved the following result in the previous lecture:

Lemma 4. $E_\pi[\rho_\pi(u, v)] \leq (1 + 8\sqrt{2}\varepsilon L)\rho(u, v)$

2 Today: Remain aspects of the algorithm

2.1 Conclude the correctness proof

We want to prove the following result:

Lemma 5. *Output of algo $\hat{MST}$ satisfies $E_\pi[\hat{MST}] \leq (1+8\sqrt{2}\varepsilon L)MST_\rho$ where MST_ρ is the optimal MST under distiance function ρ.*

First, we claim:

Claim 6. *Our algorithm $\equiv$ run Kruskal's algorithm on $\rho_\pi \equiv MST_{\rho_\pi}$, and $E_\pi[\hat{MST}] \leq E_\pi[MST_{\rho_\pi}] = E_\pi[\min_T \rho_\pi(T)] \leq E_\pi[\rho_\pi(T^*)]$ where T is one MST and T^* is the optimal MST_ρ.*
Then we can apply Lemma 4 and obtain $E_\pi[\rho_\pi(T^*) \leq (1+8\sqrt{2}\varepsilon L)MST_\rho$

Proof. The Proof for Claim 6 is by induction. On level l, we define $IH(l)$ = when done with level l, chosen edge $\equiv$ Kruskal up to $\varepsilon\triangle_l$ cost w.r.t. ρ_π.
Base Step:
When $l=0$, $\varepsilon\triangle_l < 1$ since cell side length is 1.
Inductive Step:
Assume $IH(l)$, looking at $l+1$:

Observation 7. *$\forall u, v$ in different cells at $l+1$, $\rho_\pi(u,v) \geq 2\varepsilon\triangle_{l+1} > \varepsilon\triangle_{l+1}$ by definition 2, so we can analyze the cells at $l+1$ separately.*

Observation 8. *$\forall u, v$ in the same cells at $l+1$ with size $\triangle \times \triangle$. If u, v are not inputs to $cell-algo$ (i.e. not coming from representive set N_l), $\rho_\pi(u,v) = \rho(N_l(u), N_l(v)) + 2\varepsilon\triangle_l$ which is bigger than the distance between their representatives.*

which concludes the proof. □

2.2 Implementation

1. A stage for each level of the quad-tree

2. In each level, the total input to $cell-algo$ s at level $l \leq O(n)$

3. Each cell's input size is $\sigma^2 O(\#representatives)$ where $O(\#representatives) \leq \frac{4}{\varepsilon^2}$. The input size is therefore $\leq \sqrt{S} * O(1/\varepsilon^2) < S/2$, since $\sqrt{S} > O(1/\varepsilon^2)$

4. Can distribute work to all machines, keeping input size $< S$.

   One way to arrange the jobs/cells per level as $D1, D2, \ldots, D_k$, then for the non empty ones, we can pack them in order and assign to machines once the packed size s' once $s' \in [S/2, S]$.

2.3 Remark

To use the algorithm for problems where the input grid is a $[n^2] \times [n^2]$ integer grid, we should reduce those problems to it. Here is the algorithm:

1. Given n points in $\mathbb{R}^2$. Compute $D = \max\{\max\{x_i | \forall i \in [n]\}, \max\{y_i | \forall i \in [n]\}\}$.

   Observe that $cost(MST) \in [D, 2nD]$, so it's ok to ignore distance less than $\frac{\varepsilon}{n}D$.

2. Round all points to integer multiples of $\frac{\varepsilon}{n}D$.

3. If a point is repeated, then connect together, leaving just one copy.

4. Divide all coordinates by $\frac{\varepsilon}{n}D$. For each x or y, subtract min-value, which gives all coordinates $\in [0, \frac{n}{\varepsilon}]$

3 Intro: Distributed Algorithms (Models)

3.1 CONGEST model

- Given an undirected graph G.
- Each node = a computation unit
- Communication is done on edges of G only
- In each round, a node can send $O(\log n)$ bits message to each neighbor.

We would like to solve a problem on G, e.g. MST, max flow/min cut, shortest path, etc.
Output: each node should have the "relevant" part of the output. E.g. for MST, each node knows which incident edges belong to the MST.
Time: minimum # of rounds.
For the MST problem, runtime is $\Omega(D)$ where D is the diameter / # of hops, or $\Omega(D + \sqrt{n})$. We will show an algorithm that runs in $O((D + \sqrt{n}) \log n)$.