TRIÈST: Counting Local and Global Triangles in Fully-Dynamic Streams with Fixed Memory Size

TRIÈST：使用固定大小的内存计算全动态流中的局部和全局三角形

**ABSTRACT：**

We present trièst, a suite of one-pass streaming algorithms to compute unbiased, low-variance, high-quality approximations of the global and local (i.e., incident to each vertex) number of triangles in a fully-dynamic graph represented as an adversarial stream of edge insertions and deletions.

我们提出了trièst，这是一套单通道流算法，用于计算全局和局部（即每个顶点的附带）三角形数量的无偏差、低变异、高质量的近似值，全动态图被表示为边缘插入和删除的对抗性流。

Our algorithms use reservoir sampling and its variants to exploit the user-specified memory space at all times. This is in contrast with previous approaches, which require hard-to-choose parameters (e.g., a fixed sampling probability) and offer no guarantees on the amount of memory they use. We analyze the variance of the estimations and show novel con- centration bounds for these quantities.

我们的算法使用水库取样及其变体，在任何时候都能利用用户指定的内存空间。这与以前的方法形成鲜明对比，以前的方法需要难以选择的参数（例如，固定的采样概率），而且对它们使用的内存量没有任何保证。我们分析了估计的方差，并展示了这些数量的新的集中界限。

Our experimental results on very large graphs demonstrate that trièst outperforms state-of-the-art approaches in accuracy and exhibits a small update time.

我们在非常大的图上的实验结果表明，trièst在准确性上优于最先进的方法，并表现出较小的更新时间。

**INTRODUCTION**

Exact computation of characteristic quantities of Web-scale networks is often impractical or even infeasible due to the humongous size of these graphs. It is natural in these cases to resort to efficient-to-compute approximations of these quantities, which, when of sufficiently high quality, can be used as proxies for the exact values.

由于网络规模的庞大，精确计算网络的特征量往往是不切实际的，甚至是不可行的。在这种情况下，自然要借助于这些数量的高效计算近似值，如果质量足够高，可以作为精确值的代理。

In addition to being huge, many interesting networks are fully-dynamic and can be represented as a stream whose elements are edges/nodes insertions and deletions occurring in an arbitrary (even adversarial) order. Characteristic quantities in these graphs are intrinsically volatile, hence there is limited added value in maintaining them exactly. The goal is rather to keep track, at all times, of a high-quality approximation of these quantities. For efficiency, the algorithms should aim at exploiting the available memory space as much as possible and they should require only one pass over the stream.

除了巨大之外，许多有趣的网络是完全动态的，可以表示为一个流，其元素是以任意的（甚至是对抗性的）顺序发生的边/节点插入和删除。这些图中的特征量在本质上是不稳定的，因此准确地维护它们的附加价值是有限的。我们的目标是在任何时候都保持这些数量的高质量近似值。为了提高效率，算法应该尽可能地利用可用的内存空间，并且只需要对数据流进行一次处理。

We introduce trièst, a suite of sampling-based, one-pass algorithms for adversarial fully-dynamic streams to approximate the global number of triangles and the local number of triangles incident to each vertex. Mining local and global triangles is a fundamental primitive with many applications (e.g., community detection [4], topic mining [10], spam/anomaly detection [3, 27], ego-networks mining [12] and protein interaction networks analysis [29].)

我们介绍了trièst，这是一套基于抽样的单通道算法，用于对抗性全动态流，以近似计算三角形的全局数量和每个顶点附带的三角形的局部数量。挖掘局部和全局三角形是一个基本的基元，有许多应用（例如，社区检测[4]，主题挖掘[10]，垃圾邮件/异常检测[3，27]，自我网络挖掘[12]和蛋白质交互网络分析[29]）。

Many previous works on triangle estimation in streams also employ sampling (see Sect. 3), but they usually require the user to specify in advance an edge sampling probability p that is fixed for the entire stream. This approach presents several significant drawbacks. First, choosing a p that allows to obtain the desired approximation quality requires to know or guess a number of properties of the input (e.g., the size of the stream). Second, a fixed p implies that the sample size grows with the size of the stream, which is problematic when the stream size is not known in advance: if the user specifies a large p, the algorithm may run out of memory, while for a smaller p it will provide a suboptimal estimation. Third, even assuming to be able to compute a p that ensures (in expectation) full use of the available space, the memory would be fully utilized only at the end of the stream, and the estimations computed throughout the execution would be suboptimal.

以前许多关于流中三角形估计的工作也采用了抽样（见第3节），但它们通常要求用户事先指定一个对整个流来说是固定的边缘抽样概率p。这种方法有几个明显的缺点。首先，选择一个允许获得所需近似质量的p需要知道或猜测输入的一些属性（例如，流的大小）。第二，一个固定的p意味着样本大小随着数据流的大小而增长，当数据流的大小事先不知道时，这就有问题了：如果用户指定一个大的p，算法可能会耗尽内存，而对于一个小的p，它将提供一个次优的估计。第三，即使假设能够计算出一个确保（预期）充分利用可用空间的p，内存也只有在流结束时才会被完全利用，而在整个执行过程中计算出的估计值将是次优的。

**Contributions.** We address all the above issues by taking a significant departure from the fixed-probability, independent edge sampling approach taken even by state-of-the-art methods [27]. Specifically

• We introduce trièst (TRI angle Estimation from ST reams), a suite of one-pass streaming algorithms to approximate, at each time instant, the global and local number of triangles in a fully-dynamic graph stream (i.e., a sequence of edges additions and deletions in arbitrary order) using a fixed amount of memory. This is the first contribution that enjoys all these properties. trièst only requires the user to specify the amount of available memory, an interpretable parameter that is definitively known to the user.

- 我们介绍了trièst（TRI angle Estimation from ST reams），这是一套单程流算法，在每个时间瞬间，使用固定数量的内存对全动态图流（即以任意顺序增加和删除边缘的序列）中的全局和局部三角形的数量进行近似计算。这是第一个享有所有这些特性的贡献。trièst只要求用户指定可用的内存量，这是一个用户明确知道的可解释参数。

• Our algorithms maintain a sample of edges: they use the reservoir sampling [37] and random pairing [14] sampling schemes to exploit the available memory as much as possible. To the best of our knowledge, ours is the first application of these techniques to subgraph counting in fully-dynamic, arbitrarily long, adversarially ordered streams. We present an analysis of the unbiasedness and of the variance of our estimators, and establish strong concentration results for them. The use of reservoir sampling and random pairing requires additional sophistication in the analysis, as the presence of an edge in the sample is not independent from the concurrent presence of another edge. Hence, in our proofs we must consider the complex dependencies in events involving sets of edges. The gain is worth the effort: we prove that the variance of our algorithms is smaller than that of state-of-the-art methods [27], and this is confirmed by our experiments.

- 我们的算法保持了一个边缘的样本：他们使用水库采样[37]和随机配对[14]的采样方案来尽可能地利用可用的内存。据我们所知，我们是第一次将这些技术应用于完全动态的、任意长的、有对抗性的数据流中的子图计数。我们提出了对我们估计器的无偏性和方差的分析，并为它们建立了强有力的集中结果。使用蓄水池抽样和随机配对需要在分析中增加复杂性，因为样本中一条边的存在与另一条边的同时存在并不独立。因此，在我们的证明中，我们必须考虑涉及边集的事件中的复杂依赖关系。收益是值得的：我们证明了我们算法的方差小于最先进的方法[27]，这一点在我们的实验中得到了证实。

• We conduct an extensive experimental evaluation of trièst on very large graphs, some with billions of edges, comparing the performances of our algorithms to those of existing state-of-the-art contributions [18, 27, 32]. Our algorithms significantly and consistently reduce the average estimation error by up to 90% w.r.t. the state of the art, both in the global and local estimation problems, while using the same amount of memory. Our algorithms are also extremely scalable, showing update times in the order of hundreds of microseconds for graphs with billions of edges. Due to space constraints, the proofs and additional experimental results can be found in the extended version

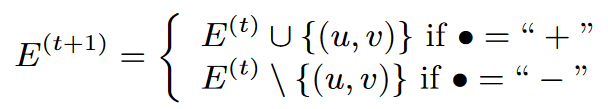
- 我们在非常大的图上对trièst进行了广泛的实验评估，有些图有数十亿条边，将我们的算法的性能与现有的最先进的贡献[18, 27, 32]进行了比较。在全局和局部估计问题上，我们的算法在使用相同内存量的情况下，与现有技术水平相比，明显且持续地减少了高达90%的平均估计误差。我们的算法还具有极强的可扩展性，对于具有数十亿条边的图形，显示出数百微秒的更新时间。由于篇幅限制，证明和其他实验结果可以在扩展版本中找到

PRELIMINARIES: 初步了解

We study the problem of counting global and local triangles in a fully-dynamic undirected graph as an arbitrary (adversarial) stream of edge insertions and deletions.

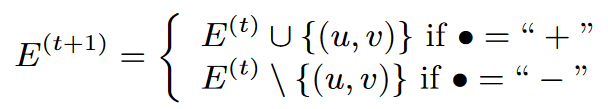
我们研究在一个完全动态的无向图中计算全局和局部三角形的问题，作为一个任意的（对抗性的）边缘插入和删除的流。

Formally, for any (discrete) time instant , let be the graph observed up to and including time . At time t = 0 we have . For any , at time we receive an element from a stream, where and are two distinct vertices. The graph is obtained by inserting a new edge or deleting an existing edge as follows:



If or do not belong to , they are added to . Nodes are deleted from V (t) when they have degree zero.

形式上，对于任何（离散的）时间瞬间, 让是到时间t为止观察到的图，包括时间t。在时间t=0时，我们有.。对于任何，在时，我们从一个流中收到一个元素，其中，是两个不同的顶点。图是通过插入一条新边或删除一条现有边得到的。



如果u或v不属于它们将被添加到。当节点的度数为零时，则从中删除。

Edges can be added and deleted in the graph in an arbitrary adversarial order, i.e., as to cause the worst outcome for the algorithm, but we assume that the adversary has no access to the random bits used by the algorithm. We assume that all operations have effect: if (resp. ),(resp.) can not be on the stream at time.

图中的边可以按照任意的对抗性顺序添加和删除，即导致算法的最坏结果，但我们假设对抗者无法获得算法使用的随机比特。我们假设所有的操作都有效果：如果（(resp. ），(resp.)不可能在时间的流上。

**Problem definition.** We study the Global (resp. Local) Triangle Counting Problem in Fully-dynamic Streams, which requires to compute, at each time an estimation of (resp. for each an estimation of ).

问题的定义。我们研究全动态流中的Global（resp. Local）三角形计数问题，它要求在每个时间计算的估计（resp.对于每个计算的估计）。

**RELATED WORK**

The literature on triangle counting is extremely rich, including exact algorithms, graph sparsifiers [35, 36], complex-valued sketches [20, 28], and MapReduce algorithms [30, 31, 33]. Here we restrict the discussion to the works most related to ours, i.e., to those presenting algorithms for counting or approximating the number of triangles from data streams. We refer to the survey by Latapy [25] for an in-depth discussion of other works.

关于三角形计数的文献非常丰富，包括精确算法、图疏导器[35, 36]、复值草图[20, 28]和MapReduce算法[30, 31, 33]。在此，我们将讨论限制在与我们最相关的工作上，即那些提出计算或近似计算数据流中三角形数量的算法。关于其他作品的深入讨论，我们参考Latapy的调查报告[25]。

Many previous contributions presented algorithms for more restricted (i.e., less generic) settings than ours, or for which the constraints on the computation are more lax [2, 6, 19, 22]. For example, Becchetti et al. [3] and Kolountzakis et al. [21] present algorithms for approximate triangle counting from static graphs by performing multiple passes over the input. Pavan et al. [32] and Jha et al. [18] propose algorithms for approximating only the global number of triangles from edge-insertion-only streams. Kutzkov and Pagh [23] present a one-pass algorithm for fully-dynamic graphs, but the triangle count estimation is (expensively) computed only at the end of the stream and the algorithm requires, in the worst case, more memory than what is needed to store the entire graph. Ahmed et al. [1] apply the sampling-and-hold approach to insertion-only graph stream mining to obtain, only at the end of the stream and using non-constant space, an estimation of many network measures including triangles.

许多以前的贡献提出了比我们更多限制（即不太通用）的算法，或者对计算的约束更宽松的算法[2, 6, 19, 22]。例如，Becchetti等人[3]和Kolountzakis等人[21]提出了通过对输入进行多次处理来实现静态图的近似三角形计算的算法。Pavan等人[32]和Jha等人[18]提出了仅从纯边缘插入流中近似计算三角形的全局数量的算法。Kutzkov和Pagh[23]提出了一个适用于全动态图的单通道算法，但是三角形数量的估计只在流的末端进行（昂贵的）计算，在最坏的情况下，该算法需要的内存比存储整个图所需要的还要多。Ahmed等人[1]将采样和保持方法应用于仅插入式图流的挖掘，仅在图流的末端和使用非恒定空间，获得包括三角形在内的许多网络措施的估计。

None of these works has all the features offered by trièst: performs a single pass over the data, handles fully-dynamic streams, uses a fixed amount of memory space, requires a single interpretable parameter, and returns an estimation at each time instant. Furthermore, our experimental results show that we outperform the algorithms from [18, 32] on insertion-only streams.

这些工作都不具备trièst提供的所有功能：对数据进行一次传递，处理完全动态的数据流，使用固定的内存空间，需要一个可解释的参数，并在每个时间瞬间返回一个估计值。此外，我们的实验结果表明，我们在仅有插入的数据流上的表现超过了[18, 32]的算法。

Lim and Kang [27] present an algorithm for insertion-only streams that is based on independent edge sampling with a fixed probability. Since the memory is not fully utilized during most of the stream, the variance of the estimate is large. Our approach handles fully-dynamic streams and makes better use of the available memory space at each time instant, resulting in a better estimation, as shown by our analytical and experimental results.

Lim和Kang[27]提出了一种用于只插入流的算法，该算法基于固定概率的独立边缘采样。由于在流的大部分时间里内存没有被完全利用，所以估计的方差很大。我们的方法处理完全动态的流，并在每个时间瞬间更好地利用可用的内存空间，导致更好的估计，正如我们的分析和实验结果所示。

Vitter [37] presents a detailed analysis of the reservoir sampling scheme and discusses methods to speed up the algorithm by reducing the number of calls to the random number generator. Random Pairing [14] is an extension of reservoir sampling to handle fully-dynamic streams with insertions and deletions. Cohen et al. [8] generalize and extend the Random Pairing approach to the case where the elements on the stream are key-value pairs, where the value may be negative (and less than −1). In our settings, where the value is not less than −1 (for an edge deletion), these generalizations do not apply and the algorithm presented by Cohen et al. [8] reduces essentially to Random Pairing.

Vitter[37]对水库采样方案进行了详细分析，并讨论了通过减少对随机数发生器的调用次数来加快算法的方法。随机配对[14]是水库抽样的一个扩展，以处理具有插入和删除的完全动态流。Cohen等人[8]将随机配对的方法概括并扩展到流中的元素是键值对的情况，其中的值可能是负的（并且小于-1）。在我们的设置中，值不小于-1（对于边缘删除），这些概括并不适用，Cohen等人[8]提出的算法基本上简化为随机配对。

**ALGORITHMS**

We present trièst, a suite of three novel algorithms for approximate global and local triangle counting from edge streams. The first two work on insertion-only streams, while the third can handle fully-dynamic streams where edge deletions are allowed.

我们提出了trièst，这是一套由三种新的算法组成的，用于对边缘流进行近似的全局和局部三角形计数。前两种算法只适用于插入流，而第三种算法可以处理允许删除边缘的完全动态流。

**Parameters.** Our algorithms keep an edge sample of up to edges from the stream, where is a positive integer parameter. For ease of presentation, we realistically assume . In Sect. 1 we motivated the design choice of only requiring as a parameter and remarked on its advantages over using a fixed sampling probability . Our algorithms are designed to use the available space as much as possible.

我们的算法从数据流中最多保留了条边的样本，其中是一个正整数参数。为了便于表述，我们现实地假设。在第1节中，我们阐述了只要求作为参数的设计选择，并评论了它比使用固定采样概率的优势。

**Counters.** trièst algorithms keep counters to compute the estimations of the global and local number of triangles. They always keep one global counter for the estimation of the global number of triangles. Only the global counter is needed to estimate the total triangle count. To estimate the local triangle counts, the algorithms keep a set of local counters for a subset of the nodes. The local counters are created on the fly as needed, and always destroyed as soon as they have a value of 0. Hence our algorithms use space (with one exception, see Sect. 4.2).

计数器。trièst算法保留计数器以计算全局和局部三角形数量的估计。它们总是保留一个全局计数器，用于估计全局的三角形数量。只需要全局计数器来估计总的三角形数量。为了估计局部三角形数量，算法为中的节点u的一个子集保留了一组局部计数器。这些局部计数器是在需要的时候创建的，一旦它们的值为0，就立即销毁。因此，我们的算法使用空间（有一个例外，见第4.2节）。

**Notation.** For any , let be the subgraph of containing all and only the edges in the current sample. We denote with the neighborhood of in : and with the shared neighborhood of and in .

符号。对于任何，让是的子图，包含所有且仅有当前样本 中的边。: ，用表示中和的共享邻居。

**A first algorithm – trièst-base**

We first present trièst-base, which works on insertiononly streams and uses standard reservoir sampling [37] to maintain the edge sample :

我们首先介绍trièst-base，它适用于只插入的数据流，并使用标准的蓄水池采样[37]来维持边缘样本。

• If , then the edge on the stream at time is deterministically inserted in.

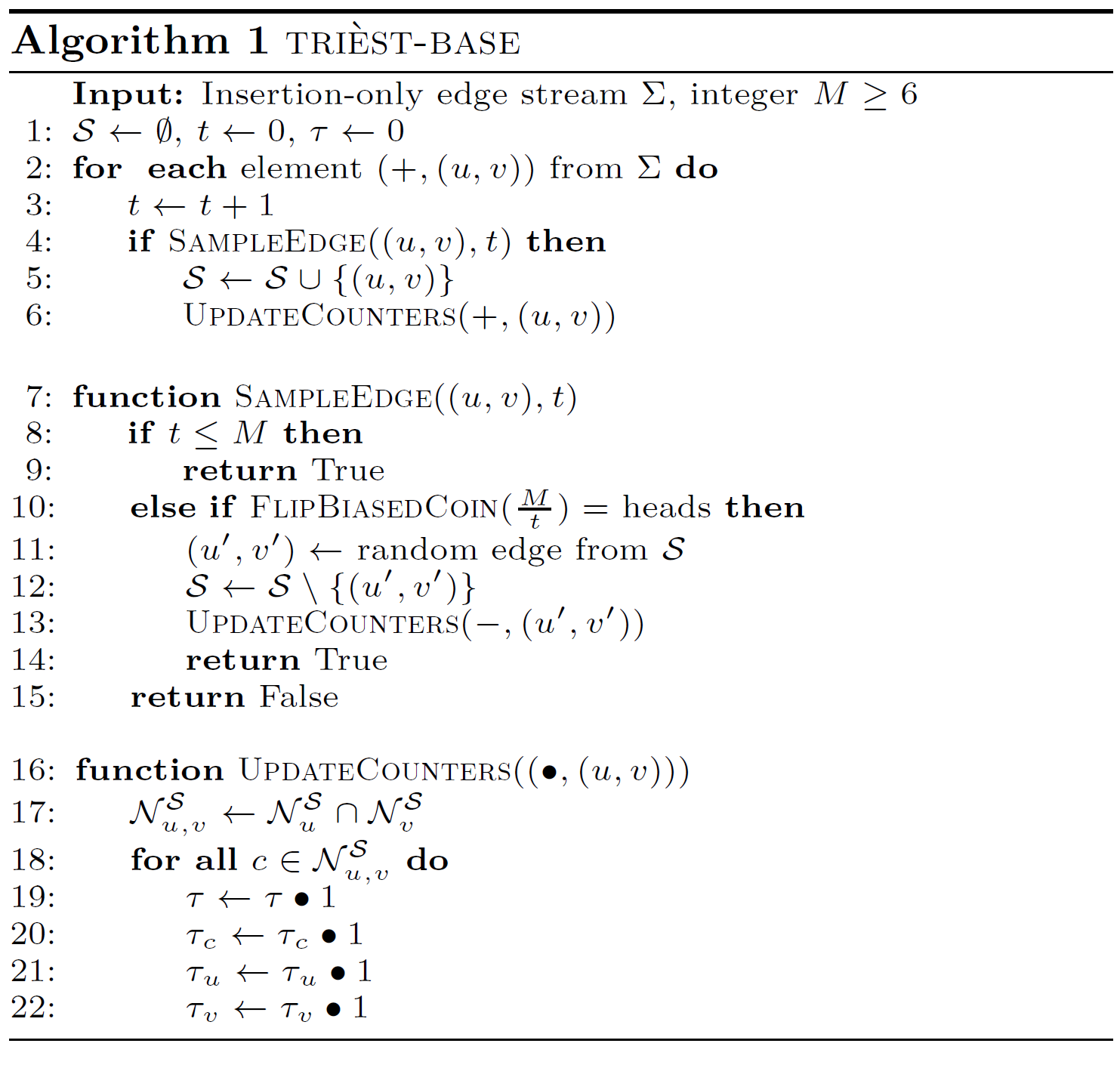
- 如果，那么在时间的流上的边被确定地插入中。

• If , trièst-base flips a biased coin with heads probability . If the outcome is heads, it chooses an edge uniformly at random, removes from , and inserts in . Otherwise, is not modified.

- 如果，trièst-base以的概率掷出一枚有偏向的硬币。如果结果是正面，它就随机选择一条边，从中删除，并在中插入，否则，不被修改。

After each insertion (resp. removal) of an edge from , trièst-base calls the procedure UpdateCounters that increments (resp. decrements) , and by, and by one, for each .

在每次从中插入（response）一条边后，trièst-base都会调用程序UpdateCounters，通过增加（response） , and ，并为每个增加1个。



**Estimation.** For any , let . Let (resp. ) be the value of the counter at the end of time step (i.e., after the edge on the stream at time has been processed by trièst-base) (resp. the value of the counter at the end of time step if there is such a counter, 0 otherwise). When queried at the end of time t, trièstbase returns (resp. ) as the estimation for the global (resp. local for) triangle count.

对于任何，让 `。让 (resp. )为时间步骤结束时（即时间t的流上的边被trièst-base处理后）计数器的值（如果有这样一个计数器，则为时间步骤t结束时计数器的值，否则为0）。当在时间t结束时被查询时，trièstbase返回 (resp. )作为全局（resp. 本地 for ）三角形计数的估计。

**Analysis.**

Theorem 1. We have

The trièst-base estimations are not only unbiased in all cases, but actually exact for, i.e., for , they are the true global/local number of triangles in .

trièst-base的估计不仅在所有情况下都是无偏的，而且对于来说实际上是精确的，也就是说，对于来说，它们是中真正的全局/局部三角形数量。

We now analyze the variance of the estimation returned by trièst-base for (the variance is 0 for .) Let . For any