Tiered Sampling: An Efficient Method for Approximate Counting Sparse Motifs in Massive Graph Streams

Lorenzo De Stefani
Department of Computer Science
Brown University
Providence, RI, USA
lorenzo@cs.brown.edu

Erisa Terolli
Department of Computer Science
Sapienza University of Rome
Rome, Italy
terolli@di.uniroma1.it

Eli Upfal
Department of Computer Science
Brown University
Providence, RI, USA
eli@cs.brown.edu

***Abstract*—We introduce Tiered Sampling, a novel technique for approximate counting sparse motifs in massive graphs whose edges are observed in a stream. Our technique requires only a single pass on the data and uses a memory of fixed size M, which can be magnitudes smaller than the number of edges.**

Our methods addresses the challenging task of counting sparse motifs - sub-graph patterns that have low probability to appear in a sample of M edges in the graph, which is the maximum amount of data available to the algorithms in each step. To obtain an unbiased and low variance estimate of the count we partition the available memory to tiers (layers) of reservoir samples. While the base layer is a standard reservoir sample of edges, other layers are reservoir samples of sub-structures of the desired motif. By storing more frequent sub-structures of the motif, we increase the probability of detecting an occurrence of the sparse motif we are counting, thus decreasing the variance and error of the estimate.

We demonstrate the advantage of our method in the specific applications of counting sparse 4 and 5-cliques in massive graphs. We present a complete analytical analysis and extensive experimental results using both synthetic and real-world data. Our results demonstrate the advantage of our method in obtaining high-quality approximations for the number of 4 and 5-cliques for large graphs using a very limited amount of memory, significantly outperforming the single edge sample approach for counting sparse motifs in large scale graphs.

***Keywords*-graph motif mining; reservoir sampling; stream computing;**

I. Introduction

Counting motifs (sub-graphs with a given pattern) in large graphs is a fundamental primitive in graph mining with numerous practical applications including link prediction and recommendation, community detection [1], topic mining [2], spam and anomaly detection [3, 4] and protein interaction networks analysis [5].

Computing exact count of motifs in massive, Web-scale networks is often impractical or even infeasible. Furthermore, many interesting networks, such as social networks, are continuously growing, hence there is a limited value in maintaining an exact count. The goal is rather to have, *at any time*, a high-quality approximation of the quantity of interest.

To obtain a scalable and efficient solution for massive size graphs we focus here on the well studied model of one-pass stream computing. Our algorithms use a memory of fixed size M, where M is significantly smaller that the size of the input graph. The input is given as a *stream* of edges in an *arbitrary* order, and the algorithm has only one pass on the input. The goal of the algorithm is to compute at any given time an unbiased, low variance estimate of the count of motif occurrences in the graph seen up to that time.

Given its theoretical and practical importance, the problem of counting motifs in graph streams has received great attention in the literature, with particular emphasis on the approximation of the number of 3-cliques (triangles) [6–8]. A standard approach to this problem is to sample up to M edges uniformly at random, using a fixed sampling probability or, more efficiently, reservoir sampling. A count of the number of motifs in the sample, extrapolated (normalized) appropriately, gives an unbiased estimate for the number of occurrences in the entire graph. The variance (and error) of this method depends on the expected number of occurrences in the sample. In particular, for sparse motifs that are unlikely to appear many times in the sample, this method exhibits high variance (larger than the actual count) which makes it useless for counting sparse motifs. Note that when the input graph is significantly larger than the memory size M, a motif that is unlikely to appear in a random sample of M edges may still have a large count in the graph. Also, as we attempt to count larger structure than triangles, these structures are more likely to be sparse in the graph. It is therefore important to obtain efficient methods for counting *sparser* motifs in massive scale graph streams.

In this work we introduce the concept of Tiered Sampling in stream computing. To obtain an unbiased and low variance estimate for the amount of sparse motif in massive scale graph we partition the available memory to tiers (layers) of reservoir samples. The base

tier is a standard reservoir sample of individual edges, other tiers are reservoir samples of sub-structures of the desired motif. This strategy significantly improves the probability of detecting occurrences of the motif.

Assume that we count motifs with k edges. If all the available memory is used to store a sample of the edges, we would need $k-1$ of the motif's edges to be in the sample when the last edge of the motif is observed on the stream. The probability of this event decreases exponentially in k.

Assume now that we use part of the available memory to store a sample of the observed occurrences of a fixed sub-motifs with $k/2$ edges. We are more likely to observe such motifs (we only need $k/2-1$ of there edges to be in the edge sample when the last edge is observed in the stream), and they are more likely to stay in the second reservoir sample since they are still relatively sparse. We now observe a full motif when the current edge in the stream completes an occurrence of the motif with edges sub-motifs in the two reservoirs. For an appropriate choice of parameters, and with fixed total memory size, this event has significantly higher probability than observing the $k-1$ edges in the edge sample. To obtain an unbiased estimate of the count, the number of observed occurrences needs to be carefully normalized by the probabilities of observing each of the components.

In this work we make the following main contributions:

- We introduce the concept of *Tiered Sampling* for counting sparse concepts in large scale graph stream using multi-layer reservoir samples.
- We develop and fully analyze two algorithms for counting the number of 4-cliques in a graph using two tier reservoir sampling.
- For comparison purpose we analyze a standard (one tier) reservoir sample algorithm for 4-cliques counting problem.
- We verify the advantage of our multi-layer algorithms by analytically comparing their performance to a standard (one tier) reservoir sample algorithm for 4-cliques counting problem on random Barábasi-Albert graphs.
- We develop the ATS4C technique, which allows to adaptively adjust the sub-division of memory space among the two tiers according to the properties of the graph being considered.
- We conduct an extensive experimental evaluation of the 4-clique algorithms on massive graphs with up to hundreds of millions edges. We show the quality of the achieved estimations by comparing them with the actual ground truth value. Our algorithms are also extremely scalable, showing update times in the order of hundreds of microseconds for graphs with billions of edges.
- We demonstrate the generality of our approach through a second application of the two-tier method, estimating the number of 5-cliques in a graph stream using a second reservoir sample of the 4-cliques observed on the stream.

To the best of our knowledge these are the first fully analyzed, one pass stream algorithms for the 4 and 5-cliques counting problem.

II. Preliminaries

For any (discrete) time step $t \geq 0$, we denote the graph observed up to and including time t as $G^{(t)} = (V^{(t)}, E^{(t)})$, where $V^{(t)}$ (resp., $E^{(t)}$) denotes the set of vertices (resp., edges) of $G^{(t)}$. At time $t = 0$ we have $V^{(t)} = E^{(t)} = \emptyset$. For any $t > 0$, at time $t+1$ we receive one single edge $e_{t+1} = (u, v)$ from a stream, where u, v are two distinct vertices. $G^{(t+1)}$ is thus obtained by *inserting the new edge*: $E^{(t+1)} = E^{(t)} \cup \{(u, v)\}$; if either u or v do not belong to $V^{(t)}$, they are added to $V^{(t+1)}$. Edges can be added just once (i.e., we do not consider *multigraphs* in this work) in an arbitrary adversarial order, i.e., as to cause the worst outcome for the algorithm. We however assume that the adversary has *no access to the random bits* used by the algorithm.

This work explores the idea of storing a sample of sub-motifs in order to enhance the count of a sparse motif. For concreteness we focus on estimating the counts of 4-cliques and 5-cliques.

Given a graph $G^{(t)} = (V^{(t)}, E^{(t)})$, a *$k$-clique* in $G^{(t)}$ is a *set* of $\binom{k}{2}$ (distinct) edges connecting a set of k (distinct) vertices. We denote by $\mathcal{C}_k^{(t)}$ the set of *all* k-cliques in $G^{(t)}$.

Our work makes use of the *reservoir sampling scheme* [9]. Consider a stream of elements e_i observed in discretized time steps. Given a fixed sample size $M > 0$, for any time step t the reservoir sampling scheme allows to maintain a uniform sample $\mathcal{S}$ of size $\min\{M, t\}$ of the t elements observed on the stream:

- If $t \leq M$, then the element $e_t = (u, v)$ on the stream at time t is deterministically inserted in $\mathcal{S}$.
- If $t > M$, then the sampling mechanism flips a biased coin with heads probability M/t. If the outcome is heads, it chooses an element e_i uniformly at random from those currently in $\mathcal{S}$ which is replaced by e_t. Otherwise, $\mathcal{S}$ is not modified.

When using reservoir sampling for estimating the subgraph count it is necessary to compute the probability of multiple elements being in $\mathcal{S}$ at the same time.

Lemma II.1 (Lemma 4.1 [6])**.** *For any time step t and any positive integer $k \leq t$, let B be any subset of size $|B| = k \leq \min\{M, t\}$ of the element observed on the stream. Then,* at the end *of time step t (i.e., after*

updating the sample at time t), we have $\Pr(B \subseteq \mathcal{S}) = 1$ *if* $t \leq M$*, and* $\Pr(B \subseteq \mathcal{S}) = \prod_{i=0}^{k-1} \frac{M-i}{t-i}$ *otherwise.*

In our experimental analysis (Sections V-C and VII) we measure the accuracy of the obtained estimator *through the evolution of the graph* in terms of their *Mean Average Percentage Error* (MAPE) [10]. The MAPE measures the relative error of an estimator (in this case, $\varkappa^{(t)}$) with respect to the ground truth (in this case, $|\mathcal{C}_k^{(t)}|$) averaged over t time steps: that is $MAPE = \frac{1}{t}\sum_{i=1}^{t} \frac{|\varkappa^{(t)} - |\mathcal{C}_k^{(t)}||}{|\mathcal{C}_k^{(t)}|}$.

III. Related Work

Counting subgraphs in large networks is a well studied problem in data mining which was originally brought to attention in the seminal work of Milo et al. [5]. In particular, many contributions in the literature have focused on the triangle counting problem, including exact algorithms, MapReduce algorithms [11, 12] and streaming algorithms [6, 7, 13, 14].

Previous works in literature on counting graph motifs [15, 16] can also be used to estimate the number of cliques in large graphs. Other recent works on clique counting introduced randomized [17] and MapReduce [18] algorithms. These require however priori information on the graph such as its degeneracy (for [17]) or the vertex degree ordering (for [18]). Further, none of these approaches can be used in the streaming setting.

The idea of using sub-structures of a graph motif in order to improve the estimation of its frequency in a massive graph has been previously explored in literature. In [19] Bordino et al. proposed a data stream algorithm which estimates the number of occurrences of a given subgraph by sampling its "*prototypes* (i.e., sub-structures). While this approach is shown to be effective in estimating the counts of motif with three and four edges, it requires multiple passes through the graph stream and further knowledge on the properties of the graph. In [20], Jha et al. proposed an algorithm which effectively and efficiently approximates the frequencies of all 4-vertex subgraphs by sampling paths of length three. This algorithm requires however knowledge of the degrees of all the vertices in the graph and cannot be used in the streaming setting.

To the best of our knowledge, our work is the first that proposes a sampling-based, one pass algorithm for insertion only streams to approximate the global number of cliques found in large graphs. Further, our algorithms do not require any further information on the properties of the graph being observed.

Using a strategy similar to our TieredSampling approach, in [14] Jha and Seshadri propose a one pass streaming algorithm for triangle counting which using a first reservoir for edges which are then used to generate a stream of *wedges* (i.e., paths of length two) stored in a second reservoir. This approach appear to be not worthwhile for triangle counting as it is consistently outperformed by a simpler strategy based on a single reservoir presented [6]. This is due to the fact that as in most large graph of interest wedges are much more frequent than edges themselves it is not worth devoting a large fraction of the available memory space to maintaining wedges over edges.

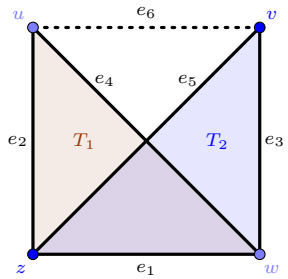


Figure 1. Detection of 4-clique using triangles

IV. TieredSampling application to 4-clique counting

In this section we present TS4C$_1$ and TS4C$_2$, two applications of our TieredSampling approach for counting the number of 4-cliques in an undirected graph observed as an edge stream. The two algorithms partition the available memory into two samples, an edge reservoir sample and a triangle reservoir sample. TS4C$_1$ attempts in each step to construct a 4-cliques using the current observed edge, two edges from the edge reservoir sample and one triangle from the triangle reservoir sample. TS4C$_2$ attempts at each step to construct a 4-clique from the current observed edge and two triangles from the triangle reservoir sample. At each time step t, both algorithms maintain a *running estimation* $\varkappa^{(t)}$ of $|\mathcal{C}_4^{(t)}|$. Clearly $\varkappa^{(0)} = 0$ and the estimator is increased every time a 4-clique is "*observed*" on the stream. The two algorithms also maintains a counter $\tau^{(t)}$ for the number of triangles observed in the stream up to time t. This value is used by the reservoir sampling scheme which manages the triangle reservoir.

A. Algorithm TS4C$_1$ description

Algorithm TS4C$_1$ maintains an *edges (resp., triangles) reservoir sample* $\mathcal{S}_e$ (resp., $\mathcal{S}_\Delta$) of *fixed* size M_e (resp., M_Δ). From Lemma II.1, for any t the probability of any edge e (resp., traingle T) observed on the stream Σ (resp., observed by the TS4C$_1$) to be included in S_e (resp., S_Δ) is M_e/t (resp. $M_\Delta/\tau^{(t)}$). We denote as $\mathcal{S}_e^{(t)}$ (resp., $\mathcal{S}_\Delta^{(t)}$) the set of edges (resp., triangles) in $\mathcal{S}_e$ (resp., $\mathcal{S}_\Delta$) *before* any update to the sample(s) occurring at step t. We denote with $\mathcal{N}_u^{\mathcal{S}_e^{(t)}}$ the *neighborhood* of u

ALGORITHM 1 TS4C$_1$ - Tiered Sampling for 4-Clique counting

Input: Insertion-only edge stream Σ, integers M, M_Δ
$\mathcal{S}_e \leftarrow \emptyset$, $\mathcal{S}_\Delta \leftarrow \emptyset$, $t \leftarrow 0$, $t_\Delta \leftarrow 0$, $\sigma \leftarrow 0$
for each element (u, v) from Σ **do**
  $t \leftarrow t + 1$
  UPDATE4CLIQUES(u, v)
  UPDATETRIANGLES(u, v)
  if SAMPLEEDGE$((u, v), t)$ **then**
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{((u, v), t)\}$

function UPDATE4CLIQUES$((u, v), t)$
  for each triangle $(u, w, z) \in \mathcal{S}_\Delta$ **do**
    if $(v, w) \in \mathcal{S}_e \wedge (v, w) \in \mathcal{S}_e$ **then**
      $p \leftarrow$ PROBCLIQUE$((u, w, z), (v, w), (v, z))$
      $\sigma \leftarrow \sigma + p^{-1}/2$
  for each triangle $(v, w, z) \in \mathcal{S}_\Delta$ **do**
    if $(u, w) \in \mathcal{S}_e \wedge (u, z) \in \mathcal{S}_e$ **then**
      $p \leftarrow$ PROBCLIQUE$((v, w, z), (u, w), (u, z))$
      $\sigma \leftarrow \sigma + p^{-1}/2$

function UPDATETRIANGLES$((u, v), t)$
  $\mathcal{N}^{\mathcal{S}}_{u,v} \leftarrow \mathcal{N}^{\mathcal{S}}_u \cap \mathcal{N}^{\mathcal{S}}_v$
  for each element w from $\mathcal{N}^{\mathcal{S}}_{u,v}$ **do**
    $t_\Delta \leftarrow t_\Delta + 1$
    if SAMPLETRIANGLE(u, v, w) **then**
      $\mathcal{S}_\Delta \leftarrow \mathcal{S}_\Delta \cup \{u, v, w\}$

function SAMPLETRIANGLE(u, v, w)
  if $t_\Delta \leq M_\Delta$ **then**
    return True
  else if FLIPBIASEDCOIN$(\frac{M_\Delta}{t_\Delta}) =$ heads **then**
    $(u_1, v_1, w_1) \leftarrow$ random triangle from $\mathcal{S}_\Delta$
    $\mathcal{S}_\Delta \leftarrow \mathcal{S}_\Delta \setminus \{(u_1, v_1, w_1)\}$
    return True
  return False

function SAMPLEEDGE$((u, v), t)$
  if $t \leq M$ **then**
    return True
  else if FLIPBIASEDCOIN$(\frac{M}{t}) =$ heads **then**
    $((u', v'), t') \leftarrow$ random edge from $\mathcal{S}$
    $\mathcal{S} \leftarrow \mathcal{S} \setminus \{((u', v'), t')\}$
    return True
  return False

with respect to the edges in $\mathcal{S}_e^{(t)}$, that is $\mathcal{N}_u^{\mathcal{S}_e^{(t)}} = \{v \in V^{\mathcal{S}_e^{(t)}} : (u, v) \in \mathcal{S}_e^{(t)}\}$.

Let $e_t = (u, v)$ be the edge observed on the stream at time t. At each step TS4C$_1$ executes three main tasks:

- *Estimation update:* TS4C$_1$ invokes the function UPDATE 4-CLIQUES to detect any 4-clique completed by (u, v) (see Figure 1 for an example). That is, the algorithm verifies whether in triangle reservoir $\mathcal{S}_\Delta$ there exists any triangle T which includes u (resp., v). Note that since each edge is observed just once and the edge (u, v) is being observed for the first time no triangle in $\mathcal{S}_\Delta$ can include both u and v. For any such triangle $T' = \{u, w, z\}$ (or $\{v, w, z\}$) the algorithm checks whether the edges (v, w) and (v, z) (resp., (v, w) and (v, z)) are currently in $\mathcal{S}_e$. When such conditions are meet, we say that a 4-clique is "*observed on the stream*". The algorithm then uses PROBCLIQUE to compute the *exact* probability p of the observation based on the timestamps of all its edges. The estimator $\varkappa$ is then increased by $p^{-1}/2$.
- *Triangle sample update:* using UPDATETRIANGLES the algorithm verifies whether the edge e_t completes any triangle with the edges in $\mathcal{S}_e^{(t)}$. If that is the case, we say that a new triangle T^* is *observed on the stream*. The counter τ is increased by one and the new triangle is a candidate for inclusion in $\mathcal{S}_\Delta$ with probability M_Δ/τ^t.
- *Edge sample update:* the algorithm updates the edge sample $\mathcal{S}_e$ according to the Reservoir Sampling scheme described in Section II.

Each time a 4-clique is observed on the stream, TS4C$_1$ uses PROBCLIQUE to compute the *exact probability* of the observation. Such computation is all but trivial as it is influenced by both the order according to which the edges of the 4-clique were observed on the stream, and by the number of triangles observed on the stream $\tau^{(t)}$. The analysis proceeds using a (somehow tedious) analysis of *all* the 5! possible orderings of the first five edges of the clique observed on the stream. Due to space limitations we refer the reader to Lemma A.2 of the extend version of this work [21] for the detailed computation.

B. Analysis of the estimator

We now present the analysis of the estimations obtained using TS4C$_1$. First we show their *unbiasedness* and we then provide a bound on their variance. We refer the reader to Appendix A of the extended version of this work for the complete proofs [21]. In the following we denote as t_Δ the first time step at for which the number of triangles seen by TS4C$_1$ exceeds M_Δ.

Theorem IV.1. *$\varkappa^{(t)} = |\mathcal{C}_k^{(t)}|$ if $t \leq \min\{M_e, t_\Delta\}$, and $\mathbb{E}\left[\varkappa^{(t)}\right] = |\mathcal{C}_k^{(t)}|$ otherwise.*

We now show an *upper bound* to the variance of the TS4C$_1$ estimations. The proof relies on a very careful analysis of the order of arrival of the edge shared between a pair of two 4-cliques. We present here the most general result for $t \geq M_e, t_\Delta$. Note that if $t \leq \min\{M_e, t_\Delta\}$, from Theorem IV.1, $\varkappa^{(t)} = |\mathcal{C}_k^{(t)}|$ and hence $\mathrm{Var}\left[\varkappa^{(t)}\right] = 0$. For $\min\{M_e, t_\Delta\} < t \leq \max\{M_e, t_\Delta\}$, $\mathrm{Var}\left[\varkappa^{(t)}\right]$ admits an upper bound similar to the one in Theorem IV.2.

Theorem IV.2. *For any time $t > \min\{M_e, t_\Delta\}$, we have*

$$\begin{aligned} Var\left[\varkappa^{(t)}\right] \leq |\mathcal{C}_4^{(t)}| &\left(c\left(\frac{t-1}{M_e}\right)^4 \left(\frac{\tau^{(t)}}{M_\Delta}\right) - 1 \right) \\ &+ 2a^{(t)} \left(c\frac{t-1}{M_e} - 1 \right) \\ &+ 2b^{(t)} \left(c\left(\frac{t-1}{M_e}\right)^2 \left(\frac{1}{4}\frac{\tau^{(t)}}{M_\Delta} + \frac{3}{4}\frac{t-1}{M_e}\right) - 1 \right), \end{aligned}$$

where $a^{(t)}$ (resp., $b^{(t)}$) denotes the number of unordered pairs of 4-cliques which share one edge (resp., three edges) in $G^{(t)}$, and $c \geq \frac{M_e^3}{(M_e-1)(M_e-2)(M_e-3)}$.

ALGORITHM 2 TS4C$_2$ - Tiered Sampling for 4-Clique counting using 2 triangle sub-structures

```
function UPDATE4CLIQUES((u, v), t)
    for each (u, w, z) ∈ S_Δ ∧ (v, w, z) ∈ S_Δ do
        p ← PROBCLIQUE((u, w, z), (v, w, z))
        σ ← σ + p^{-1}/2
```

C. Memory partition across layers

In most practical scenario we assume that a certain amount of total available memory M is available for algorithm TS4C$_1$. A natural question concern what is the best way of spitting the available memory between $\mathcal{S}_e$ and $\mathcal{S}_\Delta$. While different heuristics are possible, in our work we chose to assign the available space in such a way that the dominant first term of upper bound of TS4C$_1$ in Theorem IV.2 is minimized. For $0 < \alpha < 1$ let $M_e = \alpha M$ and $M_\Delta = (1-\alpha)M$. Then we have that $|\mathcal{C}_4^{(t)}|\left(c\left(\frac{t-1}{\alpha M}\right)^4\left(\frac{\tau^{(t)}-1}{(1-\alpha)M}\right)-1\right)$ is minimized for $\alpha = 4/5$. This convenient splitting rule works well in most cases, and is used in most of the paper. Section VI we discuss a more sophisticated dynamic allocation of memory, ATS4C, and present experimental results for this method in Section VII-B.

D. Concentration bound

We now show a concentration result on the estimation of TS4C$_1$, which relies on Chebyshev's inequality [22, Thm. 3.6].

Theorem IV.3. *Let $t > \min\{M_e, t_\Delta\}$ and assume $|\mathcal{C}_4^{(t)}| > 0$. For any $\varepsilon, \delta \in (0,1)$, if*

$$M > \max\Big\{\frac{5}{4}\sqrt[5]{\frac{12c(t-1)^4(\tau^{(t)})}{\delta\varepsilon^2|\mathcal{C}_4^{(t)}|}}, \frac{15ca^{(t)}(t-1)}{\delta\varepsilon^2|\mathcal{C}_4^{(t)}|^2}, \frac{5}{4}\sqrt[3]{\frac{3b^{(t)}c\,(t-1)^2\left(4\tau^{(t)}+3(t-1)\right)}{2\delta\varepsilon^2|\mathcal{C}_4^{(t)}|^2}}\Big\}$$

then $|\varkappa^{(t)} - |\mathcal{C}_4^{(t)}| < \varepsilon|\mathcal{C}_4^{(t)}|$ with probability $> 1-\delta$.

E. Algorithm TS4C$_2$

TS4C$_2$ detects a 4-clique when the current observed edge completes a 4-clique with *two triangles* in $\mathcal{S}_\Delta$. TS4C$_1$ and TS4C$_2$ differ *only* in the function UPDATE4CLIQUES, the pseudocode for TS4C$_2$ is presented in Algorithm 2.

The probability of detecting a 4-clique using TS4C$_2$ computed by PROBCLIQUE is different from the correspondent probability computed in TS4C$_1$. For the details we refer the reader to Lemma B.2 in Appendix B of the extended version [21].

Lemma IV.4. *Let $\lambda \in \mathcal{C}_4^{(t)}$ and let p_λ denote the probability of λ being observed on the stream by* TS4C$_2$. *We have:*

$$p_\lambda \le \left(\frac{M_e}{t-1}\right)^4\left(\frac{M_\Delta}{\tau^{(t)}}\right)^2.$$

Applying this lemma to an analysis similar to the one used in the proof of Theorem IV.1 we prove that the estimations obtained using TS4C$_2$ are *unbiased*.

Theorem IV.5. *The estimator $\varkappa$ returned by* TS4C$_2$ *is* unbiased, *that is:* $\mathbb{E}\left[\varkappa^{(t)}\right] = |\mathcal{C}_k^{(t)}|$.

The analysis presents several complications due to the interplay of the probabilities of observing each of the two triangles that share an edge. For the details of these results we refer the reader to Appendix B of the extended version of this work [21]. Following the same criterion discussed in Section IV-C, we use $|\mathcal{S}_e| = 2M/3$ and $|\mathcal{S}_\Delta| = M/3$ as a general rule for assigning the available memory space between the two sample levels. Although the difference between TS4C$_1$ and TS4C$_2$ may appear of minor interest, our experimental analysis show that it can lead to significantly different performances depending on the properties of the graph $G^{(t)}$. Intuitively, TS4C$_2$ emphasizes the importance of the triangle sub-structures compared to TS4C$_1$, thus resulting in better performance when the input graph is very sparse with smaller frequencies of 3 and 4-cliques.

V. Comparison with single sample approach

To quantify the advantage of our TIEREDSAMPLING approach we construct and fully analyze algorithm FOUREST that uses a single sample strategy. We then compare the performance of FOUREST and TS4C$_1$ analytically, and through experiments on both synthetic and real-world data.

A. FOUREST

FOUREST is an extension of the reservoir sample triangle counting algorithm in [6], using one reservoir sample to store uniform random sample $\mathcal{S}$ of size M of the edges observed over the stream. The pseudocode for FOUREST is presented in Algorithm 3.

ALGORITHM 3 FOUREST

```
Input: Edge stream Σ, integer M ≥ 6
Output: Estimation of the number of 4-cliques ϰ
S_e ← ∅, t ← 0, ϰ ← 0
for each element (u, v) from Σ do
    t ← t + 1
    UPDATE4CLIQUES(u, v)
    if SAMPLEEDGE((u, v), t) then
        S ← S ∪ {(u, v)}

function UPDATE4CLIQUES(u, v)
    N^S_{u,v} ← N^S_u ∩ N^S_v
    for each element (x, w) from N^S_{u,v} × N^S_{u,v} do
        if (x, w) in S_e then
            if t ≤ M then
                p ← 1
            else
                p ← min{1, M(M−1)(M−2)(M−3)(M−4) / (t−1)(t−2)(t−3)(t−4)(t−5)}
            ϰ ← ϰ + p^{-1}
```

The proofs of the following results can be found in the extended version of this work [21].

Theorem V.1. *Let $\varkappa^{(t)}$ the estimated number of 4-cliques in $G^{(t)}$ computed by* FourEst *using memory of size M. $\varkappa^{(t)} = |\mathcal{C}_4^{(t)}|$ if $t \leq M$ and $\mathbb{E}\left[\varkappa^{(t)}\right] = |\mathcal{C}_4^{(t)}|$ if $t > M$.*

We now show an *upper bound* to the variance of the FourEst estimations for $t > M$ (for $t \leq M$ we have $\varkappa^{(t)} = |\mathcal{C}_4^{(t)}|$ and thus the variance of $\varkappa^{(t)}$ is zero).

Theorem V.2. *For any time $t > M$, we have*

$$Var\left[\varkappa^{(t)}\right] \leq |\mathcal{C}_4^{(t)}| \left(\left(\frac{t-1}{M}\right)^5 - 1\right) + a^{(t)}\left(\frac{t-1}{M} - 1\right) + b^{(t)}\left(\left(\frac{t-1}{M_e}\right)^3 - 1\right).$$

B. Variance comparison

Although the upper bounds obtained in Theorems IV.2 and V.2 cannot be compared directly, they still provide some useful insight on which algorithm may be performing better according to the properties of $G^{(t)}$.

Let us consider the first, dominant, terms of each of the variance bounds, that is $|\mathcal{C}_4^{(t)}\left(\left(\frac{5}{4}\frac{t}{M}\right)^4 \frac{5\tau^{(t)}}{M} - 1\right)|$ for TS4C$_1$ (assigning $M_e = 4M/5$ and $M_\Delta = M/5$), and $|\mathcal{C}_4^{(t)}|\left(\left(\frac{t-1}{M}\right)^5 - 1\right)$ for FourEst. While TS4C$_1$ exhibits a slightly higher constant multiplicative term a cost due to the splitting of the memory in the TieredSampling approach, the most relevant difference is however given by the term $\frac{\tau^{(t)}}{M}$ appearing in the bound for TS4C$_1$ compared with an additional $\frac{t-1}{M}$ appearing in the bound fro FourEst. Recall that $\tau^{(t)}$ denotes here the number of triangles *observed* by the algorithm up to time t. Due to the fact that the probability of observing a triangle decreases quadratically with respect to the size of the graph t, we expect that $\tau < t$ and, for sparser graphs for which 3 and 4-cliques are indeed "*rare patterns*, we actually expect $\tau^{(t)} << t$. Under these circumstances we would therefore expect $M/5\tau >> M/t$.

This is the critical condition for the success of the TieredSampling approach. If the sub-structure selected as a tool for counting the motif of interest is not "*rare enough*" then there is no benefit in devoting a certain amount of the memory budget to maintaining a sample of occurrences of the sub-structure. Such problem would for instance arise when using the TieredSampling approach for counting triangles using *wedges* (i.e., two-hop paths) as a sub-structure, as attempted in [14], as in most real-world graph the number of wedges is much greater of the number of edges themselves making them not suited to be used as a sub-structure.

m	FourEst	TS4C$_1$	Change TS4C$_1$	TS4C$_2$	Change TS4C$_2$
50	0.8775	0.5862	-33.19%	0.5222	-40.49%
100	0.3054	0.1641	-46.27%	0.1408	-53.90%
150	0.1521	0.0937	-38.34%	0.0917	-39.70%
200	0.0899	0.0599	-33.39%	0.0549	-38.95%
300	0.0486	0.0346	-28.80%	0.0417	-14.19%
400	0.0289	0.0249	-13.87%	0.0261	-9.32%
500	0.0221	0.0197	-11.28%	0.0239	8.08%
750	0.0134	0.0132	-1.57%	0.0181	34.90%
1000	0.0088	0.0099	13.14%	0.0146	66.36%

Table I
Comparison of MAPE of FourEst. TS4C$_1$ and TS4C$_2$ for Barábasi-Albert graphs.

C. Experimental evaluation over random graphs

In this section we compare the performances of our TieredSampling algorithms of TS4C$_1$ and TS4C$_2$ with the performance of the a single sample approach FourEst, on randomly generated graphs. In particular, we analyze random Barábasi-Albert graphs [23] with n nodes and mn total edges, as they exhibit the same *scale-free* property observed in many real-world graphs of interest such as social networks.

In our experiments, we set $n = 20000$ and we consider various values for m from 50 to 2000 in order to compare the performances of the two approaches as the number of edges (and thus triangles) increases. The algorithms use a memory space whose size corresponds to 5% of the number of edges in the graph nm. In columns 2,3 and 5 of Table I we present the average of the MAPEs of the ten runs for FourEst, TS4C$_1$ and TS4C$_2$. In column 4 (resp., 6) of Table I we report the percent reduction/increase in terms of the average MAPE obtained by TS4C$_1$ (resp., TS4C$_2$) with respect to FourEst.

Both TieredSampling algorithms consistently outperform FourEst for values of m up to 400, that is for fairly sparse graphs for which we expect 3 and 4 cliques to be rare patterns. The advantage over FourEst is particularly strong for values of m up to 200 with reductions up to 30%. For denser graphs, i.e. $m \geq 750$, FourEst outperforms *both* TieredSampling algorithms. This in consistent with the intuition discussed in Section V-B, as for for denser graphs triangles are not "*rare enough*" to be worth saving in the tiered sample over edges. Note however that in these cases the quality of *all* the estimators is very high (i.e., MAPE$\leq$ 1%).

We can also observe that TS4C$_2$ outperforms TS4C$_1$ for $m \leq 200$. Vice versa, TS4C$_1$ outperforms TS4C$_2$ (and FourEst) for $300 \leq m \leq 750$. Since, as discussed in Section IV-E, TS4C$_2$ gives "*more importance*" to triangles, it works particularly well when the graph is very sparse, and triangles are particularly rare. As the graph grows denser (and the number of triangles increases), TS4C$_1$ performs better until, for highly dense graphs FourEst produces the best estimates.

VI. Adaptive Tiered Sampling Algorithm

An appropriate partition of the available memory between the layers used in the TieredSampling approach is crucial for the success of the algorithm: while assigning more memory to the triangle sample allows maintain more sub-patterns, removing too much space from the edge sample reduces the probability of observing new triangles. While in Section IV-D we provide a general rule to decide how to split the memory budget for TS4C_1 and TS4C_2, such partition may not always lead to the best possible results. For instance, if the graphs being observed is particularly sparse, assigning a large portion of the memory to the triangles would result in a considerable waste of memory space due to the low probability of observing triangles. Further, as discussed in Section V, depending on the properties of $G^{(t)}$ a single edge approach could perform better than the Tiered-Sampling algorithms. As in the graph streaming setting these properties are generally not known a priori, nor stable through the graph evolution, a fixed memory allocation policy appears not to be the ideal solution. In this section we present ATS4C, an *adaptive* variation of our TS4C_2 algorithm, which *dynamically* analyzes the properties of $G^{(t)}$ through time and consequently decides how to allocate the available memory.

Algorithm description: ATS4C has two main "*execution regimens*": **(R1)** for which it behaves exactly as FourEst, and **(R2)** for which it behaves similarly to TS4C_2. **(R1)** is the initial regimen for ATS4C. Recall that, from Lemma II.1 (resp., Lemma IV.4) the probability of a 4-clique being observed by FourEst (resp., TS4C_2)is upper bounded by $p_s = (\min\{1, M/t\})^5$ (resp., $p_\alpha = (\min\{1, \alpha M/t\})^4(\min\{1, (1-\alpha)M/\tau^{(t)}\})^2$, where $0 < \alpha < 1$). Every M time steps the algorithm evaluates, based on the number of triangles observed so far, whether to switch to **(R2)**. ATS4C $\alpha^* = \text{argmax}_{\alpha \in [2/3,1)} p_\alpha$. If $p_s > p_{\alpha^*}$, ATS4C remains in **(R1)**. Vice versa ATS4C transitions to **(R2)**: the triangle reservoir $\mathcal{S}_\Delta$ is assigned $(1-\alpha^*)M$ memory space, and it is filled with the triangles composed by the edges *currently* in the edge reservoir, using the reservoir sampling scheme. Finally the edge reservoir $\mathcal{S}_e$ is constructed by selecting $\alpha^* M$ of the edges in the current sample uniformly at random, thus ensuring that $\mathcal{S}_e$ is an *uniform sample*. Once ATS4C switches to **(R2)** it never goes back to **(R1)**. During (R2), as long as $|\mathcal{S}_\Delta| < M/3$, every M time steps ATS4C evaluates whether it is opportune to assign more memory to $\mathcal{S}_\Delta$. Let $t = iM$, rather than just using the information of the number of triangles seen so far $\tau^{(iM)}$, ATS4C computes a "*prediction*" of the total number of triangles seen until $(i+1)M$ assuming that the number of triangles seen during the next M steps will equals the number of triangles seen during the last M steps, that is $\tau^{((i+\tilde{1})M)} = 2\tau^{(iM)} - \tau^{((i-)M)}$. ATS4C then computes $\alpha^* = \text{argmax}_{a \in [2/3,1)}(\min\{1, \alpha M/(i+1)M)\})^4(\min\{1, (1-\alpha)M/\tau^{((i+1)M)}\})^2$. Let α denote the split being used by ATS4C at $t = iM$, if $\alpha^* > \alpha$ the algorithm continues its execution with no further operations (ATS4C never reduces the memory space assigned to $\mathcal{S}_\Delta$). Vice versa, if $\alpha^* < \alpha$, ATS4C removes $(\alpha - \alpha^*)M$ edges from $\mathcal{S}_e$ selected uniformly at random, and the freed space is assigned to $\mathcal{S}_\Delta$. Let us denote this space as $\mathcal{S}'_\Delta$. As ATS4C progresses and observes new triangles it fills $\mathcal{S}'_\Delta$ using the reservoir sampling scheme. $\mathcal{S}_\Delta$ and $\mathcal{S}'_\Delta$ are then merged at the first time step for which the probability p_Δ of a triangle seen before the creation of $\mathcal{S}'_\Delta$ being in $\mathcal{S}_\Delta$ becomes lower then the probability $p_{\Delta'}$ of a triangle observed after the creation of $\mathcal{S}'_\Delta$ being in it. The merged triangle sample contains all the triangles in $\mathcal{S}'_\Delta$, while the triangles in $\mathcal{S}_\Delta$ are moved to it with probability $p_{\Delta'}/p_\Delta$. This ensures that after the merge all the triangles seen on the stream are kept in the triangle reservoir with probability $p_{\Delta'}$. After the merge, ATS4C operates the samples as described in TS4C_2. Finally, ATS4C increases the memory space for $\mathcal{S}_\Delta$ only if all the currently assigned space is used.

The analysis of ATS4C is much more complicated than the one for our previous algorithms as it requires to keep track of the probabilities according to which the observed triangles appear in $\mathcal{S}_\Delta$. We claim however (without explicit proof) that the estimation provided by ATS4C is unbiased. Via the experimental analysis in Section VII-B, we show how ATS4C succeeds in merging the advantages of the single sample approach and of an adaptive splitting of the in tiers.

VII. Experimental Evaluation

In this section, we evaluate through extensive experiments the performance of our proposed TieredSampling method when applied for counting 4 and 5-cliques in large graphs observed as streams. We use several real-world graphs with size ranging form 10^6 to 10^8 edges (see Table II for a complete list). All graphs are treated as undirected. The edges are observed on the stream according to the values of the associated timestamps if available, or in random order otherwise. In order to evaluate the accuracy of our algorithms, we compute the "*ground truth*" exact number of 4-cliques (resp., 5-cliques) for each time step using an exact algorithm which maintains the entire $G^{(t)}$ in memory. Our algorithms are implemented in Python. The experiments were run on the Brown University CS department cluster[1], where each run employed a single core and used at most 4GB of RAM. The section is organized as follows: we first evaluate the performance of TS4C_1 and TS4C_2

[1]https://cs.brown.edu/about/system/services/hpc/grid/

Graph	Nodes	Edges	Source
DBLP	986,324	3,353,618	[24]
Patent (Cit)	2,745,762	13,965,132	[6]
LastFM	681,387	30,311,117	[6]
Live Journal	5,363,186	49,514,271	[24]
Hollywood	1,917,070	114,281,101	[24]
Orkut	3,072,441	117,185,083	[25]
Twitter	41,652,230	$1.20 \cdot 10^9$	[6, 24]

Table II
GRAPHS USED IN THE EXPERIMENTS

when run on several massive graphs and we compare them with the estimations obtained using FOUREST. We then present practical examples that motivate the necessity for the adaptive version of our TIEREDSAMPLING approach, and we show how our ATS4C manages to capture the best of the single and multi-level approach. Finally, we show how the TIEREDSAMPLING approach can be generalized in order to count structures other than 4-cliques.

A. Counting 4-Cliques

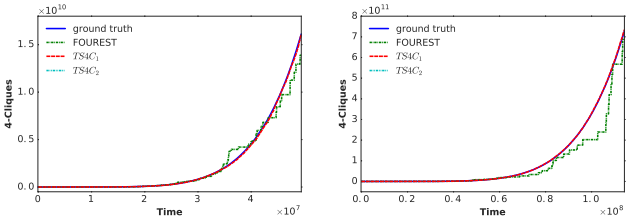
We estimate the global number of 4-cliques on insertion-only streams, starting as empty graphs and for which an edge is added at each time step, using algorithms FOUREST, TS4C$_1$ and TS4C$_2$. As discussed in Section IV-C (resp., Section IV-E),in TS4C$_1$ (resp., TS4C$_2$) we split the total available memory space M as $|\mathcal{S}_e| = 4M/5$ and $|\mathcal{S}_\Delta| = M/5$ (resp., $|\mathcal{S}_e| = 2M/3$ and $|\mathcal{S}_\Delta| = M/3$).

The experimental results show that these fixed splittings perform well for most cases. We then experiment with an adaptive splitting mechanism that handles the remaining cases. In Figure 2 we present the estimation obtained by averaging 10 runs of respectively TS4C$_1$, TS4C$_2$ and FOUREST using total memory space $M = 5 \times 10^5$ for the LiveJournal and Hollywood graphs (i.e., respectively using less than 1% and 0.5% of the graph size). While the average of the runs for TS4C$_1$ and TS4C$_2$ are almost *indistinguishable* from the ground truth, that is clearly not the case for FOUREST for which the quality of the estimator considerably worsens as the graph size increases.

In Table III, we report the average MAPE performance over 10 runs for TS4C$_1$, TS4C$_2$ and FOUREST for several graphs of different size. Depending on the size of the input graph we assign different total memory space (as reported in Table III), which in most cases amounts to at most 3% ($\sim$ 8%M for Patent(Cit)) of the input graph size. Except for LastFM, our TIEREDSAMPLING algorithms clearly outperform FOUREST, with the average MAPE reduced by up to 30%. Both TS4C$_1$ and TS4C$_2$ return very accurate estimations of the number of 4-cliques on the majority of these graphs, with average MAPE lower than 10% (16% for LastFM and Orkut). LastFM is the only graph for which FOUREST (considerably) outperforms the TIEREDSAMPLING algorithms. This is due to the high density of the graph $|E|/|V| > 500$ and to the fact that in this case triangles are not a rare enough sub-structure to justify the choice of maintaining them over simple edges.

Dataset	M	FOUREST	TS4C$_1$	TS4C$_2$
Patent (Cit)	10^6	0.0963	0.0921	0.0474
LastFM	10^6	0.0118	0.1258	0.0777
LiveJournal	10^6	0.1521	0.0543	0.0560
Hollywood	$2 \cdot 10^6$	0.0355	0.0207	0.0194
Orkut	$2 \cdot 10^6$	0.4674	0.1590	0.1417
Twitter[2]	$5 \cdot 10^6$	0.2503	0.0749	0.0742

Table III
AVERAGE MAPE OF VARIOUS APPROACHES FOR ALL GRAPHS.



(a) Live Journal (b) Hollywood

Figure 2. Comparison of $|\mathcal{C}_4^{(t)}|$ estimates obtained using TS4C$_1$, TS4C$_2$ and FOUREST with $M = 5 \times 10^5$,

We analyze the variance reduction achieved using our TIEREDSAMPLING algorithms by comparing the empirical variance observed over forty runs on the Hollywood graph using $M = 5 \times 10^5$. The results are reported in Figure 3. While for both TS4C$_1$ and TS4C$_2$ the minimum and maximum estimators are close to the ground truth throughout the evolution of the graph, TS4C$_1$ estimators exhibit very high variance especially towards the end of the stream.

Our experiments not only verify that TS4C$_1$ and TS4C$_2$ allow to obtain good quality estimations which are in most cases superior to the ones achievable using a single sample strategy, but also validate the general intuition underlying the TIEREDSAMPLING approach.

Both TIEREDSAMPLING algorithms are extremely scalable, showing average update times in the order of hundreds microseconds for all graphs.

B. Adaptive Tiered Sampling

In Section VII-A, we showed that TS4C$_2$, allows to obtain high quality estimations for the number of 4-cliques outperforming in most cases both TS4C$_1$ and FOUREST. These results were obtained splitting the available memory such that $|\mathcal{S}_e| = 2M/3$ and $|\mathcal{S}_\Delta| = M/3$. As discussed in Section VI, while this is an useful general rule, depending on the properties of the graph different splitting rules may yield better results. We verify this fact by evaluating the performance of TS4C$_2$

[2]Ground truth computed for first $3 \cdot 10^8$ edges on the stream.

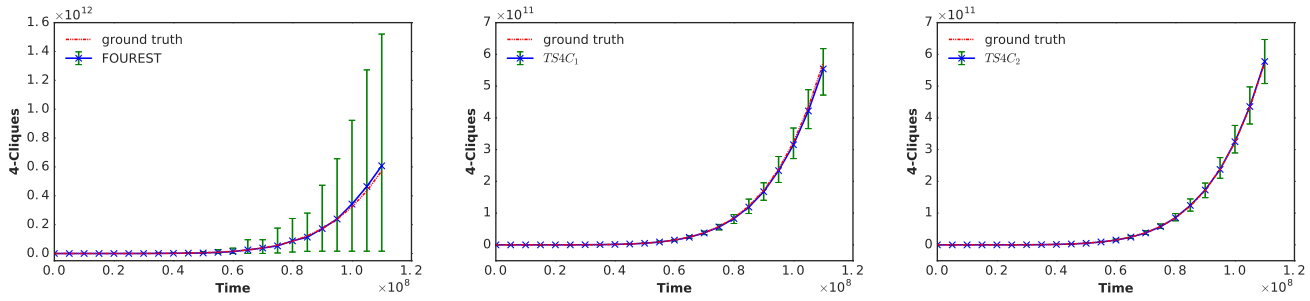


Figure 3. Variance of FOUREST, TS4C$_1$ and TS4C$_2$ for Hollywood graph with $M = 5 \times 10^5$.

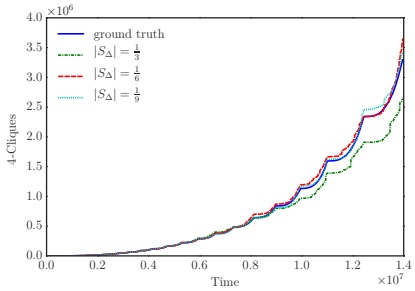
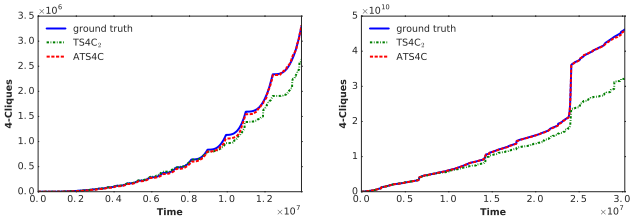


Figure 4. $|\mathcal{C}_4^{(t)}|$ estimations for Patent (Cit) using TS4C$_2$ with $M = 5 \times 10^5$ and different memory space assignments among tiers.



(a) Patent(Cit) (b) LastFM

Figure 5. Comparison of $|\mathcal{C}_4^{(t)}|$ estimates obtained using ATS4C, TS4C$_2$ and FOUREST with $M = 5 \times 10^5$.

when run on the Patent(Cit) graph using different assignments of the total space $M = 5 \times 10^5$ to the two levels. The results in Figure 4 show that decreasing the space assigned to the triangle sample from $M/3$ to $M/9$ allows to achieve estimations which are closer to the ground truth leading to a 31% reduction in the average MAPE. Due to the sparsity of the Patent(Cit) graph, TS4C$_2$ observes a very small number of triangles for large part of the stream. Assigning a large fraction of the memory space to $\mathcal{S}_\Delta$ is thus inefficient as the probability of observing new triangles is reduced, and the space assigned to $\mathcal{S}_\Delta$ is not fully used.

To overcome such difficulties, in Section VI we introduced ATS4C, an *adaptive* version of TS4C$_2$, which allows to dynamically adjust the use of the available memory space based on the properties of the graph being observed. We experimentally evaluate the performance of ATS4C over 10 runs on the Patent(Cit) and the LastFM graphs and we compare it with TS4C$_2$ and FOUREST using $M = 5 \times 10^5$.

As shown in Figure 5, for both graphs, ATS4C produces estimates that are nearly indistinguishable from the ground truth. ATS4C clearly outperforms TS4C$_2$ (with $|\mathcal{S}_\Delta| = \frac{M}{3}$) on Patent(Cit) where triangles are sparse motifs achieving an $\sim$ 85% reduction in the average MAPE compared to TS4C$_2$. ATS4C returns high quality estimations even for the LastFM graph, for which the single level approach FOUREST outperforms the TIEREDSAMPLING algorithms.

C. Counting 5-Cliques

To demonstrate the generality of our TIEREDSAMPLING approach we present TS5C, a one pass counting algorithms for 5-clique in a stream. The algorithm maintains two reservoir samples, one for edges and one for 4-cliques. When the current observed edge completes a 4-clique with edges in the edge sample the algorithm attempts to insert it to the 4-cliques reservoir sample. A 5-clique is counted when the current observed edge completes a 5-clique using one 4-clique in the reservoir sample and 3 edges in the edge sample.

Our experiments compare the performance of TS5C to that of a standard one tier edge reservoir sample algorithm FIVEEST, similar to FOUREST. We evaluate the average performance over 10 runs of the two algorithms on the DBLP graph using $M = 3 \times 10^5$. The results are presented in Figure 6. TS5C clearly outperforms FIVEEST in obtaining much better estimations of the ground truth value $|\mathcal{C}_5^{(t)}|$ achieving an $\sim$ 56% reduction in the average MAPE.

VIII. CONCLUSIONS

We developed TIEREDSAMPLING, a novel technique for approximate counting sparse motifs in massive graphs whose edges are observed in one pass stream. We studied application of this technique for the problems of counting 4 and 5-cliques. We present both analytical proofs and experimental results, demonstrating the

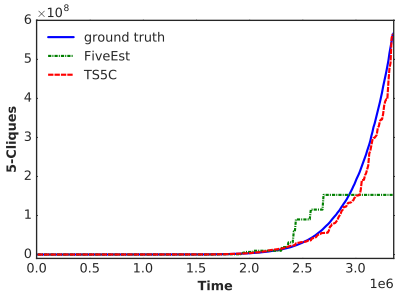


Figure 6. Comparison of $|\mathcal{C}_5^{(t)}|$ estimates for the DBLP graph obtained using TS5C and FiveEst with $M = 3 \times 10^5$.

advantage of our method in counting sparse motifs compared to the standard methods of using just edge reservoir sample. With the growing interest in discovering and analyzing large motifs in massive scale graphs in social networks, genomics, and neuroscience, we expect to see further applications of our technique.

References

[1] J. W. Berry, B. Hendrickson, R. A. LaViolette, and C. A. Phillips, "Tolerating the community detection resolution limit with edge weighting," *Physical Review E*, vol. 83, no. 5, p. 056119, 2011.

[2] J.-P. Eckmann and E. Moses, "Curvature of co-links uncovers hidden thematic layers in the World Wide Web," *PNAS*, vol. 99, no. 9, pp. 5825–5829, 2002.

[3] L. Becchetti, P. Boldi, C. Castillo, and A. Gionis, "Efficient algorithms for large-scale local triangle counting," *ACM TKDD*, pp. 13:1–13:28, 2010.

[4] Y. Lim and U. Kang, "Mascot: Memory-efficient and accurate sampling for counting local triangles in graph streams," in *KDD'15*. ACM, 2015, pp. 685–694.

[5] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[6] L. De Stefani, A. Epasto, M. Riondato, and E. Upfal, "Trièst: Counting local and global triangles in fully dynamic streams with fixed memory size," *ACM TKDD'17*, vol. 11, no. 4, p. 43, 2017.

[7] A. Pavan, K. Tangwongsan, S. Tirthapura, and K.-L. Wu, "Counting and sampling triangles from a graph stream," *VLDB'13*, pp. 1870–1881, 2013.

[8] K. Kutzkov and R. Pagh, "Triangle counting in dynamic graph streams," in *SWAT '14*. Springer, 2014, pp. 306–318.

[9] J. S. Vitter, "Random sampling with a reservoir," *TOMS*, vol. 11, no. 1, pp. 37–57, 1985.

[10] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *IJF*, vol. 22, no. 4, pp. 679–688, 2006.

[11] R. Pagh and C. E. Tsourakakis, "Colorful triangle counting and a mapreduce implementation," *Inf. Process. Lett.*, pp. 277–281, 2012.

[12] H.-M. Park and C.-W. Chung, "An efficient mapreduce algorithm for counting triangles in a very large graph," in *CIKM '13*, 2013, pp. 539–548.

[13] N. K. Ahmed, N. Duffield, J. Neville, and R. Kompella, "Graph sample and hold: A framework for big-graph analytics," in *KDD '14*, 2014, pp. 1446–1455.

[14] M. Jha, C. Seshadhri, and A. Pinar, "A space-efficient streaming algorithm for estimating transitivity and triangle counts using the birthday paradox," *ACM TKDD*, vol. 9, no. 3, pp. 15:1–15:21, Feb. 2015.

[15] M. Rahman, M. A. Bhuiyan, and M. Al Hasan, "Graft: An efficient graphlet counting method for large graph analysis," *IEEE TKDE*, vol. 26, no. 10, pp. 2466–2478, 2014.

[16] N. K. Ahmed, J. Neville, R. A. Rossi, and N. Duffield, "Efficient graphlet counting for large networks," in *ICDM'15*. IEEE, 2015, pp. 1–10.

[17] S. Jain and C. Seshadhri, "A fast and provable method for estimating clique counts using turán's theorem," in *WWW '17*, 2017, pp. 441–449.

[18] I. Finocchi, M. Finocchi, and E. G. Fusco, "Clique counting in mapreduce: Algorithms and experiments," *JEA*, vol. 20, pp. 1.7:1–1.7:20, Oct. 2015.

[19] I. Bordino, D. Donato, A. Gionis, and S. Leonardi, "Mining large networks with subgraph counting," in *ICDM'08*. IEEE, 2008, pp. 737–742.

[20] M. Jha, C. Seshadri, and A. Pinar, "Path sampling: A fast and provable method for estimating 4-vertex subgraph counts," in *WWW '15*, 2015, pp. 495–505.

[21] L. De Stefani, E. Terolli, and E. Upfal, "Tiered sampling: An efficient method for approximate counting sparse motifs in massive graph streams - extended materials," 2017. [Online]. Available: http://bigdata.cs.brown.edu/tiered_sampling.pdf

[22] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge university press, 2017.

[23] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of modern physics*, vol. 74, no. 1, p. 47, 2002.

[24] P. Boldi, M. Rosa, M. Santini, and S. Vigna, "Layered label propagation: A multiresolution coordinate-free ordering for compressing social networks," in *WWW'11*. ACM, 2011.

[25] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *ACM SIGCOMM'07*, 2007, pp. 29–42.

Appendix A.
Additional theoretical results for TS4C$_1$

Before presenting the proof for the main analytical results for TS4C$_1$ discussed in Section IV, we introduce some technical lemmas.

Lemma A.1. *Let $\lambda \in \mathcal{C}_4^{(t)}$ with $\lambda = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ using Figure 1 as reference. Assume further, without loss of generality, that the edge e_i is observed at t_i (not necessarily consecutively) and that $t_6 > \max\{t_i, 1 \leq i \leq 5\}$. λ can be observed by* TS4C$_1$ *at time t_6 either as a combination of triangle $T_1 = \{e_1, e_2, ore_4\}$ and edges $e_3 = (v, w)$ and $e_5 = (v, z)$, or as a combination of triangle $T_2 = \{e_1, e_3, e_5\}$ and edges $e_2 = (u, w)$ and $e_4 = (u, z)$.*

Proof: As presented in Algorithm 1, TS4C$_1$ can detect λ only when its last edge is observed on the stream (hence, t_6). When e_6 is observed, the algorithm first evaluates wether there is any triangle in $\mathcal{S}_\Delta^{(t_6)}$ that shares one of the two endpoint u or v form e_6. Since a this step (i.e., the execution of fucntion UPDATE4CLIQUES) the triangle sample is jet to be updated based on the observation of e_6, the only triangle sub-structures of λ which may have been observed on the stream, and thus included in $\mathcal{S}_\Delta^{(t_6)}$ are $T_1 = \{e_1, e_2, e_4\}$ and $T_2 = \{e_1, e_3, e_5\}$. If any of these is indeed in $\mathcal{S}_\Delta^{(t_6)}$, TS4C$_1$ proceeds to check wether the remaining two edges required to complete λ (resp., e_3, e_5 for T_1, or e_2, e_4 for T_2) are in $\mathcal{S}_e$. λ is thus observed either once or twice depending on which just one or both of these conditions are verified. ∎

Lemma A.2. *Let $\lambda \in \mathcal{C}_4^{(t)}$ with $\lambda = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ using Figure 1 as reference. Assume further, without loss of generality, that the edge e_i is observed at t_i (not necessarily consecutively) and that $t_6 > \max\{t_i, 1 \leq i \leq 5\}$. Let $t_{1,2,4} = \max t_1, t_2, t_4, M_e + 1$. The probability p_λ of λ being observed on the stream by* TS4C$_1$ *using the triangle $T_1 = \{e_1, e_2, e_4\}$ and the edges e_3, e_5, is computed by* PROBCLIQUE *as:*

$$p_\lambda = \frac{M_e}{t_{1,2,4}^M - 1} \frac{M_e - 1}{t_{1,2,4}^M - 2} \min\{1, \frac{M_\Delta}{\tau^{(t_6)}}\} p'$$

where if $t_6 \leq M_e$

$$p' = 1,$$

if $\min\{t_3, t_5\} > t_{1,2,4}$

$$p' = \frac{M_e}{t_6 - 1} \frac{M_e - 1}{t_6 - 2},$$

if $\max\{t_3, t_5\} > t_{1,2,4} > \min\{t_3, t_5\}$

$$p' = \frac{M_e - 1}{t_6 - 2} \frac{M_e - 2}{t_{1,2,4} - 3} \frac{t_{1,2,4} - 1}{t_6 - 1}$$

otherwise

$$p' = \frac{M_e - 2}{t_{1,2,4} - 3} \frac{M_e - 3}{t_{1,2,4} - 4} \frac{t_{1,2,4} - 1}{t_6 - 1} \frac{t_{1,2,4} - 2}{t_6 - 2}.$$

Proof: Let us define the event $E_\lambda = \lambda$ *is observed on the stream by* TS4C$_1$ *using triangle $T_1 = \{e_1, e_2, e_4\}$ and edges e_3, e_5.* Further let $E_{T_1} = T_1 \in \mathcal{S}_\Delta^{(t_6)}$, and $E_{3,5} = \{e_3, e_5\} \subseteq \mathcal{S}_e^{(t_6)}$. Given the definition of TS4C$_1$ we have:

$$E_\lambda = E_{T_1} \wedge E_{3,5},$$

and hence:

$$p_\lambda = \Pr(E_\lambda) = \Pr(E_{T_1} \wedge E_{3,5}) = \Pr(E_{3,5} | E_{T_1}) \Pr(E_{T_1}).$$

In oder to study $\Pr(E_{T_1})$ we shall introduce event $E_{S(T_1)}$ ="*triangle T_1 is observed on the stream by* TS4C$_1$". From the definition of TS4C$_1$ we know that T_1 is observed on the stream iff when the last edge of T_1 is observed on the stream at $\max\{t_1, t_2, t_4\}$ the remaining to edges are in the edge sample. Applying Bayes's rule of total probability we have:

$$\Pr(E_{T_1}) = \Pr\left(E_{T_1} | E_{S(T_1)}\right) \Pr\left(E_{S(T_1)}\right).$$

and thus:

$$p_\lambda = \Pr(E_{3,5} | E_{T_1}) \Pr\left(E_{T_1} | E_{S(T_1)}\right) \Pr\left(E_{S(T_1)}\right). \quad (1)$$

Let $t_{1,2,4} = \max\{t_1, t_2, t_4, M+1\}$, if order for T_1 to be observed by TS4C$_1$ it is required that when the last edge of T_1 is obseved on the stream at $t_{1,2,4}$ its two remaining edges are kept in $\mathcal{S}_e$. From Lemma II.1 we have:

$$\Pr\left(E_{S(T_1)}\right) = \frac{M_e}{t_{1,2,4}-1}\frac{M_e-1}{t_{1,2,4}-2}, \tag{2}$$

and:

$$\Pr\left(E_{T_1}|E_{S(T_1)}\right) = \frac{M_e}{\tau^{t_6}}. \tag{3}$$

Let us now consider $\Pr\left(E_{3,5}|E_{T_1}\right)$. While the content of $\mathcal{S}_\Delta$ itself does not influence the content of $\mathcal{S}_e$, the fact that T_1 is maintained in $\mathcal{S}\Delta^{(t_6)}$ implies that is has been observed on the stream at a previous time and hence, that two of its edges have been maintained in $\mathcal{S}_e$ *at least* until the last of its edges has been observed on the stream. We thus have $\Pr\left(E_{3,5}|E_{T_1}\right) = \Pr\left(E_{3,5}|E_{S(T_1)}\right)$. In order to study $p' = \Pr\left(E_{3,5}|E_{S(T_1)}\right)$ it is necessary to distinguish the possible (5!) different arrival orders for edges e_1, e_2, e_3, e_4 and e_5. An efficient analysis we however reduce the number of cases to be considered to just four:

- $t_6 \leq M + e$: in this case *all* the edges observed on the stream up until t_6 are deterministically inserted in $\mathcal{S}_e$ and thus $p' = 1$.
- $\min\{t_3, t_5\} > t_{1,2,4}$: in this cases both edges e_3 and e_5 are observed after *all* the edges composing T_1 have already been observed on the stream. As for any $t > t_{1,2,4}$ the event $E_{S(T_1)}$ does not imply that any of the edges of T_1 is *still* in $\mathcal{S}_e$ we have:
$$p' = \Pr\left(E_{3,5}|E_{T_1}\right) = \Pr\left(\{e_3, e_5\} \subseteq \mathcal{S}_e^{(t_6)}\right),$$
and thus, from Lemma II.1:
$$p' = \frac{M_e}{t_6-1}\frac{M_e-1}{t_6-2}.$$
- $\max\{t_3, t_5\} > t_{1,2,4} > \min\{t_3, t_5\}$: in this case only one of the edges e_3, e_5 is observed after all the edges in T_1 are observed. We need to therefore take into consideration that the two edges of T_1 are kept in S_e until $t_{1,2,4}$. Let $e_{3,5}^M$ (resp., $e_{3,5}^m$) denote the last (resp., first) edge observed on the stream between e_3 and e_5.
$$\begin{aligned} p' &= \Pr\left(e_{3,5}^M \in \mathcal{S}_e^{(t_6)} | e_{3,5}^m \in \mathcal{S}_e^{(t_6)}\right) \Pr\left(e_{3,5}^m \in \mathcal{S}_e^{(t_6)}\right), \\ &= \frac{M_e-1}{t_6-2}\Pr\left(e_{3,5}^m \in \mathcal{S}_e^{(t_6)} | e_{3,5}^m \in \mathcal{S}_e^{(t_{1,2,4})}\right) \Pr\left(e_{3,5}^m \in \mathcal{S}_e^{(t_{1,2,4})}\right), \\ &= \frac{M_e-1}{t_6-2}\frac{t_{1,2,4}-1}{t_6-1}\frac{M_e-2}{t_{1,2,4}-3}. \end{aligned}$$
- $t_{1,2,4} > \max\{t_2, t_4\}$: in all the remaining cases both e_3 and e_5 are observed before the last edge of T_1 has been observed. Hence:
$$\begin{aligned} p' &= \Pr\left(\{e_3, e_5\} \subseteq \mathcal{S}_e^{(t_6)} | \{e_3, e_5\} \subseteq \mathcal{S}_e^{(t_{1,2,4})}\right) \Pr\left(\{e_3, e_5\} \subseteq \mathcal{S}_e^{(t_{1,2,4})}\right), \\ &= \frac{M_e-2}{t_{1,2,4}-3}\frac{M_e-3}{t_{1,2,4}-4}\frac{t_{1,2,4}-1}{t_6-1}\frac{t_{1,2,4}-2}{t_6-2}. \end{aligned}$$

The lemma follows combining the result for the values of p' with (2) and (3) in (1). ∎

Using this result we can proceed to the proof of the unbiasedness of the estimator returned by TS4C$_1$.

Proof of Theorem IV.1: From Lemma A.1 we have that TS4C$_1$ can detect any 4-clique $\lambda \in \mathcal{C}_4^{(t)}$ in exactly two ways: either using triangle T_1 and edges e_3, e_5, or by using triangle T_2 and edges e_2, e_4 (use Figure 1 as a reference).

For each $\lambda \in \mathcal{C}_4^{(t)}$ let us consider the random variable δ_{λ_1} (resp. δ_{λ_2}) which takes value $p_{\lambda_1}^{-1}/2$ (resp., $p_{\lambda_2}^{-1}/2$) if the 4-clique λ is observed by TS4C$_1$ using triangle T_1 (resp., T_2) or zero otherwise. Let p_{λ_1} (resp., p_{λ_2}) denote the probability of such event: we then have $\mathrm{E}\left[\delta_{\lambda_1}\right] = \mathrm{E}\left[\delta_{\lambda_2}\right] = 1/2$.

From Lemma A.2 we the estimator $\varkappa^{(t)}$ computed using TS4C$_1$ has can be expressed as $\varkappa^{(t)} = \sum_{\lambda \in \mathcal{C}_4^{(t)}} (\delta_{\lambda_1} + \delta_{\lambda_2})$. From the previous discussion and by applying linearity of expectation we thus have:

$$\mathrm{E}\left[\varkappa^{(t)}\right] = \mathrm{E}\left[\sum_{\lambda \in \mathcal{C}_4^{(t)}} (\delta_{\lambda_1} + \delta_{\lambda_2})\right] = \sum_{\lambda \in \mathcal{C}_4^{(t)}} \left(\mathrm{E}\left[\delta_{\lambda_1}\right] + \mathrm{E}\left[\delta_{\lambda_2}\right]\right) = \sum_{\lambda \in \mathcal{C}_4^{(t)}} 1 = |\mathcal{C}_4^{(t)}|.$$

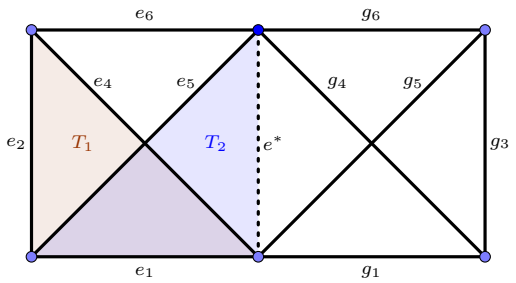


Figure 7. Cliques sharing one edge.

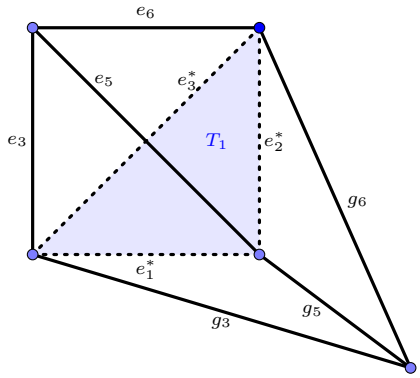


Figure 8. Cliques sharing three edges.

Finally, let t^* denote the first step at for which the number of triangles seen exceeds M_Δ. For $t \leq \min\{M_e, t^*\}$, the entire graph $G^{(t)}$ is maintained in $\mathcal{S}_e$ and all the triangles in $G^{(t)}$ are stored in $\mathcal{S}_\Delta$. Hence all the cliques in $G^{(t)}$ are deterministically observed by TS4C$_1$ *in both ways* and we therefore have $\varkappa^{(t)} = |\mathcal{C}_4^{(t)}|$. ■

Lemma A.3. *Any pair (λ, γ) of distinct 4-cliques in $G^{(t)}$ can share either one, three or no edges. If λ and γ share three edges, those three edges compose a triangle.*

Proof: Suppose that λ and γ share exactly two distinct edges. This implies that they share at least three distinct nodes, and thus must share the three edges connecting each pair out of said three nodes. This constitutes a contradiction. Suppose instead that λ and γ share four or five edges while being distinct. This implies that they must share four vertices, hence they cannot be distinct cliques. This leads to a contradiction. ■

We now proceed to the proof of the bound on the variance of TS4C$_1$ in Theorem IV.2. We remark that the bound presented here is loose as it sacrifices its precision for presentation purposes. A stronger bound would require a case by case analysis of all the 12! possible relative ordering according to which the edges of any pair of 4-cliques in $G^{(t)}$ are observed. Our proof technique simplifies the analysis by grouping these cases into macro-cases with a resulting loss in tightness.

Proof of Theorem IV.2: Assume $|\mathcal{C}_4^{(t)}| > 0$, otherwise TS4C$_1$ estimation is deterministically correct and has variance 0 and the thesis holds. For each $\lambda \in \mathcal{C}_4^{(t)}$ let $\lambda = \{e_1, e_2, e_3, e_4, e_5, e_6\}$, withouth loss of generality let us assume the edges are disposed as in Figure 1. Assume further, without loss of generality, that the edge e_i is observed at t_i (not necessarily consecutively) and that $t_6 > \max\{t_i, 1 \leq i \leq 5\}$. Let $t_{1,2,4} = \max\{t_1, t_2, t_4, M_e + 1\}$. Let us consider the random variable δ_{λ_1} (resp. δ_{λ_2}) which takes value $p_{\lambda_1}^{-1}/2$ (resp., $p_{\lambda_2}^{-1}/2$) if the 4-clique λ is observed by TS4C$_1$ using triangle $T_1 = \{e_1, e_2, e_4\}$ (resp., T_2) and edges e_3, e_5 (resp., e_2, e_4) or zero otherwise. Let p_{λ_1} (resp., p_{λ_2}) denote the probability of such event.

Since, from Lemma A.2 we know:

$$\mathrm{Var}\left[\delta_{\lambda_1}\right] = \frac{p_{\lambda_1}^{-1}}{4} - \frac{1}{4} \leq \frac{1}{4}\left(\frac{\tau^{(t)}}{M_\Delta}\prod_{i=0}^{3}\frac{t-1-i}{M_e-i} - 1\right),$$
$$\mathrm{Var}\left[\delta_{\lambda_2}\right] = \frac{{p_{\lambda_2}}^{-1}}{4} - \frac{1}{4} \leq \frac{1}{4}\left(\frac{\tau^{(t)}}{M_\Delta}\prod_{i=0}^{3}\frac{t-1-i}{M_e-i} - 1\right).$$

we have:

$$\begin{aligned}
\mathrm{Var}\left[\varkappa^{(t)}\right] &= \mathrm{Var}\left[\sum_{\lambda\in\mathcal{C}_4^{(t)}}\delta_{\lambda_1}+\delta_{\lambda_2}\right] = \sum_{\lambda\in\Delta^{(t)}}\sum_{\gamma\in\Delta^{(t)}}\sum_{i\in\{1,2\}}\sum_{j\in\{1,2\}}\mathrm{Cov}\left[\delta_{\lambda_i},\delta_{\gamma_j}\right] \\
&= \sum_{\lambda\in\Delta^{(t)}}\left(\mathrm{Var}\left[\delta_{\lambda_1}\right]+\mathrm{Var}\left[\delta_{\lambda_2}\right]\right) + \sum_{\lambda\in\Delta^{(t)}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\lambda_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\lambda_1}\right]\right) \\
&+ \sum_{\substack{\lambda,\gamma\in\Delta^{(t)}\\ \lambda\neq\gamma}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_1}\right]+\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_1}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_2}\right]\right) \\
&= \frac{|\mathcal{C}_4^{(t)}|}{2}\left(\frac{\tau^{(t)}}{M_\Delta}\prod_{i=0}^{3}\frac{t-1-i}{M_e-i}-1\right) + \sum_{\lambda\in\Delta^{(t)}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\lambda_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\lambda_1}\right]\right) \\
&+ \sum_{\substack{\lambda,\gamma\in\Delta^{(t)}\\ \lambda\neq\gamma}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_1}\right]+\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_1}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_2}\right]\right) \qquad (4) \\
&\leq \frac{|\mathcal{C}_4^{(t)}|}{2}\left(\frac{\tau^{(t)}}{M_\Delta}c\left(\frac{t-1}{M_e}\right)^4-1\right) + \sum_{\lambda\in\Delta^{(t)}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\lambda_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\lambda_1}\right]\right) \\
&+ \sum_{\substack{\lambda,\gamma\in\Delta^{(t)}\\ \lambda\neq\gamma}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_1}\right]+\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_1}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_2}\right]\right) \quad . \qquad (5)
\end{aligned}$$

We now proceed to analyze the various covariance terms appearing in (4). In the following we refer to

- The second summation in (4), $\sum_{\lambda\in\Delta^{(t)}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\lambda_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\lambda_1}\right]\right)$, concerns the sum of the covariances of pairs of random variables each corresponding to one of the two possible ways of detecting a 4-clique using $\mathrm{TS4C}_1$. Let us consider one single element of the summation:

$$\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\lambda_2}\right] = \mathrm{E}\left[\delta_{\lambda_1}\delta_{\lambda_2}\right] - \mathrm{E}\left[\delta_{\lambda_1}\right]\mathrm{E}\left[\delta_{\lambda_2}\right] = \mathrm{E}\left[\delta_{\lambda_1}\delta_{\lambda_2}\right] - 1/4.$$

Let us now focus on $\mathrm{E}\left[\delta_{\lambda_1}\delta_{\lambda_2}\right]$, according to the definition of δ_{λ_1} and δ_{λ_2} we have:

$$\begin{aligned}
\mathrm{E}\left[\delta_{\lambda_1}\delta_{\lambda_2}\right] &= \frac{p_{\lambda_1}^{-1}p_{\lambda_2}^{-1}}{4}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}\wedge\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right) \\
&= \frac{p_{\lambda_1}^{-1}p_{\lambda_2}^{-1}}{4}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right)\Pr\left(\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right) \\
&\leq \frac{p_{\lambda_1}^{-1}p_{\lambda_2}^{-1}}{4}\Pr\left(\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right) \\
&\leq \frac{p_{\lambda_1}^{-1}p_{\lambda_2}^{-1}}{4}p_{\lambda_2} \\
&\leq \frac{p_{\lambda_1}^{-1}}{4}.
\end{aligned}$$

We can therefore conclude:

$$\sum_{\lambda\in\Delta^{(t)}}\left(\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\lambda_2}\right]+\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\lambda_1}\right]\right)\leq\frac{|\mathcal{C}_4^{(t)}|}{2}\left(\frac{\tau^{(t)}}{M_\Delta}\prod_{i=0}^{3}\frac{t-1-i}{M_e-i}-1\right)\leq\frac{|\mathcal{C}_4^{(t)}|}{2}\left(\frac{\tau^{(t)}}{M_\Delta}c\left(\frac{t-1}{M_e}\right)^4-1\right)\quad(6)$$

- The third summation in (4), includes the covariances of all $|\mathcal{C}_4^{(t)}|\left(2|\mathcal{C}_4^{(t)}|-1\right)$ unordered pairs of random variables corresponding each to one of the two possible ways of counting distinct 4-cliques in $\mathcal{C}_4^{(t)}$. In order to provide a significant bound it is necessary to divide the possible pairs of 4-cliques depending on how many edges they share (if any). From Lemma A.3 we have that any pair of 4-cliques λ and γ can share either one, three or no edges. In the remainder of our analysis we shall distinguishing three group of pairs of 4-cliques based on how many edges they share. In the following, we present, without loss of generality, bounds for $\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_2}\right]$. The results steadily holds for the other possible combinations $\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_1}\right]$, $\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_1}\right]$, $\mathrm{Cov}\left[\delta_{\lambda_2},\delta_{\gamma_2}\right]$, $\mathrm{Cov}\left[\delta_{\gamma_1},\delta_{\lambda_1}\right]$, $\mathrm{Cov}\left[\delta_{\gamma_1},\delta_{\lambda_2}\right]$, $\mathrm{Cov}\left[\delta_{\gamma_2},\delta_{\lambda_1}\right]$, and $\mathrm{Cov}\left[\delta_{\gamma_2},\delta_{\lambda_2}\right]$
  1) λ and γ do not share any edge:

$$\begin{aligned}\mathrm{E}\left[\delta_{\lambda_1}\delta_{\gamma_2}\right]&=\frac{p_{\lambda_1}^{-1}p_{\gamma_2}^{-1}}{4}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}\wedge\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)\\&=\frac{p_{\lambda_1}^{-1}p_{\gamma_2}^{-1}}{4}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)\Pr\left(\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right)\end{aligned}$$

     The term $\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)$ denotes the probability of TS4C$_1$ observing λ using T_1 and edges e_3 and e_5 conditioned of the fact that γ was observed by the algorithm using T_3 and edges g_3 and g_5. Note that as λ and γ do not share any edge, no edge of γ will be used by TS4C$_1$ to detect λ. Rather, if any edge of γ is included in $\S_e$ or if T_3 is included in $\mathcal{S}_\Delta$, this would lessen the probability of TS4C$_1$ detecting λ using $T1, e_3$ and e_5 as some of space in $\mathcal{S}_e$ or $\mathcal{S}_\Delta$ may be occupied by edges or triangle sub-structures for γ. Therefore we have $\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)\leq\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}\right)$ and thus:

$$\begin{aligned}\mathrm{E}\left[\delta_{\lambda_1}\delta_{\gamma_2}\right]&=\frac{p_{\lambda_1}^{-1}p_{\gamma_2}^{-1}}{4}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)\Pr\left(\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right)\\&\leq\frac{p_{\lambda_1}^{-1}p_{\gamma_2}^{-1}}{4}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}\right)\Pr\left(\delta_{\lambda_2}=p_{\lambda_2}^{-1}\right)\\&\leq\frac{p_{\lambda_1}^{-1}p_{\gamma_2}^{-1}}{4}p_{\lambda_1}p_{\lambda_2}\\&\leq\frac{1}{4}\end{aligned}$$

     We therefore have $\mathrm{Cov}\left[\delta_{\lambda_1},\delta_{\gamma_2}\right]\leq 0$. Hence we can conclude that the contribution of the covariances of the pairs of random variables corresponding to 4-cliques that do not share any edge to the summation in (4) is less or equal to zero.
  2) λ and γ share exactly one edge $e^*=\lambda\cap\gamma$ as shown in Figure 7 Let us consider the event $E*=$"$e^*\cap T_1\cap E^{(t_{1,2,4}-1)}\in\mathcal{S}_e^{t_{1,2,4}}\cap$, and $e^*\cap\{e_3,e_5\}\cap E^{(t_6-1)}\in S_e^{(t_6)}$ if $e^*\in\{e_3,e_5\}$". Clearly $\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)\leq\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|E*\right)$. Recall from Lemma A.2 that $\Pr\left(\delta_{\lambda_1}|E^*\right)=\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}|E^*\right)\Pr\left(T_1\in\mathcal{S}_\Delta^{(t_6)}|E*\right)\Pr\left(S(T_1)|E^*\right)$, where $S(T_1)$ denotes the event "T_1 is observed on the stream by TS4C$_1$". By applying the law of total probability e have that $\Pr\left(T_1\in\mathcal{S}_\Delta^{(t_6)}|E*\right)\leq\Pr\left(T_1\in\mathcal{S}_\Delta^{(t_6)}\right)$. The remaining two terms are influenced differently depending on whether $e^*\in T_1$ or $e^*\in\{e_3,e_5\}$
     - if $e^*\in T_1$: we then have $\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}|E^*\right)\leq\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}\right)$. This follows from the properties of the reservoir sampling scheme as the fact that the edge e^* is in $\mathcal{S}_e$ means that one unit of the available memory space required to hold e_3 or e_5 is occupied, at least for some time, by e^*. If e^* is

the last edge of T_1 observed in the stream we then have:

$$\begin{aligned}\Pr\left(S(T_1)|E^*\right) &= \Pr\left(\{e_1,e_2,e_4\}\setminus\{e^*\}\in\mathcal{S}_e^{(}t_1,2,4)|E^*\right)\\ &= \Pr\left(\{e_1,e_2,e_4\}\setminus\{e^*\}\in\mathcal{S}_e^{(}t_1,2,4)\right)\\ &\leq \frac{M_e}{t_{1,2,4}-1}\frac{M_e-1}{t_{1,2,4}-2}\end{aligned}$$

Suppose instead that $e*$ is *not* the last edge of T_1. Assume further , without loss of generality, that $t_2 > \max\{t_1,t_4\}$. TS4C$_1$ observes T_1 iff $\{e_1,e_4\}\in\mathcal{S}_e^{(}t_2)$. As in E^* we assume that once observed e^* is *always* maintained in $\mathcal{S}_e$ until $t_{1,2,4}$ we have:

$$\begin{aligned}\Pr\left(S(T_1)|E^*\right) &= \Pr\left(\{e_1,e_4\}\setminus\{e^*\}\in\mathcal{S}_e^{(}t_1,2,4)|E^*\right)\\ &\leq \frac{M_e-1}{t_{1,2,4}-2}\end{aligned}$$

- if $e^*\in\{e_3,e_5\}$: we then have $\Pr\left(S(T_1)|E^*\right)\leq\Pr\left(S(T_1)\right)$. This follows from the properties of the reservoir sampling scheme as the fact that the edge e^* is in $\mathcal{S}_e$ means that one unit of the available memory space required to hold the first two edges of T_1 until $t_{1,2,4}$ is occupied, at least for some time, by e^*. Further, using Lemma A.2 we have:
  - if $t_6\leq M_e$, then $\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}|E^*\right)=1$
  - if $\min\{t_3,t_5\}>t_{1,2,4}$, then $\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}|E^*\right)\geq\frac{M_e}{t_6-1}$,
  - if $\max\{t_3,t_5\}>t_{1,2,4}>\min\{t_3,t_5\}$, then $\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}|E^*\right)\geq\max\{\frac{M_e-1}{t_6-2},\frac{M_e-2}{t_{1,2,4}-3}\frac{t_{1,2,4}-1}{t_6-1}\}$,
  - otherwise $\Pr\left(\{e_3,e_5\}\subseteq\mathcal{S}_e^{(t_6)}|E^*\right)\geq\frac{M_e-2}{t_{1,2,4}-3}\frac{t_{1,2,4}-1}{t_6-1}$.

Putting together these various results we have that $p_\lambda^{(-1)}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|\delta_{\gamma_2}=p_{\gamma_2}^{-1}\right)\leq p_\lambda^{(-1)}\Pr\left(\delta_{\lambda_1}=p_{\lambda_1}^{-1}|E*\right)\leq c\frac{t_6-1}{M_e}$ with $c=s$. Hence we have $\mathrm{E}\left[\delta_{\lambda_1}\delta_{\gamma_2}\right]\leq\frac{c}{4}\frac{t_6-1}{M_e}\leq\frac{c}{4}\frac{t-1}{M_e}$. We can thus bound the contribution to the third component of (4) given by the pairs of random variables corresponding to 4-cliques that share one edge as:

$$2a^{(t)}\left(c\frac{t-1}{M_e}-1\right), \tag{7}$$

where $a^{(t)}$ denotes the number of unordered pairs of 4-cliques which share one edge in $G^{(t)}$.

3) λ and γ share three edges $\{e_1^*,e_2^*,e_3^*\}$ which form a triangle sub-structure for both λ and γ. Let us refer to Figure 8, without loss of generality let T_1 denote the triangle shared between the two cliques. We distinguish the kind of pairs for the random variables δ_{λ_i} and δ_{γ_j} cases:
   - $\delta_{\lambda_i}=p_{\lambda_i}^{-1}$ if $T_1\in\mathcal{S}_\Delta^{t_6-1}\wedge\{e_3,e_5\}\subseteq\mathcal{S}_e^{t_6-1}$ and $\delta_{\gamma_j}=p_{\gamma_j}^{-1}$ if $T_1\in\mathcal{S}_\Delta^{t_{\gamma}-1}\wedge\{g_3,g_5\}\subseteq\mathcal{S}_e^{t_{\gamma}-1}$, where t_γ denotes the time step at which the last edge of γ is observed. This is the case for which the random variables δ_{λ_i} and δ_{γ_j} corresponds to FOURESTobserving λ and γ using the *shared triangle* T_1. Let us consider the event $E*=$"$T_1\in\mathcal{S}_\Delta^{(t_6)}$. Clearly $\Pr\left(\delta_{\lambda_i}=p_{\lambda_i}^{-1}|\delta_{\gamma_j}=p_{\gamma_j}^{-1}\right)\leq\Pr\left(\delta_{\lambda_i}=p_{\lambda_i}^{-1}|E*\right)$. In this case we have $\Pr\left(T_1\in\mathcal{S}^{(t_6)}|E^*\right)\leq 1$, while $\Pr\left(\{e_3,e_5\}\in\mathcal{S}_e^{(t_6)}|E^*\right)\leq Prob\{e_3,e_5\}\in\mathcal{S}_e^{(t_6)}$. This second fact follows from the properties of the reservoir sampling scheme as the fact that the edges e_1^*, e_2^* and e_3^* are in $\mathcal{S}_e$ *at least* for the time required for T_1 to be observed, means that at least two unit of the available memory space required to hold the edges e_3,e_5 are occupied, at least for some time. Putting together these various results we have that $p_{\lambda_i}^{(-1)}Prob\delta_{\lambda_i}=p_{\lambda_i}^{-1}|\delta_{\gamma_j}=p_{\gamma_j}^{-1}\leq p_{\lambda_i}^{(-1)}\Pr\left(\delta_{\lambda_i}=p_{\lambda_i}^{-1}|E*\right)\leq c\left(\frac{t_6-1}{M_e}\right)^2\frac{\tau^{(t)}}{\mathcal{S}_\Delta}$. Hence we have $\mathrm{E}\left[\delta_{\lambda_i}\delta_{\gamma_j}\right]\leq\frac{c}{4}\left(\frac{t_6-1}{M_e}\right)^2\frac{\tau^{(t)}}{\mathcal{S}_\Delta}$ and $\mathrm{Cov}\left[\delta_{\lambda_i},\delta_{\gamma_j}\right]\leq\frac{c}{4}\left(\frac{t_6-1}{M_e}\right)^2\frac{\tau^{(t)}}{\mathcal{S}_\Delta}-\frac{1}{4}$.
   - in all the remaining case, then the random variables δ_{λ_i} and δ_{γ_j} corresponds to FOURESTnot observing λ and γ using the shared triangle T_1 for both of them. Let T^* denote the triangle sub-structure used by TS4C$_1$ to count λ with respect to δ_{λ_i}. Let us consider the event $E^*=$"$\{e_1^*,e_2^*,e_3^*\}\cap T_1\cap E^{(t_{1,2,4}-1)}\in\mathcal{S}_e^{t_{1,2,4}}$, $T_1\in\mathcal{S}_\Delta^{(t_6)}$ unless one of its edges is the last edge of T^* observed on the stream, and $\{e_1^*,e_2^*,e_3^*\}\cap$

$\{e_3, e_5\} \cap E^{(t_6-1)} \in S_e^{(t_6)}$ if $e^* \in \{e_3, e_5\}$", where $E^{(t)}$ denotes the set of edges observed up until time t included. Clearly $\Pr\left(\delta_{\lambda_i} = p_{\lambda_i}^{-1} | \delta_{\gamma_j} = p_{\gamma_j}^{-1}\right) \leq \Pr\left(\delta_{\lambda_i} = p_{\lambda_i}^{-1} | E*\right)$. Note that in this case $|\{e_1^*, e_2^*, e_3^*\} \cap T_1| + |\{e_1^*, e_2^*, e_3^*\} \cap \{e_3, e_5\}|$. By analyzing $\Pr\left(\delta_{\lambda_i} = p_{\lambda_i}^{-1} | E*\right)$ in this case using similar steps as the ones described for the other sub-cases we have $p_{\lambda_i}^{(-1)} \Pr\left(\delta_{\lambda_i} = p_{\lambda_i}^{-1} | \delta_{\gamma_j} = p_{\gamma_j}^{-1}\right) \leq p_{\lambda_i}^{(-1)} \Pr\left(\delta_{\lambda_i} = p_{\lambda_i}^{-1} | E*\right) \leq c\left(\frac{t_6-1}{M_e}\right)^3$. Hence we have $\mathrm{E}\left[\delta_{\lambda_i}\delta_{\gamma_j}\right] \leq \frac{c}{4}\left(\frac{t_6-1}{M_e}\right)^3$ and $\mathrm{Cov}\left[\delta_{\lambda_i}, \delta_{\gamma_j}\right] \leq \frac{c}{4}\left(\frac{t_6-1}{M_e}\right)^3 - \frac{1}{4}$.

We can thus bound the contribution to the third component of (4) given by the pairs of random variables corresponding to 4-cliques that share three edges as:

$$2b^{(t)}\left(c\left(\frac{t-1}{M_e}\right)^2\left(\frac{1}{4}\frac{\tau^{(t)}}{M_\Delta} + \frac{3}{4}\frac{t-1}{M_e}\right) - 1\right), \tag{8}$$

where $b^{(t)}$ denotes the number of unordered pairs of 4-cliques which share one edge in $G^{(t)}$.

The Theorem follows by combining (6), (7) and (8) in (4). ∎

Appendix B.
Additional theoretical results for TS4C$_2$

In this section we present the main analytical results for TS4C$_2$ discussed in Section IV-E. In order to simplify the presentation we use the following notation:

$$t^M_{1,2,\ldots,i} \triangleq \max\{t_1, t_2, \ldots, t_i\}$$
$$t^m_{1,2,\ldots,i} \triangleq \min\{t_1, t_2, \ldots, t_i\}$$

Lemma B.1. *Let $\lambda \in \mathcal{C}_4^{(t)}$ with $\lambda = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ using Figure 1 as reference. Assume further, without loss of generality, that the edge e_i is observed at t_i (not necessarily consecutively) and that $t_6 > \max\{t_i, 1 \leq i \leq 5\}$. λ can be observed by* TS4C$_2$ *at time t_6 only by a combination of two triangles $T_1 = \{e_1, e_2, e_4\}$ and $T_2 = \{e_1, e_3, e_5\}$.*

Proof: TS4C$_2$ can detect λ only when its last edge is observed on the stream (hence, t_6). When e_6 is observed, the algorithm first evaluates wether there are two triangles in $\mathcal{S}_\Delta^{(t_6)}$, that share two endpoints and the other endpoints are u and v respectively. Since a this step (i.e., the execution of function UPDATE4CLIQUES) the triangle sample is jet to be updated based on the observation of e_6, the only triangle sub-structures of λ which may have been observed on the stream, and thus included in $\mathcal{S}_\Delta^{(t_6)}$ are $T_1 = \{e_1, e_2, e_4\}$ and $T_2 = \{e_1, e_3, e_5\}$. λ is thus observed if and only if both of the triangles T_1 and T_2 are found in $\mathcal{S}_\Delta^{(t_6)}$. ∎

Lemma B.2. *Let $\lambda \in \mathcal{C}_4^{(t)}$ with $\lambda = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ using Figure 1 as reference. Assume further, without loss of generality, that the edge e_i is observed at t_i (not necessarily consecutively) and that $t_6 > \max\{t_i, 1 \leq i \leq 5\}$. The probability p_λ of λ being observed on the stream by* TS4C$_2$*, is computed by* PROBCLIQUE *as:*

$$p_\lambda = \min\{1, \frac{M_e}{t^M_{1,3,5} - 1}\frac{M_e - 1}{t^M_{1,3,5} - 2}\}\min\{1, \frac{M_\Delta}{\tau^{(t_6)}}\frac{M_\Delta - 1}{\tau^{(t_6)} - 1}\}p'$$

where:

$$p' = \begin{cases} 1, & if\ t^M_{1,2,4} \leq M_e \\ \frac{M_e-2}{t_1-3}\frac{M_e-3}{t_1-4}, & if\ t_1 > t^M_{2,3,4,5} \\ \frac{M_e-2}{t_1-3}\frac{M_e-3}{t_1-4}, & if\ t^M_{3,5} > t_1 > \max\{t^m_{3,5}, t_2, t_4\} \\ \frac{M_e-2}{t^M_{2,4}-3}, & if\ t^M_{3,5} > t^M_{2,4} > \max\{t^m_{3,5}, t^m_{2,4}, t_1\} \\ \frac{M_e}{t_1-1}\frac{M_e-1}{t_1-2}, & if\ t^m_{3,5} > t_1 > t^M_{2,4} \\ \frac{M_e-1}{t^M_{2,4}-2} & if\ t^m_{3,5} > t^M_{2,4} > t_1 \\ \frac{M_e-1}{t^M_{2,4}-1}\frac{M_e-2}{t_1-3}\frac{t_1-1}{t_2-1}, & if\ t^M_{2,4} > t_1 > \max\{M_e, t^m_{2,4}, t^M_{3,5}\} \\ \frac{M_e-1}{t^M_{2,4}-1}\frac{M_e}{t_2-1} & if\ t^M_{2,4} > M_e > t_1 > \max\{t^m_{2,4}, t^M_{3,5}\} \\ \frac{t_3-1}{t^M_{2,4}-1}\frac{t_3-2}{t^M_{2,4}-2}\frac{M_e-2}{t^M_{3,5}-3}, & if\ t^M_{2,4} > t^M_{3,5} > \max\{M_e, t^m_{3,5}, t^m_{2,4}, t_1\} \\ \frac{M_e}{t^M_{2,4}-1}\frac{M_e-1}{t^M_{2,4}-2}, & if\ t^M_{2,4} > M_e > t^M_{3,5} > \max\{t^m_{3,5}, t^m_{2,4}, t_1\} \\ \frac{M_e}{t^M_{2,4}-1}\frac{M_e-1}{t^M_{2,4}-2}, & if\ t^m_{2,4} > t_1 > t^M_{3,5} \\ \frac{M_e-1}{t^M_{2,4}-2}\frac{t^M_{3,5}-1}{t^M_{2,4}-1}, & if\ t^m_{2,4} > t^M_{3,5} > \max\{M, t_1\} \\ \frac{M_e-1}{t^M_{2,4}-2}\frac{M_e}{t^M_{2,4}-1}, & if\ t^m_{2,4} > M_e > t^M_{3,5} > t_1 \end{cases}$$

Proof: Let us define the event $E_\lambda = \lambda$ *is observed* on the stream by TS4C$_2$ using triangle $T_1 = \{e_1, e_2, e_4\}$ and triangle $T_2 = \{e_1, e_3, e_5\}$. Given the definition of TS4C$_2$ we have:

$$E_\lambda = E_{T_1} \wedge E_{T_2},$$

and hence:

$$p_\lambda = \Pr(E_\lambda) = \Pr(E_{T_1} \wedge E_{T_2}) = \Pr\left(E_{T_1}|E_{T_2}\right)\Pr(E_{T_2}).$$

In oder to study $\Pr(E_{T_2})$ we shall introduce event $E_{S(T_2)}$ ="*triangle* T_2 *is observed on the stream by* TS4C$_2$". From the definition of TS4C$_2$ we know that T_2 is observed on the stream iff when the last edge of T_2 is observed on the stream at $\max\{t_1, t_3, t_5\}$ the remaining to edges are in the edge sample. Applying Bayes's rule of total probability we have:

$$\Pr(E_{T_2}) = \Pr\left(E_{T_2}|E_{S(T_2)}\right)\Pr\left(E_{S(T_2)}\right).$$

and thus:

$$p_\lambda = \Pr\left(E_{T_1}|E_{T_2}\right)\Pr\left(E_{T_2}|E_{S(T_2)}\right)\Pr\left(E_{S(T_2)}\right). \tag{9}$$

In order for T_2 to be observed by $TS4c_2$ it is required that wen the last edge of T_2 is observed on the stream at $t^M_{1,3,5}$ its two remaining edges are kept in S_e. Lemma II.1 we have:

$$\Pr\left(E_{S(T_2)}\right) = \frac{M_e}{t_{1,3,5}-1}\frac{M_e-1}{t_{1,3,5}-2}, \tag{10}$$

and:

$$\Pr\left(E_{T_2}|E_{S(T_2)}\right) = \frac{M_e}{\tau^{t_6}}. \tag{11}$$

Let us now consider $\Pr\left(E_{T_1}|E_{T_2}\right)$. In order for T_1 to be found in $\mathcal{S}_\Delta$ at t_6 it is necessary for T_1 to have been observed by TS4C$_2$. We thus have that $\Pr\left(E_{T_1}|E_{T_2}\right) = \Pr\left(E_{T_1}|E_{S(T_1)}, E_{T_2}\right)\Pr\left(E_{S(T_1)}|E_{T_2}\right)$

$$\Pr\left(E_{T_1}|E_{S(T_1)}\right) = \frac{M_\Delta - 1}{\tau^{t_6} - 1} \tag{12}$$

Let us now consider $\Pr\left(E_{S(T_1)}|E_{T_2}\right)$. while the content of $\mathcal{S}_\Delta$ itself does not influence the content of $\mathcal{S}_e$, the fact that T_2 is maintained in $\mathcal{S}_\Delta$ at t_6 implies that it has been observed on the stream at a previous time. We thus have $\Pr\left(E_{S(T_1)}|E_{T_2}\right) = \Pr\left(E_{S(T_1)}|E_{S(T_2)}\right)$. In order to study $p' = \Pr\left(E_{S(T_1)}|E_{S(T_2)}\right)$ it is necessary to distinguish the possible (5!) different arrival orders for edges e_1,e_2, e_3, e_4 and e_5. An efficient analysis we however reduce the number of cases to be considered to thirteen.

- $t^M_{1,2,4} \leq M_e$: in this case all edges of T_1 are observed on the stream before M_e so they are deterministically inserted in $\mathcal{S}_e$ and thus $p' = 1$.
- $t_1 > t^M_{2,3,4,5}$: in this case the edge e_1 that is shared by T_1 and T_2 is observed after e_2, e_3, e_4, e_5.

$$\begin{aligned} p' &= P(\{e_2, e_4\} \in {\mathcal{S}_e}^{(t_1)} | \{e_3, e_5\} \in {\mathcal{S}_e}^{(t_1)}) \\ &= P(e_2 \in \mathcal{S}_e(t_1) | \{e_3, e_4, e_5\} \in {\mathcal{S}_e}^{(t_1)}) \cdot P(e_4 \in {\mathcal{S}_e}^{(t_1)} | \{e_3, e_5\} \in {\mathcal{S}_e}^{(t_1)}) \\ &= min(1, \frac{M-3}{t_1-4}) \cdot min(1, \frac{M-2}{t_1-3}) \end{aligned}$$

- $t^M_{3,5} > t_1 > \max\{t^m_{3,5}, t_2, t_4\}$: in this case only one of the edges e_3,e_5 is observed after e_1, which is observed after all the remaining edges. Here we consider the case when e_3 is the edge to be observed last. The same procedure follows for e_5 as well.

$$\begin{aligned} p' &= P(\{e_2, e_4\} \in {\mathcal{S}_e}^{(t_1)} | e_5 \in {\mathcal{S}_e}^{(t_1)}) \\ &= P(e_2 \in \mathcal{S}_e(t_1) | \{e_4, e_5\} \in {\mathcal{S}_e}^{(t_1)}) \cdot P(e_4 \in {\mathcal{S}_e}^{(t_1)} | e_5 \in {\mathcal{S}_e}^{(t_1)}) \\ &= min(1, \frac{M-2}{t_1-3}) \cdot min(1, \frac{M-1}{t_1-2}) \end{aligned}$$

- $t^M_{3,5} > t^M_{2,4} > \max\{t^m_{3,5}, t^m_{2,4}, t_1\}$: in this case only one of the edges e_3, e_5 is observed after one of the edges e_2, e_4, which is observed after all the remaining edges. We consider the case when e_3 and e_2 are observed last. The same procedure follows for e_4 and e_5 as well.

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in {\mathcal{S}_e}^{(t_2)} | \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \\ &= P(e_1 \in \mathcal{S}_e(t_2) | \{e_1, e_4, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \cdot P(e_4 \in {\mathcal{S}_e}^{(t_2)} | \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \\ &= 1 \cdot min(1, \frac{M-2}{t_2-3}) \end{aligned}$$

- $t^m_{3,5} > t_1 > t^M_{2,4}$: in this case both edges e_3,e_5 are observed after e_1, which is observed after all the remaining edges. Here we consider the case when $t_3 > t_5 > t_1$. The same procedure follows for $t_5 > t_3 > t_1$.

$$\begin{aligned} p' &= P(\{e_2, e_4\} \in {\mathcal{S}_e}^{(t_1)} | \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \\ &= P(e_2 \in \mathcal{S}_e(t_1) | e_4 \in \mathcal{S}_e(t_1), \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \cdot P(e_4 \in {\mathcal{S}_e}^{(t_1)} | \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \\ &= min(1, \frac{M-1}{t_1-2}) \cdot min(1, \frac{M}{t_1-1}) \end{aligned}$$

- $t^m_{3,5} > t^M_{2,4} > t_1$: in this case both edges e_3,e_5 are observed after one of the edges e_2,e_4, which is observed after e_1. We consider the case $t_2 > t_4$ and $t_3 > t_5$. The same procedure follows for $t_4 > t_2$ and $t_5 > t_3$

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in {\mathcal{S}_e}^{(t_2)} | \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \\ &= P(e_4 \in \mathcal{S}_e(t_2) | e_1 \in \mathcal{S}_e(t_2), \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \cdot P(e_1 \in {\mathcal{S}_e}^{(t_3)} | \{e_1, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \\ &= min(1, \frac{M-1}{t_2-2}) \cdot 1 \end{aligned}$$

- $t^M_{2,4} > t_1 > \max\{M_e, t^m_{2,4}, t^M_{3,5}\}$: in this case both edges e_2, e_4 are observed after e_1, which is observed after the edge reservoir is filled and after all the remaining edges. We consider the case when $t_2 > t_4$. The same procedure follows for $t_4 > t_2$.

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in {\mathcal{S}_e}^{(t_2)} | \{e_3, e_5\} \in {\mathcal{S}_e}^{(t_1)}) \\ &= P(e_4 \in \mathcal{S}_e(t_2) | e_1 \in \mathcal{S}_e(t_2), \{e_3, e_5\} \in {\mathcal{S}_e}^{(t_3)}) \cdot P(e_1 \in {\mathcal{S}_e}^{(t_2)} | \{e_3, e_5\} \in {\mathcal{S}_e}^{(t_1)}) \\ &= \frac{M-1}{t_2-2} \cdot \frac{M-2}{t_1-3} \cdot \frac{t_1-1}{t_2-1} \end{aligned}$$

- $t^M_{2,4} > M_e > t_1 > \max\{t^m_{2,4}, t^M_{3,5}\}$:in this case both edges e_2, e_4 are observed after e_1, which is observed before the edge reservoir is filled and after all the remaining edges. We consider the case when $t_2 > t_4$. The same

procedure follows for $t_4 > t_2$.

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in \mathcal{S}_e{}^{(t_2)} | \{e_3, e_5\} \in \mathcal{S}_e{}^{(t_1)}) \\ &= P(e_4 \in \mathcal{S}_e(t_2) | e_1 \in \mathcal{S}_e(t_2), \{e_3, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \cdot P(e_1 \in \mathcal{S}_e{}^{(t_2)} | \{e_3, e_5\} \in \mathcal{S}_e{}^{(t_1)}) \\ &= \frac{M-1}{t_2-2} \cdot \frac{M_e}{t_2-1} \end{aligned}$$

- $t^M_{2,4} > t^M_{3,5} > \max\{M_e, t^m_{3,5}, t^m_{2,4}, t_1\}$: in this case only one of the edges e_2, e_4 is observed after one of the edges e_3, e_5 which is observed after the edge reservoir if filled and after all the remaining edges. We consider the case when $t_2 > t_4$ and $t_3 > t_5$. The same procedure follows for $t_4 > t_2$ and $t_5 > t_3$

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= P(e_1 \in \mathcal{S}_e(t_2) | e_4 \in \mathcal{S}_e(t_2), \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \cdot P(e_4 \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= \frac{t_3-1}{t_2-1} \cdot \frac{t_3-2}{t_2-2} \cdot \frac{M_e-2}{t_3-3} \end{aligned}$$

- $t^M_{2,4} > M_e > t^M_{3,5} > \max\{t^m_{3,5}, t^m_{2,4}, t_1\}$: in this case only one of the edges e_2, e_4 is observed after one of the edges e_3, e_5 which is observed before the edge reservoir if filled and after all the remaining edges. We consider the case when $t_2 > t_4$ and $t_3 > t_5$. The same procedure follows for $t_4 > t_2$ and $t_5 > t_3$

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= P(e_1 \in \mathcal{S}_e(t_2) | e_4 \in \mathcal{S}_e(t_2), \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \cdot P(e_4 \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= \frac{M_e}{t_2-1} \cdot \frac{M_e-1}{t_2-2} \end{aligned}$$

- $t^m_{2,4} > t_1 > t^M_{3,5}$: in this case both edges e_2,e_4 are observed after e_1, which is observed after all the remaining edges. Here we consider the case when $t_2 > t_4 > t_1$. The same procedure follows for $t_4 > t_2 > t_1$.

$$\begin{aligned} p' &= P(\{e_2, e_4\} \in \mathcal{S}_e{}^{(t_1)} | \{e_3, e_5\} \in \mathcal{S}_e{}^{(t_1)}) \\ &= P(e_2 \in \mathcal{S}_e(t_1) | e_4 \in \mathcal{S}_e(t_1), \{e_3, e_5\} \in \mathcal{S}_e{}^{(t_1)}) \cdot P(e_4 \in \mathcal{S}_e{}^{(t_1)} | \{e_3, e_5\} \in \mathcal{S}_e{}^{(t_1)}) \\ &= min(1, \frac{M_e-1}{t_2-2}) \cdot min(1, \frac{M_e}{t_2-1}) \end{aligned}$$

- $t^m_{2,4} > t^M_{3,5} > \max\{M, t_1\}$: in this case both edges e_2,e_4 are observed after one of the edges e_3,e_5, which is observed after the edge reservoir is filled and after all the remaining edges.

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= P(e_4 \in \mathcal{S}_e(t_2) | e_1 \in \mathcal{S}_e(t_2), \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \cdot P(e_1 \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= \frac{M_e-1}{t_2-2} \cdot \frac{t_3-1}{t_2-1} \end{aligned}$$

- $t^m_{2,4} > M_e > t^M_{3,5} > t_1$: in this case both edges e_2,e_4 are observed after one of the edges e_3,e_5, which is observed before the edge reservoir is filled and after all the remaining edges.

$$\begin{aligned} p' &= P(\{e_1, e_4\} \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= P(e_4 \in \mathcal{S}_e(t_2) | e_1 \in \mathcal{S}_e(t_2), \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \cdot P(e_1 \in \mathcal{S}_e{}^{(t_2)} | \{e_1, e_5\} \in \mathcal{S}_e{}^{(t_3)}) \\ &= \frac{M_e-1}{t_2-2} \cdot \frac{M_e}{t_2-1} \end{aligned}$$

The lemma follows combining the result for the values of p' with (10), (11) and (12) in (B). ∎

We now present the *unbiasedness* of the estimations obtained using TS4C$_2$.

Theorem B.3. *The estimator $\varkappa$ returned by* TS4C$_2$ *is* unbiased, *that is:* $\mathbb{E}\left[\varkappa^{(t)}\right] = |\mathcal{C}^{(t)}_k|$.

Proof: Let t^* denote the first step at for which the number of triangle seen exceeds M_Δ. For $t \leq \min\{M_e, t^*\}$, the entire graph $G^{(t)}$ is maintained in $\mathcal{S}_e$ and all the triangles in $G^{(t)}$ are stored in $\mathcal{S}_\Delta$. Hence all the cliques in $G^{(t)}$ are deterministically observed by TS4C$_1$and we have $\varkappa^{(t)} = |\mathcal{C}^{(t)}_4|$.

Assume now $t > \min\{M_e, t^*\}$ and assume that $|\mathcal{C}_4^{(t)}| > 0$, otherwise, the algorithm deterministically returns 0 as an estimation and the thesis follows. Recall that every time TS4C$_1$observes a 4-clique on the stream it evaluates the the probability p of observing it and it correspondingly increases the running estimator by p^{-1}. For any 4-clique $\lambda \in \mathcal{C}_4^{(t)}$ which is observed by TS4C$_1$with probability p_λ, consider a random variable X_λ which takes value p^{-1} iff λ is acutally observed by TS4C$_1$(i.e., with probability p_λ) or zero otherwise. We thus have $\mathbb{E}\left[X_\lambda\right] = p_\lambda^{-1}\Pr\left(X_\lambda = p_\lambda^{-1}\right) = p_\lambda^{-1}p_\lambda = 1$. Recall that every time TS4C$_1$observes a 4-clique on the stream it evaluates the the probability p of observing it, and it correspondingly increases the running estimator $\varkappa$ by p^{-1}. We therefore can express the running estimator $\varkappa^{(t)}$ as:

$$\varkappa^{(t)} = \sum_{\lambda \in \mathcal{C}_4^{(t)}} X_\lambda \ .$$

From linearity of expectation, we thus have:

$$\mathbb{E}\left[\varkappa^{(t)}\right] = \sum_{\lambda \in \mathcal{C}_4^{(t)}} \mathbb{E}[X_\lambda] = \sum_{\lambda \in \mathcal{C}_4^{(t)}} p_\lambda^{-1} p_\lambda = |\mathcal{C}_4^{(t)}|.$$

■