A Second Look at Counting Triangles in Graph Streams

Graham Cormode, Hossein Jowhari[a,b,]

[a]G.Cormode@warwick.ac.uk, *Corresponding author*
[b]hjowhari@sfu.ca

Abstract

In this paper we present improved results on the problem of counting triangles in edge streamed graphs. For graphs with m edges and at least T triangles, we show that an extra look over the stream yields a two-pass streaming algorithm that uses $O(\frac{m}{\epsilon^{4.5}\sqrt{T}})$ space and outputs a $(1+\epsilon)$ approximation of the number of triangles in the graph. This improves upon the two-pass streaming tester of Braverman, Ostrovsky and Vilenchik, ICALP 2013, which distinguishes between triangle-free graphs and graphs with at least T triangle using $O(\frac{m}{T^{1/3}})$ space. Also, in terms of dependence on T, we show that more passes would not lead to a better space bound. In other words, we prove there is no constant pass streaming algorithm that distinguishes between triangle-free graphs from graphs with at least T triangles using $O(\frac{m}{T^{1/2+\rho}})$ space for any constant $\rho \geq 0$.

1. Introduction

Many applications produce output in form of graphs, defined an edge at a time. These include social networks that produce edges corresponding to new friendships or other connections between entities in the network; communication networks, where each edge represents a communication (phone call, email, text message) between a pair of participants; and web graphs, where each edge represents a link between pages. Over such graphs, we wish to answer questions about the induced graph, relating to the structure and properties.

One of the most basic structures that can be present in a graph is a triangle: an embedded clique on three nodes. Questions around counting the number of triangles in a graph have been widely studied, due to the inherent interest in the problem, and because it is a necessary stepping stone to answering questions around more complex structures in graphs. Triangles are of interest within social networks, as they indicate common friendships: two friends of an individual are themselves friends. Counting the number of friendships within a graph is therefore a measure of the closeness of friendship activities. Another use of the number of triangles is as a parameter for evaluation of large graph models [LBKT08].

For these reasons, and for the fundamental nature of the problem, there have been numerous studies of the problem of counting or enumerating triangles in various models of data access: external memory [LWZW10, HTC13]; map-reduce [SV11, PT12, TKMF09]; and RAM model [SW05, Tso08]. Indeed, it seems that triangle counting and enumeration is becoming a *de facto* benchmark for testing "big data" systems and their ability to process complex queries. The reason is that the problem captures an essentially hard problem within big data: accurately measuring the degree of correlation. In this paper, we study the problem of triangle counting over (massive) streams of edges. In this case, lower bounds from communication complexity can be applied to show that exactly counting the number of triangles essentially requires storing the full input, so instead we look for methods which can approximate the number of triangles. In this direction, there has been series of works that have attempted to capture the right space complexity for algorithms that approximate the number of triangles. However most of these works have focused on one pass algorithms and thus, due to the hard nature of the problem, their space bounds have become complicated, suffering from dependencies on multiple graph parameters such as maximum degree, number of paths of length 2, number of cycles of length 4, etc.

In a recent work by Braverman *et al.* [BOV13], it has been shown that at the expense of an extra pass over stream, a straightforward sampling strategy gives a sublinear bound that depends only on m (number of edges) and T (a lower bound on the number of triangles[1]). More precisely [BOV13] have shown that one extra pass yields an algorithm that distinguishes between triangle-free graphs from graphs with at least T triangles using $O(\frac{m}{T^{1/3}})$ words of space. Although their algorithm does not give an estimate of the number of triangles and more important is not clearly superior to the $O(\frac{m\Delta}{T})$ one pass algorithm by [PT12, PTTW13] (especially for graphs with small maximum degree Δ), it creates some hope that perhaps with the expense of extra passes one could get improved and cleaner space complexities that beat the one pass bound for a wider range of graphs. In particular

[1] In this and prior works, some assumption on the number of triangles is required. This is due in part to the fact that distinguishing triangle-free graphs from those with one or more triangle requires space proportional to the number of edges. Other works have required even stronger assumptions, such as a bound on T_2, the number of paths of length 2, or the maximum degree of the graph

one might ask is there a $O(\frac{m}{T})$ space multi-pass algorithm? In this paper, while we refute such a possibility, we show that a more modest bound is possible. Specifically here we show modifications to the sampling strategy of [BOV13] along with a different analysis results in a 2-pass $(1+\epsilon)$ approximation algorithm that uses only $O(\frac{m}{\epsilon^{4.5}\sqrt{T}})$ space. We also observe that this bound is attainable in one pass– if we can make the string assumption that the order of edge arrivals is random. Additionally, via a reduction to a hard communication complexity problem, we demonstrate that this bound is optimal in terms of its dependence on T. In other words there is no constant pass algorithm that distinguishes between triangle-free graphs from graphs with at least T triangles using $O(\frac{m}{T^{1/2+\rho}})$ for any constant $\rho > 0$. We also give a similar two pass algorithm that has better dependence on ϵ but sacrifices the optimal dependence on T. Our results are summarized in Figure 2 in terms of the problem addressed, bound provided, and number of passes.

In line with prior work, we assume a simple graph – that is, each edge of the graph is presented exactly once in the stream. Note that our lower bounds immediately hold for the case when edges are repeated.

Algorithms for Triangle Counting in Graph Streams. The triangle counting problem has attracted particular attention in the model of graph streams: there is now a substantial body of study in this setting. Algorithms are evaluated on the amount of space that they require, the number of passes over the input stream that they take, and the time taken to process each update. Different variations arise depending on whether deletions of edges are permitted, or the stream is 'insert-only'; and whether arrivals are ordered in a particular way, so that all edges incident on one node arrive together, or the edges are randomly ordered or adversarially ordered.

The work of Jowhari and Ghodsi [JG05] first studied the most popular of these combinations: insert-only, adversarial ordering. The general approach, common to many streaming algorithms, is to build a randomized estimator for the desired quantity, and then repeat this sufficiently many times to provide a guaranteed accuracy. Their approach begins by sampling an edge uniformly from the stream of m arriving edges on n vertices. Their estimator then counts the number of triangles incident on a sampled edge. Since the ordering is adversarial, the estimator has to keep track of all edges incident on the sampled edge, which in the worst case is bounded by Δ, the maximum degree. The sampling process is repeated $O(\frac{1}{\epsilon^2}\frac{m\Delta}{T})$ times (using the assumed bound on the number of triangles, T), leading to a total space requirement proportial to $O(\frac{1}{\epsilon^2}\frac{m\Delta^2}{T})$ to give an ϵ relative error estimation of t, the number of triangles in the graph. The parameter ε ensures that the error in the count is at most εt (with constant probability, since the algorithm is randomized). The process can be completed with a single pass over the input. Jowhari and Ghodsi also consider the case where edges may be deleted, in which case a randomized estimator using "sketch" techniques is introduced, improving over a previous sketch algorithm due to Bar-Yossef *et al.* [BYKS02].

The work of Buriol *et al.* [BFL$^+$06] also adopted a sampling approach, and built a one-pass estimator with smaller working space. An algorithm is proposed which samples uniformly an edge from the stream, then picks a third node, and scans the remainder of the stream to see if the triangle on these three nodes is present. Recall that n is the number of nodes in the graph, m is number of edges, and $T \leq t$ is lower bound on the (true) number of triangles. To obtain an accurate estimate of number of triangles in the graph, this procedure is repeated independently $O(\frac{mn}{\varepsilon^2 T})$ times to achieve ϵ relative error.

Recent work by Pavan *et al.* [PTTW13] extends the sampling approach of Buriol *et al.*: instead of picking a random node to complete the triangle with a sampled edge, their estimator samples a second edge that is incident on the first sampled edge. This estimator is repeated $O(\frac{m\Delta}{\epsilon^2 T})$ times, where Δ represents the maximum degree of any node. That is, this improves the bound of Buriol *et al.* by a factor of n/Δ. In the worst case, $\Delta = n$, but in general we expect Δ to be substantially smaller than n.

Braverman *et al.* [BOV13] take a different approach to sampling. Instead of building a single estimator and repeating, their algorithms sample a set of edges, and then look for triangles induced by the sampled edges. Specifically, an algorithm which takes two passes over the input stream distinguishes triangle-free graphs from those with T triangles in space $O(m/T^{1/3})$.

For graphs with $W \geq m$ where W is the number of wedges (paths of length 2), Jha *et al.* [JSP13] have shown a single pass $O(\frac{1}{\epsilon^2} m/\sqrt{T})$ space algorithm that returns an additive error estimation of the number of triangles.

Pagh and Tsourakakis [PT12] propose an algorithm in the MapReduce model of computation. However, it can naturally be adapted to the streaming setting. The algorithm conceptually assigns a "color" to each vertex randomly from C colors (this can be accomplished, for example, with a suitable hash function). The algorithm then stores each monochromatic edge, i.e. each edge from the input such that both vertices have the same color. Counting the number of triangles in this induced graph, and scaling up by a factor of C^2 gives an estimator for t. The space used is $O(m/C)$ in expectation. Setting C appropriately yields a one-pass algorithm with space $\tilde{O}(\frac{m}{T} J + \frac{m}{\sqrt{T}})$, where J denotes the maximum number of triangles incident on a single edge.

n	number of vertices
m	number of edges
$t(G)$	number of triangles in graph G
T	lower bound on $t(G)$
ε	relative error
δ	probability of error
Δ	maximum degree
J	maximum number of triangles incident on an edge
K	maximum number of triangles incident on a vertex
$\mathsf{Dist}(T)$	Distinguish graphs with T triangles from triangle-free graphs
$\mathsf{Estimate}(T, c)$	c approximate the number of triangles when there are at least T
Disj_p^r	Determine if two length p bitstrings of weight r intersect

Figure 1: Table of notation

Lower bounds for triangle counting. A lower bound in the streaming model is presented by Bar-Yossef *et al.* [BYKS02]. They argue that there are (dense) families of graphs over n nodes such that any algorithm that approximates the number of triangles must use $\Omega(n^2)$ space. The construction essentially encodes $\Omega(n^2)$ bits of information, and uses the presence or absence of a single triangle to recover a single bit. Braverman *et al.* [BOV13] show a lower bound of $\Omega(m)$ by demonstrating a family of graphs with m chosen between n and n^2. Their construction encodes m bits in a graph, then adds τ edges such that there are either τ triangles or 0 triangles, which reveal the value of an encoded bit.

For algorithms which take a constant number of passes over the input stream, Jowhari and Ghodsi [JG05] show that still $\Omega(n/T)$ space is needed to approximate the number of triangles up to a constant factor, based on a similar encoding and testing argument. Specifically, they create a graph that encodes two binary strings, so that the resulting graph has T triangles if the strings are disjoint, and $2T$ if they have an intersection. In a similar way, Braverman *et al.* [BOV13] encode binary strings into a graph, so that it either has no triangles (disjoint strings) or at least T triangles (intersecting strings). This implies that $\Omega(m/T)$ space is required to distinguish the two cases. In both cases, the hardness follows from the communication complexity of determining the disjointness of binary strings.

2. Preliminaries and Results

In this section, we define additional notation and define the problems that we study.

As mentioned above, we use $t(G)$ to denote the number of triangles in a graph $G = (V, E)$. Let $J(G)$ denote the maximum number of triangles that share an edge in G, and $K(G)$ the maximum number incident on any vertex. We use t, J and K when G is clear from the context.

Problems Studied. We define some problems related to counting the number of triangles in a graph stream. These all depend on a parameter T that gives a promise on the number of triangles in the graph.

$\mathsf{Dist}(T)$: Given a stream of edges, distinguish graphs with at least T triangles from triangle-free graphs.

$\mathsf{Estimate}(T, \epsilon)$: Given the edge stream of a graph with at least T triangles, output s where $(1-\epsilon)\cdot t(G) \leq s \leq (1+\epsilon)\cdot t(G)$.

Observe that any algorithm which promises to approximate the number of triangles for $\epsilon < 1$ must at least be able to distinguish the case of 0 triangles or T triangles. Consequently, we provide lower bounds for the $\mathsf{Dist}(T)$ problem, and upper bounds for the $\mathsf{Estimate}(T, \epsilon)$ problem. Our lower bounds rely on the hardness of well-known problems from communication complexity. In particular, we make use of the hardness of Disj_p^r:

Problem 1 *The Disj_p^r problem involves two players, Alice and Bob, who each have binary vectors of length p. Each vector has Hamming weight r, i.e. r entries set to one. The players want to distinguish non-intersecting inputs from inputs that do intersect.*

This problem is "hard" in the (randomized) communication complexity setting: it requires a large amount of communication between the players in order to provide a correct answer with sufficient probability [KN97]. Specifically, Disj_p^r requires $\Omega(r)$ bits of communication for any $r < p/2$, over multiple rounds of interaction between Alice and Bob.

Our Results. We summarize the results for this problem discussed in Section 1, and include our new results, in Figure 2. We observe that, in terms of dependence on T, we achieve tight bounds for 2 passes: Theorem 3 shows that we can obtain a dependence on $T^{-1/2}$, and Theorem 7 shows that no improvement for constant passes as a function of T can be obtained. It is useful to contrast to the results of [PT12], where a one pass algorithm achieves a dependence of $m/T^{1/2}$, but has an

Problem	Passes	Bound	Reference
Dist(T)	1	$\Omega(m)$	[BOV13]
Dist(T)	$O(1)$	$\Omega(m/T)$	[BOV13]
Dist(T)	2	$O(\frac{m}{T^{1/3}})$	[BOV13]
Estimate(T,ϵ)	1	$O(\frac{1}{\epsilon^2}\frac{m\Delta}{T})$	[PTTW13]
Estimate(T,ϵ)	1	$O(\frac{1}{\epsilon^2}(\frac{mJ}{T}+\frac{m}{T^{1/2}}))$	[PT12]
Estimate(T,ϵ)	2	$O(\frac{m}{\epsilon^{4/3}}\frac{\sqrt{\log n}}{T^{1/3}})$	Theorem 1
Estimate(T,ϵ)	2	$O(\frac{m}{\epsilon^{4.5}T^{1/2}})$	Theorem 3
Dist(T)	$O(1)$	$\Omega(\frac{m}{T^{2/3}})$	Theorem 6
Dist(T)	$O(1)$	$\Omega(\frac{m}{T^{1/2}})$ for $m=\Theta(n\sqrt{T})$	Theorem 7

Figure 2: Summary of results

additional term of mJ/T. This extra term can be large: as big as m in the case that all triangles are incident on the same edge; here, we show that this term can be avoided at the cost of an additional pass.

Our results improve over the 2-pass bounds given in [BOV13]. We show that the Estimate(T,ϵ) problem can be solved with dependence on $T^{-1/3}$ (not just the decision problem Dist(T)), and that the dependence on T can be improved to $T^{-1/2}$, at the expense of higher dependence on ϵ.

Comparing with the additive estimator of [JSP13], while our sampling strategy is somewhat similar, using an extra pass over the stream we return a relative error estimation of the number of triangles. However, [JSP13] requires the condition $W \geq m$ on W (the number of paths of length 2), while our biased estimator (Algorithm I) has no such condition. This is achieved at the expense of higher dependence on ϵ.

Our analysis assumes familiarity with techniques from randomized algorithms: first, second and exponential moments methods, in the form of the Markov inequality, Chebyshev inequality, and Chernoff bounds [MR95].

3. Upper bounds

In this section, we provide our two upper bounds in the form of randomize algorithms which succeed with constant probability. The first provides a simple sampling-based unbiased estimator, which has a low dependence on ϵ, but scales with $T^{-1/3}$. The second uses a similar sampling procedure, and provides a biased estimator, whose dependence is improved to $T^{-1/2}$, but with higher cost as a function of ϵ.

Algorithm I (unbiased estimator).
Let $p \in (0, 1]$. The value of p is determined by the analysis.
In the first pass, the algorithm stores each edge independently at random with probability p. Let $G' = (V, E')$ be the sampled subgraph.
In the second pass, the algorithm, upon reading the edge $e_i \notin E'$, counts the number of new triangles in $(V, E' \cup \{e_i\})$ and adds it to a global counter s. At the end of the second pass, the algorithm outputs $Y = \frac{s}{3p^2(1-p)}$ as the estimate for $t(G)$.

Theorem 1 *Algorithm I is a 2-pass randomized streaming algorithm for* $\mathsf{Estimate}(T, \epsilon)$ *that uses* $O(\frac{m}{\epsilon^{4/3}} \frac{\sqrt{\log n}}{T^{1/3}})$ *space.*

PROOF: Let $\mathcal{T}$ represent the set of triangles in the graph. For the analysis, we partition $\mathcal{T}$ into several groups through the following process. Fix an $L \in [1, t]$ (determined below). Pick an arbitrary edge $e \in E$ with at least L triangles on it. We notionally assign the triangles on e to the edge e. Let this be the set $\mathcal{T}_e \subseteq \mathcal{T}$. Continue this process until all the remaining edges participate in fewer than L unassigned triangles. Let $\mathcal{T}'$ be the unassigned triangles. Note that either of $\mathcal{T}_e$ and $\mathcal{T}'$ may be empty.

Let X_i be the indicator random variable associated with the i-th triangle in $\mathcal{T}$. We have $X_i = 1$ with probability $3p^2(1-p)$. For each edge e, let $s_e = \sum_{i \in \mathcal{T}_e} X_i$ and define $s_{\mathcal{T}'} = \sum_{i \in \mathcal{T}'} X_i$. We have $s = \sum_{e \in E} s_e + s_{\mathcal{T}'}$ and the expectation of s is $\mathsf{E}(s) = 3p^2(1-p)t$.

First we analyse the concentration of $s_{\mathcal{T}'}$. We have $\mathsf{E}(s_{\mathcal{T}'}) = 3p^2(1-p)|\mathcal{T}'|$. We also compute

$$\begin{aligned}\mathsf{Var}(s_{\mathcal{T}'}) &= \mathsf{E}(s_{\mathcal{T}'}^2) - \mathsf{E}^2(s_{\mathcal{T}'}) \\ &\leq \sum_{i \in \mathcal{T}'} \mathsf{E}(X_i^2) + \sum_{i \in \mathcal{T}' \neq j \in \mathcal{T}'} \mathsf{E}(X_i X_j) - \mathsf{E}^2(s_{\mathcal{T}'}) \\ &\leq 3p^2(1-p)|\mathcal{T}'| + (4p^3(1-p)^2 + p^4(1-p))|\mathcal{T}'|L.\end{aligned}$$

The final term derives from considering pairs of triangles i, j. We break these into those which share an edge, and those which are disjoint. For those sharing an edge, both are sampled if either (a) the shared edge and exactly one other edge in each triangle is sampled, with total probability $4p^3(1-p)^2$ or (b) if all edges except the shared edge are sampled, which occurs with probability $p^4(1-p)$. There are at most $|\mathcal{T}'|L$ such triangle pairs. For pairs of triangles which do not share any edge, their contribution to the sum is outweighted by the term $-\mathsf{E}(s_{\mathcal{T}'})^2$.

Since $(1-p) < 1$ and $p < 1$ we simplify this expression to $\mathsf{Var}(s_{\mathcal{T}'}) < 3p^2|\mathcal{T}'| + 5p^3|\mathcal{T}'|L$. By the Chebyshev inequality,

$$\Pr[|s_{\mathcal{T}'} - \mathsf{E}(s_{\mathcal{T}'})| \geq \epsilon p^2 t] \leq \frac{\mathsf{Var}(s_{\mathcal{T}'})}{\epsilon^2 p^4 t^2} \leq \frac{3|\mathcal{T}'|}{\epsilon^2 p^2 t^2} + \frac{5|\mathcal{T}'|L}{\epsilon^2 p t^2} \tag{1}$$

To bound the deviation of each s_e, we use the Chernoff bound. Let Z_e be the event corresponding to $e \notin E'$. Since the edges are sampled independently, conditioned on Z_e, the random variables $\{X_i\}_{i \in \mathcal{T}_e}$ are independent. Moreover we have $\mathsf{E}(X_i|Z_e) = p^2$. From the Chernoff bound, we get

$$\Pr[|s_e - \mathsf{E}(s_e)| \geq \epsilon \mathsf{E}(s_e) \mid Z_e] \leq e^{-\frac{p^2|\mathcal{T}_e|\epsilon^2}{2}} \leq e^{-\frac{p^2 L \epsilon^2}{2}} \tag{2}$$

Similarly, conditioned on $\overline{Z_e}$, the random variables $\{X_i\}_{i \in \mathcal{T}_e}$ are independent and $\mathsf{E}(X_i|\bar{Z}_e) = 2p(1-p)$.

$$\Pr[|s_e - \mathsf{E}(s_e)| \geq \epsilon \mathsf{E}(s_e) \mid \overline{Z_e}] \leq e^{-p(1-p)|\mathcal{T}_e|\epsilon^2} \leq e^{-p(1-p)L\epsilon^2} \tag{3}$$

From (2) and (3), for each $e \in E$, we get

$$\Pr[|s_e - \mathsf{E}(s_e)| \geq \epsilon \mathsf{E}(s_e)] \leq e^{-\frac{p^2 L \epsilon^2}{2}} \tag{4}$$

Therefore using the union bound and the fact that the number of edges with non-empty $\mathcal{T}_e$ is bounded by t/L, we get

$$\Pr\left[\left|\sum_e s_e - \sum_e \mathsf{E}(s_e)\right| \geq \epsilon \sum_e \mathsf{E}(s_e)\right] \leq \frac{t}{L} e^{-\frac{p^2 L \epsilon^2}{2}} \tag{5}$$

Since $t \geq T$ and setting $L = (\epsilon t)^{2/3}$ and $p = \Omega(\frac{1}{\epsilon^{4/3}} \frac{\sqrt{\log n}}{T^{1/3}})$ with large enough constants, the probabilities in (1) and (5) will be bounded by a small constant. The expected number of edges in the sampled graph G' is pm, and can be shown to be tightly concentrated around its expectation via a Chernoff bound on the Binomial tail probability, so the space usage is as stated above. This proves our theorem. □

The algorithm can be implemented quite efficiently. In the second pass, we can retrieve for each edge (i, j) the neighborhood of i and j stored in the first pass, and so compute the number of triangles complete by the edge. In the worst case, this takes $O(n)$ time for edges incident on high degree vertices, but other bounds can be given for this time cost, as a function of p and the degree of each node.

It is possible to extend this algorithm to the case where each edge may appear multiple times in the stream. Two modifications are needed to handle the fact that an edge may appear multiple times. The first is that the we must sample each edge with probability p, regardless of its multiplicity in the stream. This can be achieved by using a hash function to determine whether the edge is sampled: the edge is hashed to the range $[0 \ldots 1]$, and is sampled if the hash value is below p. The second is that we need to count the number of distinct triangles formed in the second pass. This can be done by using a distinct counting algorithm in place of the global counter, applied to the triangles as described by their three nodes [BYJK$^+$02].

We now modify this algorithm to work in the random order streaming model, where all permutations of the input are equally likely [GM09].

Corollary 2 *Assuming the data arrives in random order, there is a one-pass randomized streaming algorithm for* $\mathsf{Estimate}(T, \epsilon)$ *that uses* $O(\frac{m}{\epsilon^{4/3}} \frac{\sqrt{\log n}}{T^{1/3}})$ *space.*

PROOF: The one-pass algorithm collapses the two passes of Algorithm I into one. That is, the algorithm stores each edge into graph G' with probability p, and also counts the number of triangles completed in G' by each edge from the stream G.

The analysis follows the same outline as the main theorem, with some modification. First, we now have $\Pr[X_i = 1] = p^2(1-p)$, since the unsampled edge must be the last in the stream order, and $\mathsf{E}(s)$ is correspondingly lower by a factor of 3. Then $\mathsf{E}(X_i|Z_i) = p^2/3$, since to count triangle i, we must have that the first two edges are seen before edge e in the stream. Likewise, $\mathsf{E}(X_i|\bar{Z}_i) = 2p(1-p)/3$, since we must have the unsampled edge appear after the two sampled edges. This causes us to rescale p by a constant, which does not change the asymptotic cost of the algorithm. □

Note that the requirement of random order is important for the one-pass result. Because we split the analysis based on the particular edges, the order in which these edges appear can affect the outcome. If the edge e were to always appear after the two other edges in triangle i, then $\mathsf{E}(X_i|Z_e)$ would be 0. Hence, we need the edges to appear in random order to ensure this one-pass analysis holds.

Our next algorithm builds a similar estimator, but differs in some important ways.

Algorithm II (biased estimator).
Repeat the following $l \geq 16/\epsilon$ times independently in parallel and output the minimum of the outcomes.
In the first pass, pick every edge with probability p (the value of p is determined below.)
In the second pass, count the total number of triangles detected: both those where all three edges were sampled in the first pass, and where two edges were sampled in the first pass, and the completing edge observed in the second pass. Let r be the total number of triangles detected. Output $\frac{r}{3p^2(1-p)+p^3}$.

Theorem 3 *Algorithm II is a 2-pass randomized streaming algorithm for* $\mathsf{Estimate}(T, \epsilon)$ *that uses* $O(\frac{m}{\epsilon^{4.5}\sqrt{T}})$ *space.*

Proof: Let R be the output of Algorithm II. As in the previous proof, let $\mathcal{T}$ represent the set of triangles in the graph. Consider one instance of the basic estimator, and let X be the outcome of this instance. Let X_i denote the indicator random variable associated with the ith triangle in $\mathcal{T}$ being detected. By simple calculation, we have $\Pr[X_i = 1] = 3p^2(1-p) + p^3$ and $\mathsf{E}(X) = \frac{1}{3p^2(1-p)+p^3} \sum_{i \in \mathcal{T}} \mathsf{E}[X_i] = t$. Thus, X is an unbiased estimator for t; however, R, which is the minimum of l independent repetitions of X, is biased. By the Markov inequality, $\Pr[X \geq (1+\epsilon)\mathsf{E}(X)] \leq 1/(1+\epsilon)$. Therefore, picking $\epsilon \leq 1$, we can conclude,

$$\Pr[R \leq (1+\epsilon)t] \geq (1 - \Pr[X \geq (1+\epsilon)t]^{16/\epsilon}) \geq 1 - \left(\frac{1}{2}\right)^{16} \geq 1 - 10^{-4}.$$

However, proving a lower bound on R is more complex, and requires a more involved analysis. First, we show that most triangles share an edge with a limited number of triangles. More precisely, let $L \subseteq E$ denote the set of edges where each $e \in L$ belongs to at most $3\sqrt{t/\epsilon}$ triangles. We call L the set of *light* edges and $H = E \setminus L$ the *heavy* edges. We claim there exists $S \subseteq \mathcal{T}$ such that $|S| \geq (1-\epsilon)t$ and every triangle in S has at least two light edges. This is true because there can be at most $\frac{3t}{3\sqrt{t/\epsilon}} = \sqrt{\epsilon t}$ heavy edges, and moreover every two distinct edges belong to at most one triangle.

For each triangle $i \in S$, fix two of its light edges. Let Y_i denote the indicator random variable for the event where the algorithm picks these two light edges of $i \in S$ in the first pass. We have $\mathsf{E}(Y_i) = p^2$ and always $Y_i \leq X_i$. Therefore, finding a lower bound on Y will give a lower bound on X. We will argue that the probability of Y being less than $(1-\epsilon)|S|$ is small, even after taking the minimum of the l parallel repetitions. Let $Y = \frac{1}{p^2}\sum_{i\in S} Y_i$. We have

$$\mathsf{E}(Y) = |S| \geq (1-\epsilon)t.$$

We also have

$$\mathsf{Var}(Y) = \mathsf{E}(Y^2) - \mathsf{E}^2(Y) \leq \frac{1}{p^2}|S| + \frac{1}{p}|S|\sqrt{t/\epsilon}.$$

The first term comes from $\sum_{i\in S}\frac{1}{p^4}\mathsf{E}(Y_i^2)$, and the second term arises from pairs of triangles which share a light edge, of which there are at most $|S|\sqrt{t/\epsilon}$ (since the edge is light), and which are both sampled with probability p^3. Using the Chebyshev inequality and assuming $\epsilon < \frac{1}{2}$, we have

$$\begin{aligned}
\Pr[Y < (1-\epsilon)^2 t] &\leq \Pr[Y < (1-\epsilon)|S|] \\
&\leq \frac{\mathsf{Var}(Y)}{\epsilon^2|S|^2} \\
&\leq \frac{1}{\epsilon^2}\left(\frac{1}{p^2|S|} + \frac{\sqrt{t/\epsilon}}{p|S|}\right) \\
&< \frac{1}{\epsilon^2}\left(\frac{2}{p^2 t} + \frac{2}{p\sqrt{\epsilon t}}\right).
\end{aligned}$$

Since $T \leq t$, setting $p > \frac{320}{\epsilon^{3.5}\sqrt{T}}$, allows the above probability to be bounded by $\frac{\epsilon}{160}$. Now the probability that the minimum of $16/\epsilon$ independent trials is below the designated threshold is at most $\frac{\epsilon}{160}\frac{16}{\epsilon} = 1/10$. Therefore with probability at least $1-(1/10^{-4}+1/10)$ the output of the algorithm is within the interval $[(1-2\epsilon)t, (1+\epsilon)t]$. This proves the statement of our theorem. □

Asymptotically, the running time of this algorithm is the same as Algorithm I, since in both the dominating cost is that of finding the number of triangles completed by an edge seen in the second pass. It can also be extended to handle multiple occurrences of edges in the same manner as Algorithm I, by using hash based sampling and approximate distinct counting.

Corollary 4 *Assuming the data arrives in random order, there is a one-pass randomized streaming algorithm for* $\mathsf{Estimate}(T,\epsilon)$ *that uses* $O(\frac{m}{\epsilon^{4.5}\sqrt{T}})$ *space.*

PROOF: Under random order, we can combine the first and second passes of algorithm II. We count all triangles formed as r: either those with all three edges sampled, or those with two edges sampled and the third observed subsequently in the stream. The estimator is now $\frac{r}{p^2}$, since the probability of counting any triangle is p^3 (for all three edges sampled) plus $p^2(1-p)$ (for the first two edges in the stream sampled, and the third unsampled). The same analysis as for Theorem 3 then follows: we partition the edges in to light and heavy sets, and bound the probability of sampling a subset of triangles. A triangle with two light edges is counted if both light edges are sampled, and the heavy edge arrives last. This happens with probability $p^2/3$. We can nevertheless argue that we are unlikely to undercount such triangles, following the same Chebyshev analysis as above. This allows us to conclude that the estimator is good. $\Box$

Again, random order is critical to make this algorithm work in one pass: an adversarial order could arrange the heavy edges to always come last (increasing the probability of counting a triangle under this analysis) or always first (giving zero probability of counting a triangle under this analysis). It remains an open question to understand whether these bounds can be obtained in a single pass without the random order assumption.

4. Lower bounds

We now show lower bounds for the problem $\mathsf{Dist}(T)$, to distinguish between the case $t = 0$ and $t \geq T$. Our first result builds upon a lower bound from prior work, and amplifies the hardness. We formally state the previous result:

Lemma 5 *[BOV13] Every constant pass streaming algorithm for* $\mathsf{Dist}(T)$ *requires* $\Omega(\frac{m}{T})$ *space.*

Theorem 6 *Any constant pass streaming algorithm for* $\mathsf{Dist}(T)$ *requires* $\Omega(\frac{m}{T^{2/3}})$ *space.*

PROOF: Given a graph $G = (V, E)$ with m edges we can create a graph $G' = (V', E')$ with mT^2 edges and $t(G') = T^3 t(G)$. We do so by replacing each vertex $v \in V$ with T vertices $\{v_1, \ldots, v_T\}$ and replacing the edge $(u, v) \in E$ with the edge set $\{u_1, \ldots, u_T\} \times \{v_1, \ldots, v_T\}$. Clearly any triangle in G will be replaced by T^3 triangles in G' and every triangle in G' corresponds to a triangle in G. Moreover this reduction can be peformed in a streaming fashion using $O(1)$ space. Therefore a streaming algorithm for $\mathsf{Dist}(T)$ using $o(\frac{m}{T^{2/3}})$ (applied to G') would imply an $o(m)$ streaming algorithm for $\mathsf{Dist}(1)$. But from Lemma 5, we have that $\mathsf{Dist}(1)$ requires $\Omega(m)$ space for constant pass algorithms. This is a contradiction and as a result our claim is proved. □

Our next lower bound more directly shows the hardness by a reduction to the hard communication problem of Disj_p^r for the case of up to a quadratic number of triangles.

Theorem 7 *For any $\rho > 0$ and $T \leq n^2$, there is no constant pass streaming algorithm for $\mathsf{Dist}(T)$ that takes $O(\frac{m}{T^{1/2+\rho}})$ space.*

PROOF: We show that there are families of graphs with $\Theta(n\sqrt{T})$ edges and T triangles such that distinguishing them from triangle-free graphs in a constant number of passes requires $\Omega(n)$ space. This is enough to prove our theorem.

We use a reduction from a standard set intersection problem, here denoted by $\mathsf{Disj}_n^{n/4}$. Given $y \in \{0,1\}^n$, Bob constructs a bipartite graph $G = (A \cup B, E)$ where $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_{\sqrt{T}}\}$. He connects a_i to all vertices in B iff $y[i] = 1$. On the other hand, Alice adds vertices $C = \{c_1, \ldots, c_{\sqrt{T}}\}$ to G. She adds the edge set $C \times B$. Also for each $i \in [\sqrt{T}]$ and $j \in [n]$, she adds the edge (c_i, a_j) iff $x[j] = 1$. We observe that if x and y (uniquely) intersect there will be precisely T triangles passing through each vertex of C. Since there is no edge between the vertices in C, in total we will have T triangles. On the other hand, if x and y represent disjoint sets, there will be no triangles in G. In both cases, the number of edges is between $2n\sqrt{T}$ and $3n\sqrt{T}$, over $O(n)$ vertices (using the bound $T^2 \leq n$). Considering the lower bound for the Disj_p^r (Section 2), our claim is proved following a standard argument: a space efficient streaming algorithm would imply an efficient communication protocol between Alice and Bob whose messages are the memory state of the algorithm executed on their respective portions of the input (transformed into graph form). Since any such protocol requires $\Omega(n)$ bits of communication, this shows the claimed $\Omega(n)$ space lower bound. □

Acknowledgments. We thank Andrew McGregor, Srikanta Tirthapura and Vladimir Braverman for several helpful conversations.

[BFL$^+$06] Luciana S. Buriol, Gereon Frahling, Stefano Leonardi, Alberto Marchetti-Spaccamela, and Christian Sohler, *Counting triangles in data streams*, PODS, 2006, pp. 253–262.

[BOV13] Vladimir Braverman, Rafail Ostrovsky, and Dan Vilenchik, *How hard is counting triangles in the streaming model?*, ICALP (1), 2013, pp. 244–254.

[BYJK$^+$02] Z. Bar-Yossef, T.S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisian, *Counting distinct elements in a data stream*, Proceedings of RANDOM 2002, 2002, pp. 1–10.

[BYKS02] Z. Bar-Yossef, R. Kumar, and D. Sivakumar, *Reductions in streaming algorithms, with an application to counting triangles in graphs*, ACM-SIAM Symposium on Discrete Algorithms, 2002, pp. 623–632.

[GM09] Sudipto Guha and Andrew McGregor, *Stream order and order statistics: Quantile estimation in random-order streams*, SIAM J. Comput. **38** (2009), no. 5, 2044–2059.

[HTC13] Xiaocheng Hu, Yufei Tao, and Chin-Wan Chung, *Massive graph triangulation*, Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, 2013, pp. 325–336.

[JG05] Hossein Jowhari and Mohammad Ghodsi, *New streaming algorithms for counting triangles in graphs*, COCOON, 2005, pp. 710–716.

[JSP13] Madhav Jha, C. Seshadhri, and Ali Pinar, *A space efficient streaming algorithm for triangle counting using the birthday paradox*, KDD, 2013, pp. 589–597.

[KN97] E. Kushilevitz and N. Nisan, *Communication complexity*, Cambridge University Press, 1997.

[LBKT08] Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins, *Microscopic evolution of social networks*, KDD, 2008, pp. 462–470.

[LWZW10] Zhiyu Liu, Chen Wang, Qiong Zou, and Huayong Wang, *Clustering coefficient queries on massive dynamic social networks*, WAIM, 2010, pp. 115–126.

[MR95] R. Motwani and P. Raghavan, *Randomized algorithms*, Cambridge University Press, 1995.

[PT12] Rasmus Pagh and Charalampos E. Tsourakakis, *Colorful triangle counting and a mapreduce implementation*, Inf. Process. Lett. **112** (2012), no. 7, 277–281.

[PTTW13] A. Pavan, Kanat Tangwongsan, Srikanta Tirthapura, and Kun-Lung Wu, *Counting and sampling triangles from a graph stream*, PVLDB, 2013.

[SV11] Siddharth Suri and Sergei Vassilvitskii, *Counting triangles and the curse of the last reducer*, WWW, 2011, pp. 607–614.

[SW05] Thomas Schank and Dorothea Wagner, *Finding, counting and listing all triangles in large graphs, an experimental study*, WEA, 2005, pp. 606–609.

[TKMF09] Charalampos E. Tsourakakis, U. Kang, Gary L. Miller, and Christos Faloutsos, *Doulion: counting triangles in massive graphs with a coin*, KDD, 2009, pp. 837–846.

[Tso08] Charalampos E. Tsourakakis, *Fast counting of triangles in large real networks without counting: Algorithms and laws*, ICDM, 2008, pp. 608–617.