Subgraph Counting in Data Streams
COMS 6998: Algorithms for Massive Data Final Project

Krish Singal and Yiming Fang

Fall 2023

Abstract

Subgraph counting is a fundamental problem in graph algorithms with direct applications to network analysis [**BHLP11**, **EM02**]. Most progress in recent years has been seen for simpler subgraphs such as triangles and four-cycles. This work has three main contributions. (1) We survey existing constant pass streaming algorithms on triangle counting [**MSV16**, **JK21**] and some corresponding lower bounds in the one-pass setting [**BOV13**, **KP17**]. (2) We survey an existing four-cycle counting algorithm [**V20**]. (3) Finally, we present our results – a novel 3-pass $(1 \pm \epsilon)$-approximation algorithm for diamond counting in the stream with expected space complexity $O(T^{1/4} + \epsilon^{-2} m T^{-3/4})$.

Contents

1 Introduction

Subgraph counting is of particular interest in the streaming model and has been extensively studied in recent years. For a fixed subgraph H, the H-subgraph counting problem asks for the total number of instances of H (not necessarily vertex-induced) in input graph $G = (V, E)$. The problem has received much attention in standard settings and has recently been of interest in the streaming model of computation. In this regime, we aim to achieve expected space complexity that is sublinear in the size of the graph. For this reason, we aim to *approximate* subgraph counts up to a $(1 \pm \epsilon)$ factor.

1.1 Preliminaries

This survey describes results within the context of the *insertion-only* graph streaming model, wherein a graph $G = (V, E)$ is received as a stream of edges $\sigma_{i=1}^m$ in arbitrary order. In particular, this means that the ordering can be *non-adaptively* adversarially chosen. Our discussion will also focus on *constant-pass* algorithms, which are allowed to replay the stream only a constant number of times.

A general strategy used by [**MSV16**], [**MV20**] and [**V20**] is to construct an estimator by sampling edges with fixed probability p and returning the number of surviving subgraphs scaled by p^k (where k denotes the size of the subgraph of interest). This is trivially an unbiased estimator and can be shown to give accurate estimates with high probability. Similarly, one can sample vertices with fixed probability q and store all its incident edges. Crucially, the space complexity of such algorithms is directly proportional to the number of edges we sample and store. We will see that many of the algorithms covered in this survey follow similar guiding ideas.

1.2 Previous Results

Recent work has brought significant advances to the problem under various different parameterizations. Here, we briefly outline the state of the art.

Problem	$\mathbb{E}$[**Space Complexity**]	**# Passes**	**Ref**
$\triangle$ Counting	$O(\epsilon^{-2}mn/T)$	1	Lecture 3
	$\widetilde{O}(\epsilon^{-2}m^{3/2}/T)$	3	[**MSV16**]
	$\widetilde{O}(\epsilon^{-2}m/\sqrt{T})$	2	[**MSV16**]
	$\widetilde{O}(\epsilon^{-O(1)}m(\Delta_E + \sqrt{\Delta_V})/T)$	1	[**JK21**]
$\square$ Counting	$\widetilde{O}(\epsilon^{-2}m/T^{1/4})$	3	[**MV20**]
	$\widetilde{O}(\epsilon^{-2}m/T^{1/3})$	3	[**V20**]

Table 1: Triangle and 4-cycle counting in arbitrary-order streams

Where T denotes the true number of subgraphs H in G, Δ_V denotes the maximum number of triangles sharing any given vertex, and Δ_E denotes the maximum number of triangles sharing any given edge.

Recent work by [**CEI+22**] also extends this problem to the learning-augmented setting where the algorithm has access to a prediction oracle. The challenge there is to ensure robust algorithmic guarantees even when the prediction oracle is incorrect.

In this work, we survey the 2-pass and 1-pass triangle counting algorithms from [**MSV16**] and [**JK21**] and 3-pass 4-cycle counting algorithm from [**V20**]. We also prove the following theorem about our novel algorithm regarding diamond counting:

Theorem 1 There exists a 3-pass algorithm that returns a $(1 \pm \epsilon)$ multiplicative approximation to the number of diamonds in a streamed graph with space complexity

$$\widetilde{O}\left(T^{1/4} + \epsilon^{-2}m/T^{3/4}\right)$$

and success probability 0.99. Here, T denotes the number of triangles in the graph.

2 Triangle Counting

Recent research has brought a number of algorithms for triangle counting in the stream using different number of passes. We will survey two state-of-the-art algorithms.

2.1 A 2-pass Algorithm [MSV16]

Algorithm 1 2-pass Triangle Counting with $\widetilde{O}(\epsilon^{-2}m/\sqrt{T})$ Space

```
1: First Pass:
2: Let $p = O(\epsilon^{-2}\log n/\sqrt{T})$ be the sampling rate
3: Let $Z$ be a random subset of nodes constructed by including each node w.p. $p$
4: Let $S_1$ be a random subset of edges constructed by including each edge w.p. $p$
5: Let $S_2$ be the set of edges incident to nodes in $Z$
6:
7: Second Pass: on $e = (u, v)$
8: Let $S_i^L = \{e \in S_i \mid \text{oracle}(e) = L\}, \forall i \in [2]$
9: Let $S_i^H = \{e \in S_i \mid \text{oracle}(e) = H\}, \forall i \in [2]$
10: Let $\text{oracle}(e) = \begin{cases} H & \text{if } \widetilde{x}_e \geq p\sqrt{T} \\ L & \text{if } \widetilde{x}_e < p\sqrt{T} \end{cases}$, where $\widetilde{x}_e = |\{w \in Z : u, v \in \Gamma(w)\}|$
11: if $\text{oracle}(e) = L$ then
12:     $A_L = A_L + \frac{1}{3}|\{w \mid \{u,w\},\{v,w\} \in S_1^L\}|$
13: if $\text{oracle}(e) = H$ then
14:     $A_H = A_H + \widetilde{x}_e^1 + \widetilde{x}_e^2/2 + \widetilde{x}_e^3/3$, where
15:     $\widetilde{x}_e^1 = |\{z \in Z : (u,z),(v,z) \in S_2^L\}|$
16:     $\widetilde{x}_e^2 = |\{z \in Z : (u,z) \in S_2^L, (v,z) \in S_2^H\}| + |\{z \in Z : (u,z) \in S_2^H, (v,z) \in S_2^L\}|$
17:     $\widetilde{x}_e^3 = |\{z \in Z : (u,z),(v,z) \in S_2^H\}|$
18:
19: return $\hat{T} = A_L/p^2 + A_H/p$
```

Main idea: From a high level, the algorithm spends the first pass constructing a heavy edge oracle and samples edges with fixed probability. In the second pass, the algorithms counts the number of triangles formed by each heavy and light edges with the samples (with appropriate scaling). The key insight behind the algorithm is that the sampling scheme's space requirement is highly sensitive to the existence of heavy edges, so classifying edges based on heaviness helps reduce variance.

Definition 1 (Heavy edges): For some constant $c \geq 1$, an edge is considered $\begin{cases} \text{heavy} & \text{if } x_e \geq c\sqrt{T} \\ \text{light} & \text{if } x_e < \frac{1}{c}\sqrt{T} \end{cases}$ where $e = (u, v)$ and $x_e = |\{w \in V(G) : u, v \in \Gamma(w)\}|$ is the number of subgraphs ($\triangle$) that e is part of.

Lemma 1 *For all edges $e \in E(G)$, with $1 - 1/poly(n)$ probability:* $\begin{cases} \textit{oracle}(e) = L & \implies e \textit{ is light} \\ \textit{oracle}(e) = H & \implies e \textit{ is heavy} \end{cases}$

Proof. Since each vertex $v \in V(G)$ is sampled into Z independently with probability p, and let $\widetilde{x}_e = |\{w \in Z : u, v \in \Gamma(w)\}|$. Then we can apply Chernoff bound. For any given e, if e is heavy, then $x_e \geq c\sqrt{T}$, and

$$\Pr[\text{oracle}(e) = L] = \Pr\Big[\widetilde{x}_e < p\sqrt{T}\Big] \leq e^{-O(p\sqrt{T})} = e^{-O(\epsilon^{-2}\log n)} = 1/\text{poly}(n)$$

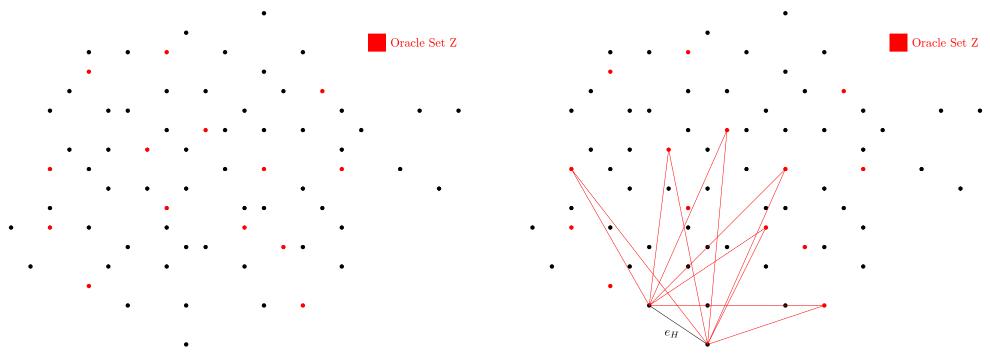

Figure 1: (a) Construction of oracle by sampling into set Z with fixed probability p (b) Heavy edge e_H that forms many triangles with vertices in Z

.

The other side can be proved similarly. By a union bound over all $e \in E(G)$, the oracle implies the correct result with probability $1 - 1/\text{poly}(n)$. □

Theorem 2 [**MSV16**] Alg. 1 returns $\hat{T} \in (1 \pm \epsilon)T$ using $\widetilde{O}(\epsilon^{-2}m/\sqrt{T})$ space, w.p. $\geq 90\%$.

Proof. Using the property of the oracle in Lemma 1, we can consider the following cases separately.

- Consider light triangles: Let T_L denote the number of triangles formed with only edges from the set $E_L = \{e \in E : \text{oracle}(e) = L\}$. Since each light triangle contains 3 light wedges, one can estimate T_L by $\frac{1}{3}A$, where $A = \sum_{i \in S} A_i$ is the number of light wedges in the stream S. To estimate A, the algorithm counts all light wedges sampled in S_1^L, and scales the quantity by $1/p^2$. Therefore, we can estimate T^L with A^L/p^2 and we analyze the estimator's expectation and variance. For each light wedge in the stream, the probability that both edges are sampled is p^2, so $\mathbb{E}[A_i] = p^2$ and

$$\mathbb{E}[A^L] = \frac{1}{3p^2} \sum_{i \in S} \mathbb{E}[A_i] = \frac{1}{3p^2} \sum_{i \in S} p^2 = \frac{1}{3} A = T$$

  Furthermore, $\mathbb{V}[A_i] \leq p^2$ and $\mathbb{COV}[A_i, A_j] \leq \mathbb{E}[A_i A_j] = p^3$ if the two wedges W_i, W_j share an edge, and $\mathbb{COV}[A_i, A_j] = 0$ otherwise. So

$$\mathbb{V}[A] = \sum_{i \in T} \mathbb{V}[A_i] + \sum_{i \neq j} \mathbb{COV}[A_i, A_j] \leq Tp^2 + \sum_{e \in E_L} \sum_{W_i \cap W_j = \{e\}} p^3 \leq Tp^2 + O(\sqrt{T})p^3$$

  where the last step follows from lightness of the edges. Applying Chebyshev's Inequality and union bounding over the failure probability of the oracle gives us $A_L/p^2 \in T_L \pm \epsilon T/2$ w.h.p.

- Consider heavy triangles: Let T^H denote the number of triangles with at least one edge from the set $E^H = \{e \in E : \text{oracle}(e) = H\}$. For each heavy e, let x_e^i denote the number of triangles that includes e and exactly i heavy edges. Since each triangle with i heavy edges appears in i different terms, and each edge is sampled independently with probability p, we have

$$T^H = \sum_{e:\text{oracle}(e)=H} \left(x_e^1 + x_e^2/2 + x_e^3/3\right) = (1 \pm \epsilon) A_H/p$$

  with probability $\geq 1 - 2e^{-O(\epsilon^{-2}p\sqrt{T})} = 1 - 1/\text{poly}(n)$ by applying the Chernoff bound.

Union bounding over failure probability in both cases gives us the desired success probability. The expected space complexity is $\mathbb{E}[|S_1| + |S_2|] = pm + \sum_v p\deg(v) = 3pm = \widetilde{O}(\epsilon^{-2}m/\sqrt{T})$ as desired. □

2.2 Lower Bounds on One-Pass Algorithms [BOV13, KP17]

A series of work on approximate triangle counting culminated in a $\widetilde{O}(\frac{md}{T})$ space, one-pass algorithm by [**PTTW13**]. While substantial progress was made, the line of work hit a fundamental barrier after $\Omega(m)$ space lower bounds were shown for distinguishing between two distributions over triangle-free graphs and graphs with $\Omega(m)$ triangles. We discuss two such constructions here by [**BOV13**] and [**KP17**] which motivate a new parameterization and the one-pass algorithm of [**JK21**].

[**BOV13**]'s lower bound is derived via a reduction from the randomized one-way communication complexity of the INDEX$_n$ problem. Here, Alice is given a bit-string $\mathbf{x}$ of length n and Bob is given an index $\ell \in [n]$. Alice must communicate to Bob and Bob must output $\mathbf{x}_\ell$ with probability $\geq 2/3$. In particular, [**CCKM10**] shows an $\Omega(n)$ lower bound on the communication complexity of INDEX$_n$.

Assume, for the sake of contradiction, that there exists an $o(m)$ space algorithm $\mathcal{A}$ that computes a $(1 \pm \epsilon)$ approximate triangle count with constant success probability. It is easy to see that $\mathcal{A}$ can then also distinguish between two distributions on graphs, $\mathcal{G}_1$ that has positive mass on triangle-free graphs and $\mathcal{G}_2$ that has positive mass on a graphs with $\Omega(T)$ triangles, with constant success probability. We then construct a one-way communication protocol for INDEX$_n$ using $o(n)$ bits and derive a contradiction.

1. Alice first constructs G by creating $2n$ vertices $\{u_i\}_{i=1}^{2n}$ and creating an edge (u_{2i}, u_{2i+1}) iff $\mathbf{x}_i = 1$
2. Alice runs $\mathcal{A}$ on G and communicates $\mathcal{A}$'s memory to Bob
3. Bob then adds n vertices $\{v_j\}_{j=1}^{n}$ to G and connects $u_{2\ell}$ and $u_{2\ell+1}$ to each of $\{v_j\}_{j=1}^{n}$
4. Bob then returns the answer of $\mathcal{A}$ on G

Notice that G has exactly n triangles if $\mathbf{x}_\ell = 1$ and 0 otherwise. Therefore, Bob is correct with the same constant success probability as $\mathcal{A}$. Furthermore, $\mathcal{A}$ uses $o(n)$ space by assumption, so the above protocol contradicts the $\Omega(n)$ lower-bound of [**CCKM10**]. Because our constructed graph has $O(n)$ edges, we conclude an $\Omega(m)$ space complexity lower bound to distinguish between triangle-free graphs and those with $\Omega(m)$ triangles in this family of graphs. However, [**BOV13**]'s above construction of a "hard-instance" family of graphs is more of a special case, where all triangles share an edge. This motivated a new parameter, Δ_E, that counts the maximum number of triangles sharing an edge. Reparameterizing using Δ_E, we see that [**BOV13**]'s construction implies an $\Omega(\frac{m\Delta_E}{T})$ space lower bound.

The lower bound by [**KP17**] is similarly derived via a reduction from the randomized one-way communication complexity of the Boolean Hidden Matching problem [**GKK+07**]. We skip the details of the proof here for the sake of brevity, but we note that the construction involves a distribution of graphs with $\Omega(m)$ triangles sharing a single vertex, similar to [**BOV13**]'s construction in which every triangle shares an edge. This motivated a new parameter, Δ_V, that counts the maximum number of triangles sharing a vertex. Reparameterizing using Δ_V, it turns out that [**KP17**]'s construction implies an $\Omega(\frac{m\sqrt{\Delta_V}}{T})$ space lower bound.

Because one does not strictly dominate the other asymptotically, [**BOV13**, **KP17**] give a combined lower bound of $\Omega(\frac{m}{T}(\Delta_E + \sqrt{\Delta_V}))$.

2.3 Optimal One-pass Algorithm [JK21]

The lower bounds from section 2.2 were met with a matching upper bound by [**JK21**]. It is important to note that access to parameters T, Δ_E, Δ_V makes computation redundant. The algorithm must allocate sufficient space prior to receiving any information regarding graph G, so we assume that the algorithm has access to constant factor approximations of the quantities in question. Under these new parameterizations, [**JK21**] designs an optimal one-pass algorithm with space complexity $O\left(\frac{m}{T}\left(\Delta_E + \sqrt{\Delta_V}\right)\log n \frac{\log\frac{1}{\delta}}{\epsilon^2}\right)$ that matches the lower bounds of [**KP17**] and [**BOV13**] up to poly-logarithmic factors.

We briefly outline the algorithm and its analysis. The key insight of [**JK21**] is to sample vertices with fixed probability p and edges with fixed probability q. We choose to store edges *adaptively* in $\widehat{E}$ and increment our triangle estimator when an edge update uw closes an existing *wedge* $\{uv, vw\}$ stored in $\widehat{E}$.

Algorithm 2 One-pass Triangle Counting with $O\left(\frac{m}{T}\left(\Delta_E + \sqrt{\Delta_V}\right)\log n \frac{\log\frac{1}{\delta}}{\epsilon^2}\right)$ Space

```
1: $\widehat{E} \leftarrow \emptyset$, $\widehat{T} \leftarrow 0$
2: for edge $uw$ in stream do
3:     for $v \in V$ do
4:         if $\mathcal{H}_V(v) = 1$ and $vu, vw \in \widehat{E}$ then
5:             $\widehat{T} \leftarrow \widehat{T} + 1$
6:     if $\mathcal{H}_E(uw)(\mathcal{H}_V(u) + \mathcal{H}_V(w)) \geq 1$ then
7:         $\widehat{E} \leftarrow \widehat{E} \cup \{uw\}$
8: return $\frac{\widehat{T}}{pq^2}$
```

Where $\mathcal{H}_V : V \to \{0,1\}$ is a pairwise-independent hash family such that $\mathbb{E}_{h\in\mathcal{H}_V}[h(v)] = p$ and $\mathcal{H}_E : E \to \{0,1\}$ is a fully-independent hash function such that $\mathbb{E}_{h\in\mathcal{H}_E}[h(e)] = q$.

The details of the algorithm will help us determine the correct settings for p and 1 respectively. In particular, note that there are at least $\frac{T}{\Delta_V}$ many vertices involved in some triangle. So to ensure we sample a non-zero number of these "triangle-bearing" vertices in expectation, we must set $p \geq \frac{\Delta_V}{T}$. Similarly, there are at least $\frac{T}{\Delta_E}$ many "triangle-bearing" edges. Our algorithm samples edges with probability $\leq 2pq$, so we must set $pq \geq \frac{\Delta_E}{T}$. Lastly, triangle is preserved and contributes to the estimator with probability pq^2, so $pq^2 \geq \frac{1}{T}$. Solving for q using these constraints, we get that $q = \max\{\frac{\Delta_E}{\Delta_V}, \frac{1}{\sqrt{\Delta_V}}\}$. These settings of p and q are sufficient to achieve optimality.

We now show correctness of the algorithm.

Lemma 2 *Estimator $\widehat{T}$ is unbiased. That is,* $\mathbb{E}[\widehat{T}] = T$

Proof. Notice that

$$\widehat{T} = \frac{1}{pq^2} \sum_{t=(uv,vw,uw)\in G} X_t$$

where the edges of triangle t arrive in order (uv, vw, uw) (to mitigate triple-counting) and X_t is the indicator random variable denoting whether triangle t is preserved via the sampling or not. Therefore,

$$\begin{aligned}
\mathbb{E}[\widehat{T}] &= \mathbb{E}\left[\frac{1}{pq^2}\sum_{t=(uv,vw,uw)\in G} X_t\right] = \frac{1}{pq^2}\sum_{t=(uv,vw,uw)\in G}\mathbb{E}[X_t] \\
&= \frac{1}{pq^2}\sum_{t=(uv,vw,uw)\in G}\Pr[\mathcal{H}_E(vu)=1]\Pr[\mathcal{H}_E(vw)=1]\Pr[\mathcal{H}_V(v)=1] \\
&= \sum_{t=(uv,vw,uw)\in G} 1 \\
&= T
\end{aligned}$$

where we have used the fact that $\Pr[X_t=1]=\Pr[\mathcal{H}_E(vu)=1]\Pr[\mathcal{H}_E(vw)=1]\Pr[\mathcal{H}_V(v)=1]$ since vertex v, edge vu, and edge vw must be sampled to preserve the triangle.

□

Lemma 3 $\mathbb{V}[\widehat{T}] \le \frac{T}{pq^2}+\frac{T\Delta_E}{pq}+\frac{T\Delta_V}{p} \le 3T^2$

Proof. We compute $\mathbb{E}[\widehat{T}^2]$ case by case. When

$$\mathbb{E}[\widehat{T}^2]=\frac{1}{p^2q^4}\sum_{(uv,vw,uw),(xy,yz,xz)\in G}\mathbb{E}[X_{t_1}X_{t_2}]$$

1. When $(uv,vw,uw)=(xy,yz,xz)$, the two triangles are the same. Therefore, $\mathbb{E}[X_{t_1}X_{t_2}]=\mathbb{E}[X_t^2]=\Pr[X_t=1]=pq^2$. There are T such pairs of triples.
2. When $|\{uv,vw\}\cap\{xy,yz\}|=1$, it follows that $v=y$. Therefore, $\mathbb{E}[X_{t_1}X_{t_2}]=pq^3$ since vertex v must be sampled and all 3 unique edges in the set $\{uv,vw,xy,yz\}$ must be sampled. Because each triangle can share an edge with at most Δ_E others, there are at most $T\Delta_E$ many such pairs of triples.
3. When $|\{uv,vw\}\cap\{xy,yz\}|=0$ but $u=x$, $\mathbb{E}[X_{t_1}X_{t_2}]=pq^4$ since vertex v must be sampled and all 4 unique edges in the set $\{uv,vw,xy,yz\}$ must be sampled. Because each triangle can share a vertex with at most Δ_V others, there are at most $T\Delta_V$ many such pairs of triples.
4. Lastly, when $\{u,v\}\cap\{x,y\}=\emptyset$, $\mathbb{E}[X_{t_1}X_{t_2}]=p^2q^4$ since vertices v and y must be sampled and all 4 unique edges in the set $\{uv,vw,xy,yz\}$ must be sampled. There are at most T^2 many pairs of triangles, and therefore T^2 many such pairs of triples.

These cases are exhaustive and therefore give

$$\begin{aligned}
\mathbb{E}[\widehat{T}^2] &\le \frac{1}{p^2q^4}(pq^2T+pq^3T\Delta_E+pq^4T\Delta_V+p^2q^4T^2) \\
&= T/pq^2+T\Delta_E/pq+T\Delta_V/p+T^2
\end{aligned}$$

From lemma 2, we then get

$$\begin{aligned}
\mathbb{V}[\widehat{T}] &= \mathbb{E}[\widehat{T}^2]-\mathbb{E}[\widehat{T}]^2 \\
&\le T/pq^2+T\Delta_E/pq+T\Delta_V/p+T^2-T^2 \\
&\le T/\left(\frac{\Delta_V}{T}\cdot\frac{1}{\Delta_V}\right)+T\Delta_E/\left(\frac{\Delta_V}{T}\cdot\frac{\Delta_E}{\Delta_V}\right)+T/\left(\frac{\Delta_V}{T}\right) \\
&= 3T^2
\end{aligned}$$

□

Theorem 3 There exists an algorithm that returns $\widehat{T} \in (1 \pm \epsilon)T$ with probability $1 - \delta$ and uses $O\left(\frac{m}{T}\left(\Delta_E + \sqrt{\Delta_V}\right) \log n \frac{\log \frac{1}{\delta}}{\epsilon^2}\right)$ space in expectation

Proof. By our settings of p and q, algorithm 2 returns an unbiased estimator $\widehat{T}$ with variance at most $3T^2$. A new estimator $\widehat{T}'$ constructed by independently running algorithm 2 $\frac{6}{\epsilon^2}$ times and taking the mean is similarly unbiased and has variance $\mathbb{V}[\widehat{T}'] = \frac{6}{\epsilon^2} \cdot \frac{\epsilon^4}{6^2} \mathbb{V}[\widehat{T}] \leq \frac{\epsilon^2 T^2}{2}$. By Chebyshev's inequality, $\widehat{T}' \notin (1 \pm \epsilon)T$ with probability $\leq \frac{1}{2}$. Finally, we construct estimator $\widehat{T}''$ by independently running and computing $\widehat{T}'$ $O(\log(\frac{1}{\delta}))$ times and taking the median. Estimator $\widehat{T}''$ is then $\notin (1 \pm \epsilon)$ with probability $1 - \delta$.

Notice that algorithm 2 stores an edge uv if $\mathcal{H}_E(uv) \cdot (\mathcal{H}_V(u) + \mathcal{H}_V(v)) \geq 1$ which happens with probability $\leq 2pq$. Each edge takes $\log n$ bits to store. Furthermore, has functions $\mathcal{H}_V$ and $\mathcal{H}_E$ can be constructed and stored in $O(\log n)$ bits of space [**CW79**]. Therefore, algorithm 2 has expected space complexity $O(mpq \log n)$. Thus, computing $\widehat{T}''$ takes $O(mpq \log n \frac{\log \frac{1}{\delta}}{\epsilon^2}) = O\left(\frac{m}{T}\left(\Delta_E + \sqrt{\Delta_V}\right) \log n \frac{\log \frac{1}{\delta}}{\epsilon^2}\right)$ space in expectation. □

3 Four-Cycle Counting

3.1 A 3-pass Algorithm [V20]

Algorithm 3 3-pass Four-Cycle Counting with $\widetilde{O}(\epsilon^{-2} m / T^{1/3})$ Space

```
1: **First Pass:**
2: Let $p = \widetilde{O}(\epsilon^{-2}/T^{1/3})$ be the sampling rate
3: Let $S_E$ be a random subset of edges constructed by including each edge w.p. $p$
4: Let $Z_V, Q_V$ be random subsets of nodes constructed by including each node w.p. $p$
5: Collect incident edges of $Z_V, Q_V$, call then $Z_E, Q_E$
6:
7: **After First Pass:**
8: For each pair $(u, v)$, let $q(u, v)$ be the number of wedges with center in $Q_V$ and endpoints $u, v$
9: Let $C_{uv}$ be a crystal with endpoints in $u, v$, it is called heavy if $q(u, v) \geq pT^{1/3}$
10: A wedge is called heavy if it belongs to a heavy crystal
11: Let $\hat{t}(C_{uv}) = \binom{q(u,v)/p}{2}$
12: Let $\hat{T}_H = \sum_{\text{heavy } C_{uv}} \hat{t}(C_{uv})$
13:
14: **Second Pass:** For each $e$ in the stream:
15: **if** $e$ forms a 4-cycle with 3 edges in $S_E$ without heavy wedges **then**
16:     Store $(e, \tau)$
17:
18: **Third Pass:**
19: Let $A_0$ be the number of $(e, \tau)$ containing no heavy edges
20: Let $A_1$ be the number of $(e, \tau)$ where $e$ is heavy and the other edges are light
21: $\hat{T}_L = A_0/(4p^3) + A_1/p^3$
22:
23: **return** $\hat{T} = \hat{T}_H + \hat{T}_L$
```

Main idea: The high-level approach towards 4-cycle counting is to sample a set of edges in one pass, and in separate pass try to form 4-cycles using the incoming edge and 3 edges from the sampled set. The challenge in this approach is that if an edge or wedge is involved in many 4-cycles, then the variance of the estimator becomes large. To address this problem, we adopt an approach similar to the triangle counting, by considering heavy edges and wedges separately from the rest of edges.

- For heavy wedges, we use the samples Q_V, Q_E to estimate the number of 4-cycles in which wedge w is involved. A graph structure that helps intuiting this situation is the crystal graph in Fig. 2, which contains k wedges that share the same endpoints u, v. It is not hard to see that each k-crystal contributes $\binom{k}{2}$ many 4-cycles, which is significant. Therefore, if we estimate a crystal to be heavy, we add its expected contribution to the overall count.
- For heavy edges, we build an oracle using samples Z_V, Z_E that estimate how many 4-cycles in which edge e is involved. It can be shown that this oracle can be constructed satisfying Def. 1. Using this oracle, we consider 4-cycles that contains no heavy wedge and at most one heavy edge. This gives an accurate estimate since the number of 4-cycles that contains no heavy wedges and more than one heavy edges is at most $\epsilon T/6$ [**MV20**].

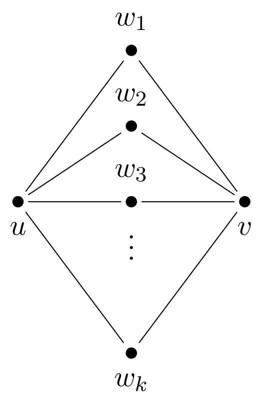

Figure 2: Crystal graph C_{uv} with weight k

Lemma 4 *$\hat{T}_H \in (1 \pm \epsilon/4)T_H$ w.h.p.*

Proof. We first show that

- if $q(u,v) \geq pT^{1/3}$, then $|C_{uv}| \geq \frac{1}{2}T^{1/3}$
- if $q(u,v) \leq pT^{1/3}$, then $|C_{uv}| \leq 2T^{1/3}$

where $|C_{uv}|$ denotes the number of wedges in C_{uv}. By applying the Chernoff bound, if $|C_{uv}| \geq \frac{1}{2}T^{1/3}$, then

$$\Pr\left[q(u,v) \leq pT^{1/3}\right] \leq e^{-2pT^{1/3}} = e^{-O(\log n)} = 1/\text{poly}(n)$$

The other side follows similarly. Now we want to estimate how far $q(u,v)/p$ deviates from $t(C_{uv})$. Again, by Chernoff bound,

$$\Pr[|q(u,v) - p|C_{uv}| \geq (\epsilon/20)p|C_{uv}|] \leq 2e^{-\epsilon^2 p|C_{uv}|/1200} \leq 1/\text{poly}(n)$$

Since

$$\binom{(1 \pm \epsilon/20)|C_{uv}|}{2} \in (1 \pm \epsilon/4)\binom{|C_{uv}|}{2}$$

we have $\hat{t}(C_{uv}) \in (1 \pm \epsilon/4)\binom{|C_{uv}|}{2}$.

Lastly, we consider the possibility that a 4-cycle is included in two heavy crystals, which causes double counting. Since each heavy crystal contains $\Omega(T^{2/3}$ 4-cycles the number of heavy crystals is at most $O(T^{1/3})$. Two distinct 4-cycles can belong to at most two crystals, so the maximal double counting is $T^{2/3} \le (\epsilon/4)T$. Thus, we can obtain the desired accuracy by union bounding all heavy crystals. □

Lemma 5 $\widehat{T}_L \in (1 \pm \epsilon/2)T_L$ *w.h.p.*

Proof. Let T_i be the number of 4-cycles in G with i heavy edges. Let $\hat{T}_0 = A_0/(4p^3), \hat{T}_1 = A_1/p^3$. It is immediate that $\mathbb{E}[\hat{T}_0] = T_0$ and $\mathbb{E}[\hat{T}_1] = T_1$.

We want to use Chebyshev's bound to show that

$$\Pr\Big[|\hat{T}_0 - T_0| \le (\epsilon/4)T\Big] \le 1/16$$

It suffices to show that $\mathbb{V}[\hat{T}_0] \le \epsilon^2 T^2/256$. Let $\mathcal{H}_0$ denote the set of 3-paths that contribute to T_0. Let X_q be indicator variable denoting whether all 3 edges of the 3-path $q \in \mathcal{H}_0$ are sampled.

$$\begin{aligned}
\mathbb{V}[\hat{T}_0] &= \mathbb{V}\left[\frac{1}{4p^3}\sum_{q\in\mathcal{H}_0} X_q\right] \\
&\le \frac{1}{16p^6}\left(\sum_{q\in\mathcal{H}_0}\mathbb{V}[X_q] + \sum_{q,t\in\mathcal{H}_0: q\ne t, q\cap t\ne\emptyset} \mathbb{COV}[X_q X_t]\right) \\
&\le \frac{1}{16p^6}\left(\sum_{q\in\mathcal{H}_0}\mathbb{V}[X_q] + \sum_{q,t\in\mathcal{H}_0: q\ne t, q\cap t\ne\emptyset} \mathbb{E}[X_q X_t]\right) \\
&\le \frac{1}{16p^6}\left(\sum_{q\in\mathcal{H}_0} p^3 + \sum_{q\in\mathcal{H}_0}\sum_{t\in\mathcal{H}_0:|q\cap t|=1} p^5 + \sum_{q\in\mathcal{H}_0}\sum_{t\in\mathcal{H}_0:|q\cap t|=2} p^4\right) \\
&\le \frac{1}{16p^6}\left(|\mathcal{H}_0|p^3 + |\mathcal{H}_0|\cdot cT^{2/3}p^5 + |\mathcal{H}_0|\cdot cT^{1/3}p^4\right) \\
&\le T/4p^3 + cT^{2/3}/4p + cT^{1/3}/4p^2 \\
&\le \epsilon^2 T^2/256
\end{aligned}$$

□

which in essence analyzes ways in which 3-paths intersect with each other (at one edge and at two edges). $\Pr\Big[|\hat{T}_1 - T_1| \le (\epsilon/4)T\Big] \le 1/16$ can be proved similarly.

Theorem 4 [**V20**] Alg. 3 returns $\hat{T} \in (1 \pm \epsilon)T$ using $\widetilde{O}(\epsilon^{-2}m/T^{1/3})$ space, w.p. $\ge 90\%$.

Proof. Combining Lemma 4 and 5, and the fact that the number of 4-cycles that contains no heavy wedges and more than one heavy edges is at most $\epsilon T/6$ [**MV20**], we can conclude that the accuracy of the algorithm is as claimed.

The space complexity follows from the fact that sets S_E, Q_E, Z_E all have the expected size $mp = O(\epsilon^{-2} m \log n/T^{1/3})$. The expected number of sampled cycles from the second pass is $4T/p^3 = O(\log n)$. Therefore, the total expected space complexity is same as claimed. □

4 Diamond Counting – Our Results

We present a novel 3-pass streaming algorithm for approximately counting diamonds, a subgraph with the structure ◁▷ (we remark that the presented algorithm, theorems, and results are a work in progress and may potentially have bugs). The diamond exhibits many properties that can be exploited by triangle/4-cycle counting techniques that we have seen in the literature. Because the diamond differs only by one edge from the 4-clique, our techniques for counting diamonds may provide insights for counting 4-cliques, which is a problem of great practical importance.

4.1 A 3-pass Algorithm

Algorithm 4 3-pass Diamond Counting with $\widetilde{O}\left(T^{1/4} + \epsilon^{-2}m/T^{3/4}\right)$ Space

```
1:  Let $p = O(\epsilon^{-2}\log n/T^{3/4})$ be the sampling rate
2:  $E_L = \emptyset, V_L = \emptyset, E_H = \emptyset, E_S = \emptyset,\ \hat{D}_L = 0,\ \hat{D}_H = 0, \hat{T}_H = [0, ..., 0], \Theta = T^{3/4}$
3:
4:  **First Pass:** Build heavy edge oracle with threshold $\Theta$.
5:
6:  **Second Pass:** on $e = (u, v)$
7:  **if** oracle($e$) $= L$ **then**
8:      With probability $p$, **do**
9:          $E_L \leftarrow E_L \cup \{e\}$, $V_L \leftarrow V_L \cup \{u, v\}$, $\hat{D}_L(e) = 0$
10: **if** oracle($e$) $= H$ **then**
11:     $E_H \leftarrow E_H \cup \{e\}$, $\hat{D}_H(e) = 0$
12:
13: **Third Pass:** on $e = (u, v)$
14: **if** $u \in V_L$ or $v \in V_L$ **then**
15:     $E_S \leftarrow E_S \cup \{e\}$
16: **for** each $e_L \in E_L, e_H \in E_H$ **do**
17:     **if** $(e_L, e_H, e)$ is a triangle **then**
18:         $\hat{T}_H(e_H) \leftarrow \hat{T}_H(e_H) + 1/p$
19: **for** each $e_{H_1}, e_{H_2}, \in E_H$ **do**
20:     **if** $(e_{H_1}, e_{H_2}, e)$ is a triangle **then**
21:         $\hat{T}_H(e_{H_1}) \leftarrow \hat{T}_H(e_{H_1}) + 1$
22:         $\hat{T}_H(e_{H_2}) \leftarrow \hat{T}_H(e_{H_2}) + 1$
23:
24: **After Pass:**
25: **for** $e_L = (u_1, u_2) \in E_L$ **do**
26:     **for** $v_1, v_2 \in V$ **do**
27:         **if** $(v_1, u_1), (v_1, u_2), (v_2, u_1), (v_2, u_2) \in E_S$ **then**
28:             $\hat{D}_L(e_L) \leftarrow \hat{D}_L(e_L) + 1$
29: $\hat{D}_L \leftarrow \frac{1}{p}\sum_{e_L \in E_L} \hat{D}_L(e_L)$
30: $\hat{D}_H \leftarrow \sum_{e_H \in E_H} \binom{\hat{T}_H(e_H)}{2}$
31: **return** $\hat{D}_L + \hat{D}_H$
```

Before analyzing the algorithm, we provide some intuition. First, we observe that a diamond contains a middle edge that is incident to two triangles. We define the *lotus* structure based in (i, j) (Fig. 3) where the base edge (i, j) is incident to k triangles (petals). The key insight is that a k-petal lotus graph contributes $\binom{k}{2}$-many diamonds, which is similar to the contribution of crystal graphs in the 4-cycle counting case. Therefore, to count diamonds in G, we have to approximate the number of petals based in every $e \in E(G)$. Similar to triangle/4-cycle counting, we will consider *heavy/light* edges separately in order to reduce variance introduced by dropping heavy edges while sampling. Similar to many triangle and four-cycle counting algorithms, we build a heavy edge oracle in the first pass.

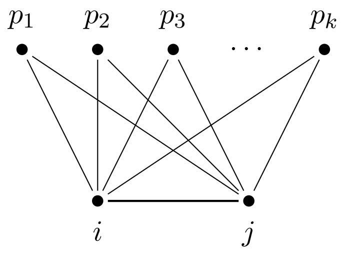

Figure 3: Lotus graph with k petals

The algorithm has three passes. In the first pass, we build a heavy edge oracle as before by sampling edges. In the second pass, we categorize heavy and light edges using the oracle, collect all heavy ones into E_H, and sample the light edges with a fixed rate into E_L. In the third pass, we collect all the edges incident to the light edges in E_L, and then try to make triangles using the new edge and two edges from E_L and E_H. For each heavy edge $e_H \in E_H$, we keep count of how many triangles are formed; in other words, we approximate how many petals are in the lotus based in e_H. After all passes, we iterate through all $e_L \in E_L$ and all pairs of vertices, and count how many diamonds based in e_L can be formed. Finally, we output an estimator that re-weights our counts according to p.

We first show that the our settings of $p = O(\epsilon^{-2} \log n / T^{3/4})$ and $\Theta = T^{3/4}$ give the same oracle guarantees as that of lemma 1.

Proof. Sample each vertex $v \in V(G)$ into set Z independently with probability p, and let $\widetilde{x}_e = |\{w \in Z : u, v \in \Gamma(w)\}|$, and let x_e be the true number of triangles that e is part of. Then, we can apply Chernoff bound. For any given e, if e is heavy, then $x_e \geq cT^{3/4}$, and

$$\Pr[\text{oracle}(e) = L] = \Pr\left[\widetilde{x}_e < pT^{3/4}\right] \leq e^{-O(pT^{3/4})} = e^{-O(\epsilon^{-2} \log n)} = n^{-O(\epsilon^{-2})}$$

The other side can be proved similarly. By a union bound over all $e \in E(G)$, the oracle implies the correct result with probability $1 - 1/\text{poly}(n)$. □

For the remainder of the proof, we will assume that each edge is correctly labeled by the oracle and a union bound over each edge. We now proceed to prove that the heavy and light diamond count estimators concentrate well around their true values. Let $T(e)$ denote the number of triangles containing edge e. Let e_H and e_L denote a sampled edge $e \in E$ that is labeled heavy and light by the oracle respectively.

Lemma 6 *$\hat{D}_H \in (1 \pm \epsilon/4) D_H$ with probability* 0.99

Proof. We claim that it suffices to show that $\widehat{T}_H(e) \in (1 \pm \frac{\epsilon}{20}) T(e)$ for all $e \in E$ considered heavy by the oracle. This is because

$$\binom{(1 + \frac{\epsilon}{20}) T(e)}{2} \leq \left(1 + \frac{\epsilon}{4}\right) \binom{T(e)}{2} = \left(1 + \frac{\epsilon}{4}\right) D(e)$$

and

$$\binom{(1-\frac{\epsilon}{20})T(e)}{2} \geq \left(1-\frac{\epsilon}{4}\right)\binom{T(e)}{2} = \left(1+\frac{\epsilon}{4}\right)D(e)$$

From algorithm 4,

$$\hat{T}_H(e_H) = \sum_{e_H \in E} X_{e_{H_1},e_{H_2}} + \sum_{e_L \in E} X_{e_H,e_L}$$

where X_{e_1,e_2} is the indicator random variable that takes value 1 if the unique triangle containing e_1 and e_2 is preserved by sampling. Because every edge is correctly labeled by the oracle and we store all heavy edges, the first term in the summation exactly counts the number of heavy-heavy triangles that e_H is a part of. We now show that the second term concentrates well around the true number of heavy-light triangles that e_H is a part of, which we denote as $T_L(e_H)$.
We split the analysis into two cases

1. First, suppose that $\frac{1}{p}\sum_{e_L \in E} X_{e_H,e_L} \geq \Theta$. One potential issue in the algorithm is that triangles containing one heavy edge and two light edges can be double-counted if both light edges are sampled. We first show that each double count changes the estimator only by a small fraction, then proceed to show concentration of the second term.

   Each time we double count a triangle with e_H, we would have to increment $\hat{T}_H(e_H)$ by $1/p$ twice, and we double counted one triangle. This means that there is at most a $\frac{1}{2/p} = p/2 = O(1/T^3/4)$ fraction of $\hat{T}_H(e_H)$ that is caused by double count. As long as $\hat{T}_H(e_H) < T$ (which must be true if $\hat{T}_H(e_H)$ is an accurate estimator), then the amount of contribution of double counting is at most $T \cdot O(1/T^3/4) = T^{1/4} \leq \epsilon T_H(e)$. Therefore, we can afford to neglect the contribution of double counts, and assume that all preserved triangles are not double counted.

   Notice that the estimator is unbiased:

$$\mathbb{E}\left[\frac{1}{p}\sum_{e_L \in E} X_{e_H,e_L}\right] = \frac{1}{p}\sum_{e_L \in E}\mathbb{E}[X_{e_H,e_L}] = |T_L(e_H)|$$

   Since $\Pr[X_{e_H,e_L} = 1] = p$, the probability that e_L is sampled. Here, $T_L(e_H)$ is the total number of triangles with heavy-light wedges containing e_H. By the Chernoff bound,

$$\Pr\left[\frac{1}{p}\sum_{e_L \in E_L} X_{e_h,e_L} \notin (1 \pm \epsilon)\sum_{e_L \in E} X_{e_H,e_L}\right] \leq e^{-\epsilon^2 \cdot \log n/\epsilon^2} = \frac{1}{poly(n)}$$

2. Otherwise, suppose that $\frac{1}{p}\sum_{e_L \in E} X_{e_H,e_L} <= o(\Theta)$. In this case, $\hat{T}_H(e_H)$ is asymptotically dominated by the first term. Because e_H is heavy, the first term is $\geq \Theta$. Because the first term is an exact count, we are done.

□

Lemma 7 $\widehat{D}_L \in (1 \pm \epsilon/2)D_L$ *with probability* 0.99

Proof. Let $\mathcal{D}$ be the set of all diamonds, and $D = |\mathcal{D}|$.
Let $D(e)$ be the number of diamonds with base e.

Let $T(e)$ be the number of triangles containing e.
Let $E(t)$ denote the number of edges that are part of exactly t triangles. Note that since each triangle has 3 edges, $\sum_{t\geq 1} E(t) \leq 3T$.
Let X_e be the indicator function denoting whether edge e is sampled.
Let X_d be the indicator function denoting whether diamond d is sampled.
We first show that the estimator is unbiased:

$$\mathbb{E}[\widehat{D}_L] = \sum_{e\in E: e \text{ light}} \mathbb{E}[X_e] D(e) = p \sum_{e\in E: e \text{ light}} D(e) = pD_L$$

Then, we show that we can bound the variance to $\mathbb{V}(\widehat{D}_L) \leq \epsilon^2 D^2$ in order to apply Chebyshev's bound:

$$\begin{aligned}
\mathbb{V}[\widehat{D}_L] &= \mathbb{V}\left[\frac{1}{p}\sum_{e_L\in E_L} \widehat{D}_L(e_L)\right] \\
&\leq \frac{1}{p^2}\left(\sum_{d\in\mathcal{D}} \mathbb{V}[X_d] + \sum_{\substack{d_1,d_2\in\mathcal{D}\\ d_1\neq d_2, d_1\cap d_2\neq\emptyset}} \mathbb{COV}[X_{d_1}, X_{d_2}]\right) \\
&\leq \frac{1}{p^2}\left(\sum_{d\in\mathcal{D}} \mathbb{E}[X_d^2] + \sum_{\substack{d_1,d_2\in\mathcal{D}\\ d_1\neq d_2, d_1\cap d_2\neq\emptyset}} \mathbb{E}[X_{d_1}, X_{d_2}]\right) \\
&\leq \frac{1}{p^2}\left(pD + \sum_{\substack{d_1,d_2\in\mathcal{D}\\ |d_1\cap d_2|=1}} \mathbb{E}[X_{d_1}, X_{d_2}]\right) \\
&\leq \frac{1}{p^2}\left(pD + p\sum_{e: T(e)\leq\Theta} T(e)^4\right) && (1) \\
&\leq \frac{1}{p^2}\left(pD + p\sum_{t=1}^{\Theta} t^4 E(t)\right) && (2) \\
&\leq \frac{1}{p^2}\left(pD + p\Theta^3\sum_{t=1}^{\Theta} tE(t)\right) \\
&\leq \frac{1}{p^2}\left(pD + 3p\Theta^3 T\right) && (3) \\
&\leq D/p + 3\Theta^3 T/p
\end{aligned}$$

where (1) follows by considering that at if edge e is involved in $T(e)$ triangles, then the number of ways that e can be shared among diamonds is upper bounded by $\binom{T(e)}{4} \leq T(e)^4$. (2) follows by summing over all values of $T(e)$ instead of over e. (3) follows from the fact $\sum_{t\geq 1} E(t) \leq 3T$.

By noting that $D \leq T^2$, we can set $\Theta = O(T^{3/4}), p = \widetilde{O}(\epsilon^{-2}/T^{3/4})$, which gives us $\mathbb{V}(\widehat{D}_L) \leq \epsilon^2 D^2$, which allows us to apply Chebyshev's bound over our estimator. □

Theorem 1 (restated) There exists a 3-pass algorithm that returns a $(1 \pm \epsilon)$ multiplicative approximation to the number of diamonds in a streamed graph with space complexity

$$\widetilde{O}\left(T^{1/4} + \epsilon^{-2} m / T^{3/4}\right)$$

and success probability 0.99. Here, T denotes the number of triangles in the graph.

Proof. The accuracy follows from the combination of Lemma 6 and 7. The expected space complexity follows from the fact that the sampled set E_S has expected size $mp = O(\epsilon^{-2} m \log n / T^{3/4})$, and the expected number of heavy edges in E_H is upper bounded by $T^{1/4}$ by our definition of heaviness. Therefore, the overall space complexity is as claimed. $\square$

References

[BHLP11] Jonathan W. Berry, Bruce Hendrickson, Randall A. LaViolette, and Cynthia A. Phillips. Tolerating the community detection resolution limit with edge weighting. Phys. Rev. E, 83:056119, May 2011.

[CCKM10] Amit Chakrabarti, Graham Cormode, Ranganath Kondapally, and Andrew McGregor. Information cost tradeoffs for augmented index and streaming language recognition. In Proceedings of the 51st FOCS, pages 387–396. IEEE, 2010.

[CEI+22] Chen, Justin Y., et al. “Triangle and four cycle counting with predictions in graph streams.” arXiv preprint arXiv:2203.09572 (2022).

[CW79] J Lawrence Carter and Mark N Wegman. Universal classes of hash functions. Journal of computer and system sciences, 18(2):143–154, 1979.

[EM02] Jean-Pierre Eckmann and Elisha Moses. Curvature of co-links uncovers hidden thematic layers in the world wide web. Proceedings of the National Academy of Sciences, 99(9):5825–5829, 2002.

[GKK+07] Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald De Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In Proceedings of the thirty-ninth annual ACM symposium on Theory of computing, pages 516–525. ACM, 2007.

[JK21] Jayaram, Rajesh, and John Kallaugher. ”An optimal algorithm for triangle counting in the stream.” arXiv preprint arXiv:2105.01785 (2021).

[MSV16] McGregor, Andrew, Sofya Vorotnikova, and Hoa T. Vu. ”Better algorithms for counting triangles in data streams.” Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. 2016.

[MV20] McGregor, Andrew, and Sofya Vorotnikova. ”Triangle and four cycle counting in the data stream model.” Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems. 2020.

[V20] Vorotnikova, Sofya. ”Improved 3-pass algorithm for counting 4-cycles in arbitrary order streaming.” arXiv preprint arXiv:2007.13466 (2020).