Sublinear Time and Space Algorithms 2016B – Lecture 6
Triangle Counting, Geometric Streams and Coresets*

Robert Krauthgamer

1 Triangle Counting

Goal: Report the number of triangles in G. We will denote it by T.

Motivation: The relative frequency of how often 2 friends of a person know each other is defined as

$$F = \frac{3T}{\sum_{v \in V} \binom{\deg(v)}{2}}.$$

We can compute $\sum_{v \in V} \binom{t}{2}$ in $O(n)$ space, by maintaining the degree of every vertex.

Distinguishing $T = 0$ from $T = 1$ is known to require $\Omega(m)$ space [Braverman, Ostrovsky, and Vilenchik, 2013].

We will henceforth assume a known lower bound $0 < t \leq T$.

First Approach [Bar-Yossef, Kumar and Sivakumar, 2002]:

Define vector $x \in R^{\binom{n}{3}}$, where every coordinate x_S for a subset $S \subset V$ of 3 vertices, counts the number of edges internal to S.

T is the number of coordinates in x that have value 3.

We will use the frequency moments $F_p = \|x\|_p^p$ for $p = 0, 1, 2$.

Lemma: $T = F_0 - 1.5F_1 + 0.5F_2$.

Proof: If $x_S \in 0, 1, 2$ then its contribution to RHS is 0. If $x_S = 3$ then its contribution to RHS is $1 - 4.5 + 4.5 = 1$.

Why such formula exists?: We are looking for a function $f(x_S) : \mathbb{R} \to \mathbb{R}$ with specific value on $0, 1, 2, 3$. We can do polynomial interpolation. It would generally require degree 3, but $F_0 = \mathbb{1}_{\{x_S > 0\}}$ gives an extra degree of freedom.

Algorithm 1:

*These notes summarize the material covered in class, usually skipping proofs, details, examples and so forth, and possibly adding some remarks, or pointers. The exercises are for self-practice and need not be handed in. In the interest of brevity, most references and credits were omitted.

Maintain the frequency moments $p = 0, 1, 2$ of vector $x \in \mathbb{R}^{\binom{n}{3}}$. Initially $x = 0$, and when an edge (u, v) arrives, increment x_S for every S of the form $\{u, v, w\}$.

Correctness: As was seen in class, we can compute frequency estimates $\hat{F}_P \in (1 \pm \gamma)F_p$, and if we set $\gamma = O(\frac{t}{\varepsilon mn})$, we would get additive error $\varepsilon t \leq \varepsilon T$.

Storage: The storage requirement is $O(\gamma^{-2} \log n) = O(\varepsilon^{-2}(\frac{mn}{t})^2 \log n)$ words.

Algorithm 2 [Buriol, Frahling, Leonardi, Marchetti-Spaccamela, and Sohler, 2006; Ahn, Guha and McGregor, 2012]:

Notation: $N = \binom{n}{3}$.

1. Init: pick k random subsets $S_1, \ldots, S_k$ each of size 3

2. Update: maintain each x_{S_i} (explicitly)

3. Output: let c be the number of sets S_i for which $x_{S_i} = 3$, and report $\tilde{T} = c \cdot N/k$

Analysis: As was seen in class, we can get additive error $\varepsilon t \leq \varepsilon T$ by choosing $k = O(\frac{TN}{\varepsilon^2 t^2}) = O(\frac{n^3}{\varepsilon^2 t})$,

The storage requirement is of course $O(k)$ words.

Algorithm 2+:

Improve the previous algorithm by choosing sets S_i more selectively. Specifically, pick only sets S for which $x_S \geq 1$. The size of this set is $N' \leq mn$, which is often much smaller than n^3.

Note that the estimator needs to know (a good estimate of) $N' = \|x\|_0$.

The storage requirement would be $k' = O(\frac{mn}{\varepsilon^2 t})$ words.

Exer: Show how to implement this sampling in the streaming model.

Hint: ℓ_0-sampling (one that returns a coordinate and its value)

Exer: Which of the algorithms above can be implemented in the presence of edge deletions (dynamic graphs)?

2 Geometric Streams and Coresets

Geometric stream: The input is a stream of points in $\mathbb{R}^d$ denoted $P = \langle p_1, \ldots, p_n \rangle$.

Problem definition: The goal is to minimize some cost function $C_P : \mathbb{R}^d \to \mathbb{R}$, where $C_P(x)$ represents the cost of using x as a solution ("center") for input P.

For example, in the *Minimum Enclosing Ball (MEB)* the goal is to find a ball of minimum radius that contains P. This problem is captured by the cost function

$$C_P^{MEB}(x) = \max_{p \in P} \|p - x\|_2.$$

Other clustering problems where a similar approach may work: enclosing the points in a box (axis-

parallel or not) or in a slab (between two parallel hyperplanes), or in a cylinder (the center x is replaced by a line).

Definition: We say that such a cost function C is monotone if

$$\forall x \in \mathbb{R}^d, Q \subset P, \qquad C_Q(x) \leq C_P(x).$$

Definition [Agarwal, Har-Peled, and Varadarajan, 2004]: Given a monotone C, we say that $Q \subset P$ is an *α-coreset* for P if

$$\forall x \in \mathbb{R}^d, T \subset \mathbb{R}^d, \qquad C_{Q \cup T}(x) \leq C_{P \cup T}(x) \leq \alpha \cdot C_{Q \cup T}(x).$$

The idea is that by storing the small subset Q we can approximate the optimum for P within factor α, even if more points will be added later.

Plan: We will show that MEB admits a small coreset, and that small coresets (with certain properties) yield low-storage streaming algorithms.

Theorem 1: For every $d \geq 2$ and $\varepsilon \in (0, 1/2)$, the cost function C_P^{MEB} has a $(1+\varepsilon)$-coreset of size $O(1/\varepsilon^{(d-1)/2})$.

Merge Property: If Q is an α-coreset of P, and Q' is an α'-coreset of P', then $Q \cup Q'$ is an $(\alpha \cdot \alpha')$-coreset of $P \cup P'$.

Reduce Property: If Q is an α-coreset of P, and R is a β-coreset of Q, then R is an $(\alpha\beta)$-coreset of P.

Disjoint Union Property ("strong" version of merge): If Q is an α-coreset of P, and Q' is an α'-coreset of P', then $Q \cup Q'$ is an $\max\{\alpha, \alpha'\}$-coreset of $P \cup P'$.

Lemma: Every monotone coreset satisfies the Merge and Reduce properties. C_P^{MEB} satisfies also the Disjoint Union property.

Exer: Prove this lemma.

Theorem 2: Suppose the cost function C is monotone, that it admits $(1+\varepsilon')$-coresets of size $f(\varepsilon')$ for every $\varepsilon' \in (0, 1/2)$, and that these coresets have the Disjoint Union property. Then there is a streaming algorithm for minimizing C_P, that achieves $1 + O(\varepsilon)$ approximation using $O(f(\varepsilon/\log n) \cdot \log n)$ words of space.

Remark: We (implicitly) assume that when $|P| \leq 2f(\varepsilon')$ (small inputs), (i) a coreset as above can be computed using space $O(f(\varepsilon'))$, and (ii) a solution x that minimizes $C_P(x)$ can be computed.

Proof of Theorem 2: The algorithm uses the "merge and reduce" approach. We will first describe it as a non-streaming algorithm, based on a hierarchical partitioning of the stream.

Suppose the stream is partitioned into "blocks" of size B, which is a "buffer" size to be chosen later, and let $\varepsilon' = \varepsilon/\log n$. Now build a binary tree on these blocks in the natural order. Specifically, at level 0 (the n/B leaves of the tree), each node i gets as input the i-th block and outputs it without processing. At level $h = 1, \ldots, \log_2(n/B)$, the input for each node is the concatenation of its two children's outputs Q and Q'. The node then computes a $(1+\varepsilon')$-coreset R for $Q \cup Q'$, and then outputs this R.

At the top level h, the algorithm further computes for the final R an optimal $\tilde{x} \in \mathbb{R}^d$ and outputs this $\tilde{x}$.

The output of each node at level $h \geq 1$ is a subset of size $f(\varepsilon')$, and this bound extends also to level $h = 0$ by setting $B = f(\varepsilon')$.

Correctness: We prove by induction that the output of every node at level h is a $(1+\varepsilon')^h$-coreset of the points fed into its descendant leaves. Indeed, consider a node at level h. Suppose it receives from its children two sets Q and Q' that are $(1+\varepsilon')^{h-1}$-coresets of the respective original points P and P'. Then by the Disjoint Union property, $Q \cup Q'$ is a $(1+\varepsilon')^{h-1}$-coreset of $P \cup P'$. By the Reduce property, this node's output R is a $(1+\varepsilon')^h$-coreset of $P \cup P'$.

The output $\tilde{x}$ is optimal for the final $(1+\varepsilon')^h$-coreset R, and thus achieves approximation factor $(1+\varepsilon')^h \leq e^{\varepsilon' h} \leq e^{\varepsilon} \leq 1 + 2\varepsilon$.

Streaming Implementation: We will run $\log_2(n/B)$ algorithms in parallel, one for each level of the tree. The algorithm at each level $h \geq 1$ reads a virtual stream produced by the algorithm of level $h-1$, and produces a virtual stream for level $h+1$. It uses a buffer of size $2B$ to store the inputs P and P' from the "next" two children. When these arrive, it computes a new coreset R and outputs this R, and now the buffer is emptied and the process starts again.

The total storage requirement (for all levels) is $O(B \log(n/B)) = O(f(\varepsilon/\log n) \cdot \log n)$ words of space.

QED.

Corollary 3: The Minimum Enclosing Ball has a streaming algorithm that achieves $(1+\varepsilon)$-approximation with storage requirement $O(\frac{\log^{(d+1)/2} n}{\varepsilon^{(d-1)/2}})$.

Remark: In the particular case of MEB that we will see below, the coreset can actually be easily computed in a streaming fashion, yielding a streaming algorithm with storage $O(f(\varepsilon)) = O(\varepsilon^{(d-1)/2})$.

3 Coreset for Minimum Enclosing Ball

Grids in $\mathbb{R}^d$: For non-zero vectors $u, v \in \mathbb{R}^d$ define $\text{angle}(u,v) = \arccos \frac{\langle u,v \rangle}{\|u\|_2 \|v\|_2}$.

We say that $U \subset \mathbb{R}^d \setminus \{0\}$ is a θ-grid (or δ-cover) if

$$\forall x \in \mathbb{R}^d,\ \exists u \in U, \quad 0 \leq \text{angle}(x,u) \leq \theta.$$

We will need the following theorem (without proof).

Theorem 4: For every $\delta > 0$ there exists a δ-grid U of size $O(1/\theta^{d-1})$. In fact, we may assume it consists of unit-length vectors.

Proof of Theorem 1: Fix a θ-grid U for $\theta = \sqrt{\varepsilon}$. Given P, define

$$Q = \bigcup_{u \in U} \{\operatorname*{argmax}_{p \in P} \langle p, u \rangle\}.$$

That is, Q stores for each direction $u \in U$ the "extreme" point in this direction (as measured by projection on u).

To prove that Q is a $(1+\theta^2)$-coreset, consider $T \subset \mathbb{R}^d$ and $x \in \mathbb{R}^d$. Let $z \in \mathbb{R}^d$ be the farthest point from x in $P \cup T$, then $C_{P \cup T}(x) = \|z - x\|_2$.

We now have two cases. If $z \in T$, then clearly $\|z - x\|_2 \leq C_{Q \cup T}(x)$.

Otherwise (i.e., $z \in P$), there is $u \in U$ such that $0 \leq \text{angle}(z - x, u) \leq \theta$. Let $q \in P$ be the point that maximizes $\langle q, u \rangle$. Then $q \in Q$, and we get that

$$C_{Q \cup T}(x) \geq \|q - x\|_2.$$

Since $z \in P$ is a candidate for this maximization, $\langle q, u \rangle \geq \langle z, u \rangle$, and we get (recall u has unit length)

$$\|q - x\|_2 \geq \langle q - x, u \rangle \geq \langle z - x, u \rangle \geq \cos\theta \cdot \|z - x\|_2.$$

A more geometric way to see the last inequality: let z' be the projection of z on the line $\{x + \gamma u : \gamma \in \mathbb{R}\}$, and let q' be the projection of q on the same line. Since $z \in P$ is a candidate for the maximization (projection on the line),

$$\|q - x\|_2 \geq \|q' - x\|_2 \geq \|z' - x\|_2 \geq \cos\theta \cdot \|z - x\|_2,$$

where the last inequality follow from the angle $\text{angle}(u, z - x) \leq \theta$ in the triangle x, z, z'.

To complete the proof, recall that $\|z - x\|_2 = C_{P \cup T}(x)$ and use $\cos\theta \geq 1 - \theta^2/2 \geq \frac{1}{1+\theta^2}$, hence $C_{Q \cup T}(x) \geq \frac{1}{1+\varepsilon} C_{P \cup T}(x)$.

Finally, use Theorem 4 to bound the size of the coreset

$$|Q| \leq |U| = O(1/\varepsilon^{(d-1)/2}).$$