Even the Easiest(?) Graph Coloring Problem Is Not Easy in Streaming!

Anup Bhattacharya
Indian Statistical Institute, Kolkata, India
bhattacharya.anup@gmail.com

Arijit Bishnu
Indian Statistical Institute, Kolkata, India
arijit@isical.ac.in

Gopinath Mishra
Indian Statistical Institute, Kolkata, India
gopianjan117@gmail.com

Anannya Upasana
Indian Statistical Institute, Kolkata, India
anannya.upasana23@gmail.com

Abstract

We study a graph coloring problem that is otherwise easy in the RAM model but becomes quite non-trivial in the one-pass streaming model. In contrast to previous graph coloring problems in streaming that try to find an assignment of colors to vertices, our main work is on estimating the number of conflicting or monochromatic edges given a coloring function that is streaming along with the graph; we call the problem Conflict-Est. The coloring function on a vertex can be read or accessed only when the vertex is revealed in the stream. If we need the color on a vertex that has streamed past, then that color, along with its vertex, has to be stored explicitly. We provide algorithms for a graph that is streaming in different variants of the vertex arrival in one-pass streaming model, viz. the Vertex Arrival (VA), Vertex Arrival With Degree Oracle (VAdeg), Vertex Arrival in Random Order (VArand) models, with special focus on the random order model. We also provide matching lower bounds for most of the cases. The mainstay of our work is in showing that the properties of a random order stream can be exploited to design efficient streaming algorithms for estimating the number of monochromatic edges. We have also obtained a lower bound, though not matching the upper bound, for the random order model. Among all the three models vis-a-vis this problem, we can show a clear separation of power in favor of the VArand model.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Probabilistic algorithms

Keywords and phrases Streaming, random ordering, graph coloring, estimation, lower bounds

Digital Object Identifier 10.4230/LIPIcs.ITCS.2021.15

Related Version A full version of the paper is available at `https://arxiv.org/pdf/2010.13143.pdf`.

Funding *Anup Bhattacharya*: Funded by NPDF fellowship at ISI, Kolkata.

1 Introduction

The *chromatic number* $\chi(G)$ of an n-vertex graph $G = (V, E)$ is the minimum number of colors needed to color the vertices of V so that no two adjacent vertices get the same color. The *chromatic number* problem is NP-hard and even hard to approximate within a factor of $n^{1-\varepsilon}$ for any constant $\varepsilon > 0$ [14, 28, 20]. For any connected undirected graph G with maximum degree Δ, $\chi(G)$ is at most $\Delta + 1$ [27]. This existential coloring scheme can be made constructive across different models of computation. A seminal result of recent vintage is that the $\Delta + 1$ coloring can be done in the streaming model [3]. Of late, there has been interest

in graph coloring problems in the sub-linear regime across a variety of models [1, 3, 4, 8, 6]. Keeping with the trend of coloring problems, these works look at assigning colors to vertices. Since the size of the output will be as large as the number of vertices, reseachers study the semi-streaming model [22] for streaming graphs. In the semi-streaming model, $\widetilde{\mathcal{O}}(n)$[1] space is allowed.

In a marked departure from the above works that look at the classical coloring problem, the starting point of our work is (inarguably?) the simplest question one can ask in graph coloring – given a coloring function $f : V \to \{1, \ldots, C\}$ on the vertex set V of a graph $G = (V, E)$, is f a valid coloring, i.e., for any edge $e \in E$, do both the endpoints of e have different colors? This is the problem one encounters while proving that the problem of chromatic number belongs to the class NP [15]. Conflict-Est, the problem of estimating the number of monochromatic (or, conflicting) edges for a graph G given a coloring function f, remains a simple problem in the RAM model; it even remains simple in the one-pass streaming model if the coloring function f is marked on a *public board*, readable at all times. We show that the problem throws up interesting consequences if the coloring function f on a vertex is revealed only when the vertex is revealed in the stream. For a streaming graph, if the vertices are assigned colors arbitrarily or randomly on-the-fly while it is exposed, our results can also be used to estimate the number of conflicting edges. These problems also find their use in estimating the number of conflicts in a job schedule and verifying a given job schedule in a streaming setting. This can also be extended to problems in various domains like frequency assignment in wireless mobile networks and register allocation [13]. As the problem, by its nature, admits an estimate or a yes-no answer, we can try for space efficient algorithms in the conventional graph streaming models like Vertex Arrival [11]. We also note in passing that many of the trend setting problems in streaming, like frequency moments, distinct elements, majority, etc. have been simple problems in the ubiquitous RAM model as the coloring problem we solve here.

2 Preliminaries

2.1 Notations and the streaming models

Notations. We denote the set $\{1, \ldots, n\}$ by $[n]$. $G(V(G), E(G))$ denotes a graph where $V(G)$ and $E(G)$ denote the set of vertices and edges of G, respectively; $|V| = n$ and $|E| = m$. We will write only V and E for vertices and edges when the graph is clear from the context. We denote $E_M \subseteq E$ as the set of monochromatic edges. The set of neighbors of a vertex $u \in V(G)$ is denoted by $N_G(u)$ and the degree of a vertex $u \in V(G)$ is denoted by $d_G(u)$. Let $N_G(u) = N^-_G(u) \uplus N^+_G(u)$ where $N^-_G(u)$ and $N^+_G(u)$ denote the set of neighbors of u that have been exposed already and are yet to be exposed, respectively in the stream. Also, $d_G(u) = d^-_G(u) + d^+_G(u)$ where $d^-_G(u) = \left|N^-_G(u)\right|$ and $d^+_G(u) = \left|N^+_G(u)\right|$. For a monochromatic edge $(u, v) \in E_M$, we refer to u and v as monochromatic neighbors of each other. We define $d_M(u)$ to be the number of monochromatic neighbors of u and hence, the monochromatic degree of u.

Let $\mathbb{E}[X]$ denote the expectation of the random variable X. For an event $\mathcal{E}$, $\overline{\mathcal{E}}$ denotes the complement of $\mathcal{E}$. $\mathbb{P}(\mathcal{E})$ denotes the probability of an event $\mathcal{E}$. The statement "event $\mathcal{E}$ occurs with high probability" is equivalent to $\mathbb{P}(\mathcal{E}) \geq 1 - \frac{1}{n^c}$, where c is an absolute constant. The statement "a is a $1 \pm \varepsilon$ multiplicative approximation of b" means $|b - a| \leq \varepsilon \cdot b$. For $x \in \mathbb{R}$, $\exp(x)$ denotes the standard exponential function, that is, e^x. By polylogarithmic, we mean $\mathcal{O}\left((\log n/\varepsilon)^{\mathcal{O}(1)}\right)$. The notation $\widetilde{\mathcal{O}}(\cdot)$ hides a polylogarithmic term in $\mathcal{O}(\cdot)$.

[1] $\widetilde{\mathcal{O}}(\cdot)$ hides a polylogarithmic factor.

Streaming models for graphs. As alluded to earlier, the crux of the problem depends on the way the coloring function f is revealed in the stream. The details follow.

(i) Vertex Arrival (VA): The vertices of V are exposed in an arbitrary order. After a vertex $v \in V$ is exposed, all the edges between v and pre-exposed neighbors of v, are revealed. This set of edges are revealed one by one in an arbitrary order. Along with the vertex v, only the color $f(v)$ is exposed, and not the colors of any pre-exposed vertices. So, we can check the monochromaticity of an edge (v, u) only if u and $f(u)$ are explicitly stored.

(ii) Vertex Arrival with Degree Oracle (VAdeg) [23, 7]: This model works same as the VA model in terms of exposure of the vertex v and the coloring on it; but we are allowed to know the degree $d_G(v)$ of the currently exposed vertex v from a degree oracle on G.

(iii) Vertex Arrival in Random Order (VArand) [25, 26]: This model works same as the VA model but the vertex sequence revealed is equally likely to be any one of the permutations of the vertices.

(iv) Edge Arrival (EA): The stream consists of edges of G in an arbitrary order. As the edge e is revealed, so are the colors on its endpoints. Thus the conflicts can be easily checked.

(v) Adjacency List (AL): The vertices of V are exposed in an arbitrary order. When a vertex v is exposed, all the edges that are incident to v, are revealed one by one in an arbitrary order. Note that in this model each edge is exposed twice, once for each exposure of an incident vertex. As in the VA model, here also only v's color $f(v)$ is exposed.

As the conflicts can be checked easily in the EA model in $O(1)$ space, a logarithmic counter is enough to count the number of monochromatic edges. The AL model works almost the same as the VAdeg model. So, we focus on the three models – VA, VAdeg and VArand in this work and show that they have a clear separation in their power vis-a-vis the problem we solve. A crucial takeaway from our work is that the random order assumption on exposure of vertices has huge improvements in space complexity.

2.2 Problem definitions, results and the ideas

Problem definition. Let the vertices of G be colored with a function $f : V(G) \to [C]$, for $C \in \mathbb{N}$. An edge $(u, v) \in E(G)$ is said to be *monochromatic* or *conflicting* with respect to f if $f(u) = f(v)$. A coloring function f is called *valid* if no edge in $E(G)$ is monochromatic with respect to f. For a given parameter $\varepsilon \in (0, 1)$, f is said to be ε-far from being *valid* if at least $\varepsilon \cdot |E(G)|$ edges are monochromatic with respect to f. We study the following problems.

▶ **Problem 2.1** (Conflict Estimation aka Conflict-Est). *A graph $G = (V, E)$ and a coloring function $f : V(G) \to [C]$ are streaming inputs. Given an input parameter $\varepsilon > 0$, the objective is to estimate the number of monochromatic edges in G within a $(1 \pm \varepsilon)$-factor.*

▶ **Problem 2.2** (Conflict Separation aka Conflict-Sep). *A graph $G = (V, E)$ and a coloring function $f : V(G) \to [C]$ are streaming inputs. Given an input parameter $\varepsilon > 0$, the objective is to distinguish if the coloring function f is valid or is ε-far from being valid.*

▶ Remark 2.3. Problem 2.1 is our main focus, but we will mention a result on Problem 2.2 in Section 5. Notice that the problem Conflict-Est is at least as hard as Conflict-Sep.

The results and the ideas involved. All our upper and lower bounds on space are for one-pass streaming algorithms. Table 1 states our results for the Conflict-Est problem, the main problem we solve in this paper, across different variants of the VA model. The main thrust of our work is on estimating monochromatic edges under random order stream. For random order stream, we present both upper and lower bounds in Sections 3 and 4. There is a gap between the upper and lower bounds in the VArand model, though we have a strong hunch that our upper bound is tight. Apart from the above, using a structural result on graphs, we show in Section 5 that the Conflict-Sep problem admits an easy algorithm in the VArand model. To give a complete picture across different variants of VA models, we show matching upper and lower bounds for constant $\varepsilon > 0$ in the VA and VAdeg models in [9][2].

Table 1 This table shows our results on Conflict-Est on a graph $G(V, E)$ across different Vertex Arrival models. Here, $T > 0$ denotes the promised lower bound on the number of monochromatic edges. This paper discusses the results mentioned in the middle column corresponding to VArand. The other results are discussed in the full version of the paper [9].

Model	VA	VArand	VAdeg
Upper Bound	$\widetilde{\mathcal{O}}\left(\min\{\lvert V\rvert, \frac{\lvert V\rvert^2}{T}\}\right)$ (Thm. 3.1 in [9])	$\widetilde{\mathcal{O}}\left(\frac{\lvert V\rvert}{\sqrt{T}}\right)$ (Sec. 3, Thm. 3.1)	$\widetilde{\mathcal{O}}\left(\min\{\lvert V\rvert, \frac{\lvert E\rvert}{T}\}\right)$ (Thm. 3.2 in [9])
Lower Bound	$\Omega\left(\min\{\lvert V\rvert, \frac{\lvert V\rvert^2}{T}\}\right)$ (Thm. E.1 in [9])	$\Omega\left(\frac{\lvert V\rvert}{T^2}\right)$ (Sec. 4, Thm. 4.1)	$\Omega\left(\min\{\lvert V\rvert, \frac{\lvert E\rvert}{T}\}\right)$ (Thm. E.2 in [9])

The promise T on the number of monochromatic edges is a very standard assumption for estimating substructures in the world of graph streaming algorithm [17, 19, 18, 23, 5]. [3]

We now briefly mention the salient ideas involved. For the simpler variant of Conflict-Est in VA model, we first check if $|V| \geq T$. If yes, we store all the vertices and their colors in the stream to determine the exact value of the number of monochromatic edges. Otherwise, we sample each pair of vertices $\{u, v\}$ in $\binom{V}{2}$ [4], with probability $\widetilde{\mathcal{O}}(1/T)$ independently [5] before the stream starts. When the stream comes, we compute the number of monochromatic edges from this sample. The details are in Section 3 of [9]. Though the algorithm looks extremely simple, it matches the lower bound result for Conflict-Est in VA model, presented in Appendix E of [9]. The VAdeg model with its added power of a degree oracle, allows us to know $d_G(u)$ for a vertex u and as edges to pre-exposed vertices are revealed, we also know $d^-_G(u)$ and $d^+_G(u)$. This allows us to use sampling to store vertices and to use a technique which we call *sampling into the future* where indices of random neighbors, out of $d^+_G(u)$ neighbors, are selected for future checking. The upper bound result for Conflict-Est in VAdeg model, presented in Section 3 of [9], is tight as we also prove a matching lower bound in Appendix E of [9].

[2] The reference [9] is the full version of this submission.
[3] Here we have cited a few. However, there are huge amount of relevant literature.
[4] $\binom{V}{2}$ denotes the set of all size 2 subsets of $V(G)$.
[5] Note that we might sample some pairs that are not forming edges in the graph.

The algorithm for Conflict-Est in VArand model is the mainstay of our work and is presented in Section 3. We redefine the degree in terms of the number of monochromatic neighbors a vertex has in the randomly sampled set. Here, we estimate the high monochromatic degree and low monochromatic degree vertices separately by sampling a random subset of vertices. While the monochromatic degree for the high degree vertices can be extrapolated from the sample, handling low monochromatic degree vertices individually in the same way does not work. To get around, we group such vertices having similar monochromatic degress and treat them as an entity. We also provide a lower bound for the VArand model, in Section 4, using a reduction from *multi-party set disjointness*; though there is a gap in terms of the exponent in T.

The highlights of our work are as follows:

- We show that possibly the easiest graph coloring problem is worth studying over streams.
- For researchers working in streaming, the *gold standard* is the EA model as most problems are non-trivial in this model. We point out a problem that is harder to solve in the VA model as compared to the EA model.
- We show that the three VA related models have a clear separation in their space complexities vis-a-vis the problem we solve. We could exploit the random order of the arrival of the vertices to get substantial improvements in space complexity.
- We could obtain lower bounds for all the three models and the lower bounds are matching for the VA and VAdeg models.

2.3 Prior works on graph coloring in semi-streaming model.

Bera and Ghosh [8] commenced the study of vertex coloring in the semi-streaming model. They devise a randomized one pass streaming algorithm that finds a $(1+\varepsilon)\Delta$ vertex coloring in $\widetilde{\mathcal{O}}(n)$ space. Assadi et al. [3] find a proper vertex coloring using $\Delta+1$ colors via various classes of sublinear algorithms. Their state of the art contributions can be attributed to a key result called the *palette-sparsification theorem* which states that for an n-vertex graph with maximum degree Δ, if $\mathcal{O}(\log n)$ colors are sampled independently and uniformly at random for each vertex from a list of $\Delta+1$ colors, then with a high probability a proper $\Delta+1$ coloring exists for the graph. They design a randomized one-pass dynamic streaming algorithm for the $\Delta+1$ coloring using $\widetilde{\mathcal{O}}(n)$ space. The algorithm takes post-processing $\widetilde{\mathcal{O}}(n\sqrt{\Delta})$ time and assumes a prior knowledge of Δ. Alon and Assadi [2] improve the palette sparsification result of [3]. They consider situations where the number of colors available is both more than and less than $\Delta+1$ colors. They show that sampling $\mathcal{O}_\varepsilon(\sqrt{\log n})$ colors per vertex is sufficient and necessary for a $(1+\varepsilon)\Delta$ coloring. Bera et al. [6] give a new graph coloring algorithm in the semi-streaming model where the number of colors used is parameterized by the degeneracy κ. The key idea is a *low degeneracy partition*, also employed in [8]. The numbers of colors used to properly color the graph is $\kappa+o(\kappa)$ and post-processing time of the algorithm is improved to $\widetilde{\mathcal{O}}(n)$, without any prior knowledge about κ. Behnezhad et al. [4] were the first to give one-pass W-streaming algorithms (streaming algorithms where outputs are produced in a streaming fashion as opposed to outputs given finally at the end) for edge coloring both when the edges arrive in a random order or in an adversarial fashion.

3 Conflict-Est in VArand model

In this Section, we show that the power of randomness can be used to design a better solution for the Conflict-Est problem in the VArand model. The Conflict-Est problem is the main highlight of our work. We feel that the crucial use of randomness in the input that is used to estimate a substructure (here, monochromatic edges) in a graph, will be of independent interest.

In this variant, we are given an $\varepsilon \in (0,1)$ and a promised lower bound T on $|E_M|$, the number of monochromatic edges in G, as input and our objective is to determine a $(1 \pm \varepsilon)$-approximation to $|E_M|$.

▶ **Theorem 3.1.** *Given any graph $G = (V, E)$ and a coloring function $f : V(G) \to [C]$ as input in the stream, the* Conflict-Est *problem in the* VArand *model can be solved with high probability in $\widetilde{\mathcal{O}}\left(\frac{|V|}{\sqrt{T}}\right)$ space, where T is a lower bound on the number of monochromatic edges in the graph.*

The proof idea

A random sample comes for free – pick the first few vertices

Let $v_1, \ldots, v_n$ be the random ordering in which the vertices of V are revealed. Let R be a random subset of $\Gamma = \widetilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$ many vertices of G sampled without replacement [6]. As we are dealing with a random order stream, consider the first Γ vertices in the stream; they can be treated as R, the random sample. We start by storing all the vertices in R as well as their colors. Observe that if the monochromatic degree of any vertex v_i is *large* (say roughly more than $\sqrt{T}$), then it can be well approximated by looking at the number of monochromatic neighbors that v_i has in R. As a vertex v_i streams past, there is no way we can figure out its monochromatic degree, unless we store its monochromatic neighbors that appear before it in the stream; if we could, we were done. Our only savior is the stored random subset R.

Classifying the vertices of the random sample R based on its monochromatic degree

Our algorithm proceeds by figuring out the influence of the color of v_i on the monochromatic degrees of vertices in R. To estimate this, let κ_{v_i} denote the number of monochromatic neighbors that v_i has in R. We set a threshold $\tau = \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}$, where $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$. The significance of t will be clear from the discussion below. Any vertex v_i will be classified as a high-m_R or low-m_R degree vertex depending on its monochromatic degree within R, i.e., if $\kappa_{v_i} \geq \tau$, then v_i is a high-m_R vertex, else it is a low-m_R vertex, respectively. (We use the subscripts m_R to stress the fact that the monochromatic degrees are induced by the set R.) Let H and L be the partition of V into the set of high-m_R and low-m_R degree vertices in G. Let H_R and L_R denote the set of high-m_R and low-m_R degree vertices in R. Notice that, because of the definition of high-m_R and low-m_R degree vertices, not only the sets H_R, L_R are subsets of R, but they are determined by the vertices of R only.

Let m_h and m_ℓ denote the sum of the monochromatic degrees of all the high-m_R degree vertices and low-m_R degree vertices in G, respectively. So, $m_h = \sum_{v \in H} d_M(v)$ and $m_\ell = \sum_{v \in L} d_M(v)$. Note that $\widehat{m} = |E_M| = \frac{1}{2}\sum_{v \in V} d_M(v) = \frac{1}{2}(m_h + m_\ell)$. We will describe how to approximate m_h and m_ℓ separately. The formal algorithm is described in Algorithm 1

[6] $\widetilde{\Theta}(\cdot)$ hides a polynomial factor of $\log n$ and $\frac{1}{\varepsilon}$ in the upper bound.

as RANDOM-ORDER-EST(ε, T) (in Appendix D) that basically executes steps to approximate m_h and m_ℓ in parallel.

To approximate m_h, the random sample R comes to rescue

We can find $\widehat{m_h}$, that is, a $\left(1 \pm \frac{\varepsilon}{10}\right)$ approximation of m_h as described below. For each vertex $v_i \in R$ and each monochromatic edge (u, v_i), $u \in R$, we see in the stream, we increase the value of κ_u for u and κ_{v_i} for v_i. After all the vertices in R are revealed, we can determine H_R by checking whether $\kappa_{v_i} \geq \tau$ for each $v_i \in R$. For each vertex $v_i \in H_R$, we set its approximate monochromatic degree $\widehat{d_{v_i}}$ to be $\frac{n}{|R|}\kappa_{v_i}$. We initialize the estimated sum of the monochromatic degree of high degree vertices as $\widehat{m_h} = \sum_{v_i \in H_R} \widehat{d_{v_i}}$. For each vertex $v_i \notin R$ in the stream, we can determine κ_{v_i}, as we have stored all the vertices in R along with their colors, and hence we can also determine whether v_i is a high-m$_R$ degree vertex in G. If $v_i \notin R$ is a high-m$_R$ degree vertex, we determine $\widehat{d_{v_i}} = \frac{n}{|R|}\kappa_{v_i}$ and update $\widehat{m_h}$ by $\widehat{m_h} + \widehat{d_{v_i}}$. Observe that, at the end, $\widehat{m_h}$ is $\sum_{v_i \in H} \widehat{d_{v_i}}$. Recall that H is the set of all high-m$_R$ degree vertices in G. For each $v_i \in H$, we will show, as in Claim 3.3, that $\widehat{d_{v_i}}$ is a $\left(1 \pm \frac{\varepsilon}{10}\right)$-approximation to $d_M(v_i)$ with high probability. This implies that

$$\left(1 - \frac{\varepsilon}{10}\right) m_h \leq \widehat{m_h} \leq \left(1 + \frac{\varepsilon}{10}\right) m_h \tag{1}$$

To approximate m_ℓ, group the vertices in L based on similar monochromatic degree

Recall that $m_\ell = \sum_{v_i \in L} d_M(v_i)$. Unlike the high-m$_R$ degree vertices, it is not possible to approximate the monochromatic degree of $v_i \in L$ from κ_{v_i}. To cope up with this problem, we partition the vertices of L into t buckets $B_1, \ldots, B_t$ such that all the vertices present in a bucket have *similar* monochromatic degrees, where $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$. The bucket B_j is defined as follows: $B_j = \{v_i \in L : \left(1 + \frac{\varepsilon}{10}\right)^{j-1} \leq d_M(v_i) < \left(1 + \frac{\varepsilon}{10}\right)^{j}\}$.

Note that our algorithm will not find the buckets explicitly. It will be used for the analysis only. Observe that $\sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j \in [t]} |B_j| \left(1 + \frac{\varepsilon}{10}\right)^{j}$. We can surely approximate m_ℓ by approximating $|B_j|$s suitably. We estimate $|B_j|$s as follows. After the stream of the vertices in R has gone past, we have the set of low-m$_R$ degree vertices L_R in R and $\widehat{d_{v_i}} = \kappa_{v_i}$ for each $v_i \in L_R$. For each $v_i \notin R$ in the stream, we determine the monochromatic neighbors of v_i in L_R. It is possible as we have stored all the vertices in R and their colors. For each monochromatic neighbor $v_{i'} \in L_R$ of v_i, we increase the value of $\widehat{d}_{v_{i'}}$ of $v_{i'}$. Observe that, at the end of the stream, $\widehat{d_{v_{i'}}} = d_M(v_{i'})$ for each $v_{i'} \in L_R$, i.e., we can accurately estimate the monochromatic degree of each $v_{i'} \in L_R$. So, we can determine the bucket where each vertex in L_R belongs. Let A_j $(= L_R \cap B_j)$ be the bucket B_j projected onto L_R in the random sample; note that as $B_j \subseteq L$ and $L_R = L \cap R$, $A_j = R \cap B_j$ also. We determine $\widehat{m_\ell} = \frac{n}{|R|} \sum_{j \in [t]} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^{j}$. We can show that $\frac{n}{|R|} |A_j|$ is a $\left(1 + \frac{\varepsilon}{10}\right)$-approximation of $|B_j|$, with high probability, if $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$. Also, we can show that, if $|B_j| < \frac{\sqrt{\varepsilon T}}{10t}$, then $|A_j| \leq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$ with high probability. Now using the fact that we consider bucketing of only low-m$_R$ degree vertices (L_R), we can show that

$$\left(1 - \frac{\varepsilon}{10}\right) \left(m_\ell - \frac{\varepsilon T}{63t}\right) \leq \widehat{m_\ell} \leq \left(1 + \frac{\varepsilon}{10}\right)^2 \left(m_\ell + \frac{\varepsilon T}{56t}\right). \tag{2}$$

Note that $\varepsilon \in (0, 1)$ and $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$. Assuming $T \geq 63t^2$, Equations 1 and 2 imply that $\widehat{m} = \frac{1}{2}(\widehat{m_h} + \widehat{m_\ell})$ is a $(1 \pm \varepsilon)$-approximation to $|E_M|$. If $T < 63t^2$, then note that

$n = \widetilde{\mathcal{O}}\left(\frac{n}{\sqrt{T}}\right)$. So, in that case, we store all the vertices along with their colors and compute the exact value of $|E_M|$.

Proof of correctness

The correctness of the algorithm follows trivially if $T < 63t^2$. So, let us assume that $T \geq 63t^2$. In the VARAND model, we consider the first $\widetilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$ vertices as the random sample R without replacement. Using the Chernoff bound for sampling without replacement (See Lemma A.2 in Appendix A), we can have the following lemma (The proof is in Appendix B), which will be useful for the correctness proof of Algorithm 1 (RANDOM-ORDER-EST(ε, T)) in case of $T \geq 63t^2$.

▶ **Lemma 3.2.**

(i) *For each $j \in [t]$ with $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(\left||B_j \cap R| - \frac{|R||B_j|}{n}\right| \geq \frac{\varepsilon}{10}\frac{|R||B_j|}{n}\right) \leq \frac{1}{n^{10}}$.*

(ii) *For each $j \in [t]$ with $|B_j| < \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(|B_j \cap R| \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}$.*

(iii) *For each vertex v_i with $d_M(v_i) \geq \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(\left|\kappa_{v_i} - \frac{|R|d_M(v_i)}{n}\right| \geq \frac{\varepsilon}{10}\frac{|R|d_M(v_i)}{n}\right) \leq \frac{1}{n^{10}}$.*

(iv) *For each vertex v_i with $d_M(v_i) < \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(\kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}$.*

The correctness proof of the algorithm is divided into the following two claims.

▷ Claim 3.3. $\left(1 - \frac{\varepsilon}{10}\right) m_h \leq \widehat{m_h} \leq \left(1 + \frac{\varepsilon}{10}\right) m_h$ with probability at least $1 - \frac{1}{n^9}$.

▷ Claim 3.4. $\left(1 - \frac{\varepsilon}{10}\right)\left(m_\ell - \frac{\varepsilon T}{63t}\right) \leq \widehat{m_\ell} \leq \left(1 + \frac{\varepsilon}{10}\right)^2\left(m_\ell + \frac{\varepsilon T}{56t}\right)$ with probability at least $1 - \frac{1}{n^7}$.

Assuming the above two claims hold and taking $\varepsilon \in (0, 1)$, $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$ and $T \geq 63t^2$, observe that $\widehat{m} = \frac{1}{2}(\widehat{m_h} + \widehat{m_\ell})$ is a $(1 \pm \varepsilon)$ approximation of $|E_M| = m_h + m_\ell$ with high probability. Thus, it remains to prove Claims 3.3 and 3.4.

Proof of Claim 3.3. Note that $m_h = \sum_{v_i : \kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} d_M(v_i)$ and $\widehat{m_h} = \sum_{v_i : \kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} \widehat{d_{v_i}}$.

From Lemma 3.2 (iv) and (iii), $\kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}$ implies that $\widehat{d_{v_i}}$ is an $\left(1 \pm \frac{\varepsilon}{10}\right)$ approximation to $d_M(v_i)$ with probability at least $1 - \frac{2}{n^{10}}$. Hence, we have $\left(1 - \frac{\varepsilon}{10}\right) m_h \leq \widehat{m_h} \leq \left(1 + \frac{\varepsilon}{10}\right) m_h$ with probability at least $1 - \frac{1}{n^9}$. ◁

Proof of Claim 3.4. Note that $m_\ell = \sum_{v_i \in L} d_M(v_i) = \sum_{v_i : \kappa_{v_i} < \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} d_M(v_i)$ and $\widehat{m_\ell} = \frac{n}{|R|}\sum_{j \in [t]} |A_j|\left(1 + \frac{\varepsilon}{10}\right)^j$. Recall that the vertices in L are partitioned into t buckets as follows:
$B_j = \{v_i \in L : \left(1 + \frac{\varepsilon}{10}\right)^{j-1} \leq d_M(v_i) < \left(1 + \frac{\varepsilon}{10}\right)^j\}$, where $j \in [t]$. By Lemma 3.2 (iv), $\kappa_{v_i} < \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}$ implies that $d_M(v_i) \leq \frac{\sqrt{\varepsilon T}}{7t}$ with probability $1 - \frac{1}{n^{10}}$. So, we have the following observation.

▶ **Observation 3.5.** *Let $j \in [t]$ be such that $|A_j| \neq 0$ ($|B_j| \neq 0$). Then, with probability at least $1 - \frac{1}{n^{10}}$, the monochromatic degree of each vertex in A_j as well as B_j is at most $\frac{\sqrt{\varepsilon T}}{7t}$, that is, $\left(1 + \frac{\varepsilon}{10}\right)^j \leq \frac{\sqrt{\varepsilon T}}{7t}$.*

To upper and lower bound $\widehat{m_\ell}$ in terms of m_ℓ, we upper and lower bound m_ℓ in terms of $|B_j|$'s as follows; for the upper bound, we break the sum into two parts corresponding to large and small sized buckets:

$$\sum_{j\in[t]} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j\in[t]} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j}$$

$$\sum_{j\in[t]} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j\in[t]:|B_j|\geq\frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j} + \sum_{j\in[t]:|B_j|<\frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j}$$

By Observation 3.5, we bound m_ℓ in terms of $|B_j|$'s with probability $1-\frac{1}{n^9}$.

$$\sum_{j\in[t]} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j\in[t]:|B_j|\geq\frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j} + t\cdot\frac{\sqrt{\varepsilon T}}{9t}\frac{\sqrt{\varepsilon T}}{7t}$$

This implies the following Observation:

▶ **Observation 3.6.** $\sum_{j\in[t]} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j-1} \leq m_\ell < \sum_{j\in[t]:|B_j|\geq\frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j} + \frac{\varepsilon T}{63t}$ *holds with probability at least* $1-\frac{1}{n^9}$.

Now, we have all the ingredients to show that $\widehat{m_\ell}$ is a $(1\pm\varepsilon)$ approximation of m_ℓ. To get to $\widehat{m_\ell}$, we need to focus on low-m$_R$ vertices of R, i.e., A_j's. Breaking $\widehat{m_\ell} = \frac{n}{|R|}\sum_{j\in[t]} |A_j| \left(1+\frac{\varepsilon}{10}\right)^{j}$ depending on small and large values of $|A_j|$'s (recall $A_j = L_R \cap B_j = R\cap B_j$), we have

$$\widehat{m_\ell} = \frac{n}{|R|}\left[\sum_{j\in[t]:|A_j|\geq\frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} |A_j| \left(1+\frac{\varepsilon}{10}\right)^{j} + \sum_{j\in[t]:|A_j|<\frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} |A_j| \left(1+\frac{\varepsilon}{10}\right)^{j}\right] \tag{3}$$

Note that $A_j = B_j \cap R$. By Lemma 3.2 (ii), $|A_j| \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}$ implies $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$ with probability at least $1-\frac{1}{n^{10}}$. Also, applying Lemma 3.2 (i), $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$ implies $|A_j|$ is an $\left(1\pm\frac{\varepsilon}{10}\right)$-approximation to $\frac{|R||B_j|}{n}$ with probability at least $1-\frac{1}{n^{10}}$. So, we have the following observation.

▶ **Observation 3.7.** *Let $j\in[t]$ be such that $|A_j| \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}$. Then $|A_j|$ is an $\left(1\pm\frac{\varepsilon}{10}\right)$-approximation to $\frac{|R||B_j|}{n}$ with probability at least $1-\frac{2}{n^{10}}$, that is, $\frac{n}{|R|}|A_j|$ is an $\left(1\pm\frac{\varepsilon}{10}\right)$-approximation to $|B_j|$ with probability at least $1-\frac{2}{n^{10}}$*

By the above observation along with Equation 3, we have the following upper bound on $\widehat{m_\ell}$ with probability at least $1-\frac{1}{n^9}$.

$$\begin{aligned}\widehat{m_\ell} &\leq \sum_{j\in[t]:|A_j|\geq\frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} \left(1+\frac{\varepsilon}{10}\right)|B_j| \left(1+\frac{\varepsilon}{10}\right)^{j} + \sum_{j\in[t]:|A_j|<\frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} \frac{n}{|R|}|A_j| \left(1+\frac{\varepsilon}{10}\right)^{j} \\ &\leq \left(1+\frac{\varepsilon}{10}\right)^{2}\left[\sum_{j\in[t]:|A_j|\geq\frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^{j-1} + \sum_{j\in[t]:|A_j|<\frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} \frac{\sqrt{\varepsilon T}}{8t}\left(1+\frac{\varepsilon}{10}\right)^{j-2}\right]\end{aligned}$$

Now by Observations 3.6 and 3.5, we have the following with probability at least $1-\frac{1}{n^8}$.

$$\begin{aligned}\widehat{m_\ell} &\leq \left(1+\frac{\varepsilon}{10}\right)^{2}\left(m_\ell + t\cdot\frac{\sqrt{\varepsilon T}}{8t}\frac{\sqrt{\varepsilon T}}{7t}\right) \\ &= \left(1+\frac{\varepsilon}{10}\right)^{2}\left(m_\ell + \frac{\varepsilon T}{56t}\right)\end{aligned}$$

Now, we will lower bound $\widehat{m_\ell}$. From Equation 3, we have

$$\widehat{m_\ell} \geq \frac{n}{|R|} \sum_{j\in[t]:|A_j|\geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} |A_j| \left(1+\frac{\varepsilon}{10}\right)^j$$

By Observation 3.7, $|A_j| \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}$ implies $\frac{n}{|R|}|A_j|$ is an $\left(1\pm\frac{\varepsilon}{10}\right)$-approximation to $|B_j|$ with probability at least $1-\frac{2}{n^{10}}$. So, the following lower bound on $\widehat{m_\ell}$ holds with probability at least $1-\frac{1}{n^9}$.

$$\widehat{m_\ell} \geq \left(1-\frac{\varepsilon}{10}\right) \sum_{j\in[t]:|A_j|\geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^j$$

By Lemma 3.2 (i), if $|B_j| \geq \frac{\sqrt{\varepsilon T}}{9t}$, then $|A_j| \geq \frac{\sqrt{\varepsilon T}}{8t}$ with probability at least $1-\frac{1}{n^{10}}$. Hence, we have the following lower bound on $\widehat{m_\ell}$ with probability at least $1-\frac{1}{n^8}$.

$$\widehat{m_\ell} \geq \left(1-\frac{\varepsilon}{10}\right) \sum_{j\in[t]:|B_j|\geq \frac{\sqrt{\varepsilon T}}{9t}} |B_j| \left(1+\frac{\varepsilon}{10}\right)^j$$

Now by Observation 3.6, we have the following with probability at least $1-\frac{1}{n^7}$.

$$\widehat{m_\ell} \geq \left(1-\frac{\varepsilon}{10}\right)\left(m_\ell - \frac{\varepsilon T}{63t}\right). \qquad \triangleleft$$

4 Lower bound for Conflict-Est in VArand model

In this Section, we show a lower bound of $\Omega\left(\frac{n}{T^2}\right)$ for Conflict-Est in Vertex Arrival in Random Order via a reduction from a variation of Multiparty Set Disjointness problem called Disjointness$_R(t,n,p)$, played among p players: Consider a matrix of order $t\times n$ having t (rows) vectors $M_1,\ldots,M_t \in \{0,1\}^n$ such that each entry of matrix M is given to one of the p players chosen uniformly at random. The objective is to determine whether there exists a column where all the entries are 1s. If $t\geq 2$ and $p=\Omega(t^2)$, Chakrabarti et al. showed that any randomized protocol requires $\Omega\left(\frac{n}{t}\right)$ bits of communication [10]. They showed that the lower bound holds under a promise called the unique intersection promise which states that there exists at most a single column where all the entries are 1s and every other column of the matrix has Hamming weight either 0 or 1. Moreover, the lower bound holds even if all the p players know the random partition of the entries of matrix M.

▶ **Theorem 4.1.** *Let $n,T\in\mathbb{N}$ be such that $4\leq T\leq\binom{n}{2}$. Any constant pass streaming algorithm that takes the vertices and edges of a graph $G(V,E)$ (with $|V|=\Theta(n)$ and $|E|=\Theta(m)$) and a coloring function $f:V\to[C]$ in the* VArand *model, and determines whether the monochromatic edges in G is 0 or $\Omega(T)$ with probability $2/3$, requires $\Omega\left(\frac{n}{T^2}\right)$ bits of space.*

Proof. Without loss of generality, assume that $\sqrt{T}\in\mathbb{N}$. Consider the Disjointness$_R\left(\sqrt{T},\frac{n}{\sqrt{T}},p\right)$ problem with Unique Intersection promise when all of the p players know the random partition of the entries of the relevant matrix M. Note that M is of order $[\sqrt{T}]\times\left[\frac{n}{\sqrt{T}}\right]$ and $p=AT$ for some suitable constant $A\in\mathbb{N}$. Also, consider a graph G, with $V(G)=\{v_{ij}: i\in\left[\sqrt{T}\right], j\in\left[\frac{n}{\sqrt{T}}\right]\}$, having $\frac{n}{\sqrt{T}}$ many vertex disjoint cliques such that $\{v_{1j},\ldots,v_{\sqrt{T}j}\}$ forms a clique for each $j\in[n]$, i.e., a column of M forms a clique. Also, notice

that each clique has $\Theta(T)$ edges. Let us assume that there is an r-pass streaming algorithm $\mathcal{S}$, with space complexity s bits, that solves Conflict-Est for the above graph G in the VArand model. Now, we give a protocol $\mathcal{A}$ for $\textsc{Disjointness}_R\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$ with communication cost $O(rsp)$. Using the fact that the lower bound of $\textsc{Disjointness}\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$ is $\Omega\left(\frac{n/\sqrt{T}}{\sqrt{T}}\right)$ along with the fact that $p = AT$ and r is a constant, we get $s = \Omega\left(\frac{n}{T^2}\right)$.

Protocol $\mathcal{A}$ for $\textsc{Disjointness}_R\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$

Let $P_1, \ldots, P_p$ denote the set of p players. For $k \in [p]$, $V_k = \{v_{ij} : M_{ij} \text{ is with } P_k\}$, where M_{ij} denotes the element present in the i-th row and j-th column of matrix M. Note that there is a one-to-one correspondence between the entries of M and the vertices in $V(G)$. Furthermore, there is a one-to-one correspondence between the columns of matrix M and the cliques in graph G. We assume that all the p players know the graph structure completely as well as both the one-to-one correspondences. The protocol proceeds as follows: for each $k \in [p]$, player P_k determines a random permutation π_k of the vertices in V_k. Also, for each $k \in [p]$, player P_k determines the colors of the vertices in V_k by the following rule: if $M_{ij} = 1$, then color vertex v_{ij} with color C_*. Otherwise, for $M_{ij} = 0$, color vertex v_{ij} with color C_i. Player P_1 initiates the streaming algorithm and it goes over r-rounds.

Rounds 1 to $r-1$: For $k \in [p]$, each player resumes the streaming algorithm by exposing the vertices in V_k, along with their colors, in the order dictated by π_k. Also, P_k adds the respective edges to previously exposed vertices when the current vertex is exposed to satisfy the basic requirement of VA model. This is possible because all players know the graph G and the random partition of the entries of matrix M among p players. After exposing all the vertices in V_k, as described, P_k sends the current memory state to player P_{k+1}. Assume that $P_1 = P_{p+1}$.

Round r: All the players behave similarly as in the previous rounds, except that, the player P_p does not send the current memory state to P_1. Rather, P_p decides whether there is a column in M with all 1s if the streaming algorithm $\mathcal{S}$ decides that there are $\Omega(T)$ many monochromatic edges in G. Otherwise, if $\mathcal{S}$ decides that there is no monochromatic edge in G, then P_p decides that all the columns of M have weight either 0 or 1. Then P_p sends the output to all other players.

The vertices of graph G are indeed exposed randomly to the streaming algorithm. It is because the entries of matrix M are randomly partitioned among the players and each player also generates a random permutation of the vertices corresponding to the entries of matrix M available to them. From the description of the protocol $\mathcal{A}$, the memory state of the streaming algorithm (of space complexity s) is communicated $(r-1)p + (p-1)$ times and $p-1$ bits is communicated at the end by player P_p to broadcast the output. Hence, the communication cost of the protocol $\mathcal{A}$ is at most $O(rsp)$.

Now we are left to prove the correctness of the protocol $\mathcal{A}$. If there is a column in M with all 1s, then all the vertices corresponding to entries of that column are colored with color C_*. Recall that there is a one-to-one correspondence between the columns in matrix M and cliques in the graph G. So, all the vertices of the clique, corresponding to the column having all 1s, are colored with the color C_*. As the size of each clique in the graph G is $\sqrt{T}$, there are at most $\Omega(T)$ monochromatic edges. To prove the converse, assume that there is no column in the matrix M having all 1s. By Unique Intersection Promise, all the

columns have hamming weight at most 1. We will argue that there is no monochromatic edge in G. Consider an edge e in G. By the structure of G, the two vertices of e must be in the same clique, say the j-th clique, that is, let $e = \{v_{i_1 j}, v_{i_2 j}\}$. By the coloring scheme used by the protocols, $v_{i_1 j}$ and $v_{i_2 j}$ are colored according to the values of $M_{i_1 j}$ and $M_{i_2 j}$, respectively. Note that both $M_{i_1 j}$ and $M_{i_2 j}$ belong to j-th column. As the hamming weight of every column is at most 1, there are three possibilities:

(i) $M_{i_1 j} = M_{i_2 j} = 0$, that is, $v_{i_1 j}$ and $v_{i_2 j}$ are colored with color C_{i_1} and C_{i_2}, respectively;
(ii) $M_{i_1 j} = 0$ and $M_{i_2 j} = 1$, that is, $v_{i_1 j}$ and $v_{i_2 j}$ are colored with color C_{i_1} and C_*, respectively;
(iii) $M_{i_1 j} = 1$ and $M_{i_2 j} = 0$, that is, $v_{i_1 j}$ and $v_{i_2 j}$ are colored with color C_* and C_{i_2}, respectively.

In any case, the edge $e = \{v_{i_1 j}, v_{i_2 j}\}$ is not monochromatic. This establishes the correctness of protocol $\mathcal{A}$ for $\textsc{Disjointness}_R\left(\sqrt{T}, \frac{n}{\sqrt{T}}, p\right)$. ◀

5 Conflict-Sep in VArand model

Using a structural property of the graph, we design a simple algorithm to solve the Conflict-Sep problem in the VArand model.

▶ **Theorem 5.1.** *Given any graph $G = (V, E)$ and a coloring function $f : V(G) \to [C]$ and a parameter $\varepsilon > 0$ as input, there exists an algorithm that solves the* Conflict-Sep *problem in the* VArand *streaming model using space $\widetilde{\mathcal{O}}\left(\frac{|V|}{\sqrt{\varepsilon |E|}}\right)$ with high probability.*

Let G' denote the subgraph of G consisting of only monochromatic edges in G. The lemma stated below guarantees that either there exists a large matching of size at least $\sqrt{\varepsilon m}$ in G' or there exists a vertex of degree at least $\sqrt{\varepsilon m}$ in G'.

▶ **Lemma 5.2** ([16])**.** *Let $G = (V, E)$ be a graph and $f : V(G) \to [C]$ be a coloring function such that at least ε fraction of the edges of $E(G)$ are known to be monochromatic. Then, either there is a matching of size at least $\sqrt{\varepsilon m}$ or there exists a vertex of degree at least $\sqrt{\varepsilon m}$ in the subgraph G' defined on the monochromatic edges of G.*

The algorithm is as simple as it can get. We sample independently and uniformly at random the vertices in stream with probability $p = \min\{1, \frac{10 \log n}{\sqrt{m}}\}$ [7] and store these vertices along with their colors. Let $S \subseteq V$ be the set of sampled vertices. When a vertex appears in a stream, we check if it forms a monochromatic edge with one of the stored vertices in S. At the end of the stream, the algorithm declares the graph to be properly colored (valid) if it can not find a monochromatic edge, else it declares the instance to be ε-far from being monochromatic.

We show that Theorem 5.1 follows easily using Lemma 5.2.

[7] For simplicity of presentation, we assumed that, the number of edges m in graph G is known before the stream starts. However, this assumption can be removed by a simple tweak of starting with a value of m and increasing it in stages and adjusting the random sample accordingly. This is common in streaming algorithms.

Proof. We consider the following two cases.

- Case 1 – There exists a matching of size at least $\sqrt{\varepsilon m}$: Note that all these matched edges are monochromatic. Let (u, v) denote an arbitrary matched edge where u appears in the stream before v. Now, the edge (u, v) will be detected as monochromatic if vertex u has been sampled by the algorithm. The probability that vertex u is sampled is $\frac{10 \log n}{\sqrt{m}}$. Since, there are $\sqrt{\varepsilon m}$ matched monochromatic edges, the algorithm will detect at least one of these matched monochromatic edges with probability at least $(1 - 1/n^2)$.
- Case 2 – There exists a vertex of degree at least $\sqrt{\varepsilon m}$: In this case most of the monochromatic edges may be incident on very few high degree vertices. To detect these edges, we want to store either the high degree vertices or one of its neighbours. But, if these high degree vertices appear at the beginning of the stream and we fail to sample them, then we may not detect a monochromatic edge. This is where the *random order* of vertices arriving in the stream comes into play. Now, assuming random order of vertices in the stream, at least $\frac{1}{5}\sqrt{\varepsilon m}$ neighbors of v should appear before v in the stream with probability at least $(1 - e^{-\frac{9}{50}\sqrt{\varepsilon m}})$. Since we sample every vertex with probability $\frac{10 \log n}{\sqrt{m}}$, with high probability at least $(1 - 1/n^2)$ one of its neighbors will be stored. ◀

6 Conclusion and Discussion

In this paper, we introduced a graph coloring problem to streaming setting with a different flavor – the coloring function streams along with the graph. We study the problem of Conflict-Est (estimating the number of monochromatic edges) and Conflict-Sep (detecting a separation between the number of valid edges) in VA, VAdeg, and VArand models. Our algorithms for VA and VAdeg are tight upto polylogarithmic factors. However, a matching lower bound on the space complexity for VArand model is still elusive. There is a gap between our upper and lower bound results for VArand model in terms of the exponent in T. Our hunch is that the upper bound is tight. Specifically, we obtained an upper bound of $\widetilde{\mathcal{O}}\left(\frac{n}{\sqrt{T}}\right)$) and the lower bound is $\Omega\left(\frac{n}{T^2}\right)$. Here we would like to note that the lower bound also holds in AL and VAdeg model when the vertices are exposed in a random order. However, we feel that our algorithm for Conflict-Est in VArand model is tight upto polylogarithmic factors. We leave this problem open.

We feel the *edge coloring* counterpart of the vertex coloring problem proposed in the paper will be worthwhile to study. Let the edges of G be colored with a function $f : E(G) \to [C]$, for $C \in \mathbb{N}$. A vertex $u \in V(G)$ is said to be a *validly* colored vertex if no two edges incident on u have the same color. An edge coloring is valid if all vertices are validly colored. Consider the AL model for the edge coloring problem. As all edges incident on an exposed vertex u are revealed in the stream, if we can solve a duplicate element finding problem on the colors of the edges incident on u, then we are done! It seems at a first glance that all the three models of VA, AL and EA will be difficult to handle for the edge coloring problem on streams of graph and edge colors. It would be interesting to see if the edge coloring variant of the problems we considered in this paper, admit efficient streaming algorithms. We plan to look at this problem next.

References

1 Noga Alon and Sepehr Assadi. Palette sparsification beyond (Δ+1) vertex coloring. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 6:1–6:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.6.

2 Noga Alon and Sepehr Assadi. Palette sparsification beyond (Δ+1) vertex coloring. *CoRR*, abs/2006.10456, 2020. arXiv:2006.10456.

3 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for (Δ + 1) vertex coloring. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 767–786. SIAM, 2019. doi:10.1137/1.9781611975482.48.

4 Soheil Behnezhad, Mahsa Derakhshan, MohammadTaghi Hajiaghayi, Marina Knittel, and Hamed Saleh. Streaming and massively parallel algorithms for edge coloring. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 15:1–15:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.15.

5 Suman K. Bera and Amit Chakrabarti. Towards tighter space bounds for counting triangles and other substructures in graph streams. In Heribert Vollmer and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, volume 66 of *LIPIcs*, pages 11:1–11:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.11.

6 Suman K. Bera, Amit Chakrabarti, and Prantar Ghosh. Graph coloring via degeneracy in streaming and other space-conscious models. *CoRR*, abs/1905.00566, 2019. arXiv:1905.00566.

7 Suman K. Bera and C. Seshadhri. How the degeneracy helps for triangle counting in graph streams. In Dan Suciu, Yufei Tao, and Zhewei Wei, editors, *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 457–467. ACM, 2020. doi:10.1145/3375395.3387665.

8 Suman Kalyan Bera and Prantar Ghosh. Coloring in graph streams. *CoRR*, abs/1807.07640, 2018. arXiv:1807.07640.

9 Anup Bhattacharya, Arijit Bishnu, Gopinath Mishra, and Anannya Upasana. Even the easiest(?) graph coloring problem is not easy in streaming! *CoRR*, abs/2010.13143, 2020. arXiv:2010.13143.

10 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Robust lower bounds for communication and stream computation. *Theory Comput.*, 12(1):1–35, 2016. doi:10.4086/toc.2016.v012a010.

11 Graham Cormode, Jacques Dark, and Christian Konrad. Independent sets in vertex-arrival streams. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 45:1–45:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.45.

12 Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009. URL: http://www.cambridge.org/gb/knowledge/isbn/item2327542/.

13 Guy Even, Magnús M. Halldórsson, Lotem Kaplan, and Dana Ron. Scheduling with conflicts: online and offline algorithms. *J. Sched.*, 12(2):199–224, 2009. doi:10.1007/s10951-008-0089-1.

14 Uriel Feige and Joe Kilian. Zero knowledge and the chromatic number. *J. Comput. Syst. Sci.*, 57(2):187–199, 1998. doi:10.1006/jcss.1998.1587.

15 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

16 Stasys Jukna. *Extremal Combinatorics - With Applications in Computer Science*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2011. doi:10.1007/978-3-642-17364-6.

17 John Kallaugher, Michael Kapralov, and Eric Price. The sketching complexity of graph and hypergraph counting. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 556–567. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00059.

18 John Kallaugher, Andrew McGregor, Eric Price, and Sofya Vorotnikova. The complexity of counting cycles in the adjacency list streaming model. In Dan Suciu, Sebastian Skritek, and Christoph Koch, editors, *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 119–133. ACM, 2019. doi:10.1145/3294052.3319706.

19 Daniel M. Kane, Kurt Mehlhorn, Thomas Sauerwald, and He Sun. Counting arbitrary subgraphs in data streams. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part II*, volume 7392 of *Lecture Notes in Computer Science*, pages 598–609. Springer, 2012. doi:10.1007/978-3-642-31585-5_53.

20 Subhash Khot and Ashok Kumar Ponnuswami. Better inapproximability results for maxclique, chromatic number and min-3lin-deletion. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part I*, volume 4051 of *Lecture Notes in Computer Science*, pages 226–237. Springer, 2006. doi:10.1007/11786986_21.

21 Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

22 Andrew McGregor. Graph stream algorithms: a survey. *SIGMOD Rec.*, 43(1):9–20, 2014. doi:10.1145/2627692.2627694.

23 Andrew McGregor, Sofya Vorotnikova, and Hoa T. Vu. Better algorithms for counting triangles in data streams. In Tova Milo and Wang-Chiew Tan, editors, *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 401–411. ACM, 2016. doi:10.1145/2902251.2902283.

24 Wolfgang Mulzer. Five proofs of chernoff's bound with applications. *Bull. EATCS*, 124, 2018. URL: http://eatcs.org/beatcs/index.php/beatcs/article/view/525.

25 Isabelle Stanton and Gabriel Kliot. Streaming graph partitioning for large distributed graphs. In Qiang Yang, Deepak Agarwal, and Jian Pei, editors, *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 1222–1230. ACM, 2012. doi:10.1145/2339530.2339722.

26 Charalampos E. Tsourakakis, Christos Gkantsidis, Bozidar Radunovic, and Milan Vojnovic. FENNEL: streaming graph partitioning for massive scale graphs. In Ben Carterette, Fernando Diaz, Carlos Castillo, and Donald Metzler, editors, *Seventh ACM International Conference on Web Search and Data Mining, WSDM 2014, New York, NY, USA, February 24-28, 2014*, pages 333–342. ACM, 2014. doi:10.1145/2556195.2556213.

27 V. G. Vizing. On an estimate of the chromatic class of a p-graph, 1964.

28 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007. doi:10.4086/toc.2007.v003a006.

A Some probability results

▶ **Lemma A.1** ([12], Chernoff-Hoeffding bound). *Let $X_1, \ldots, X_n$ be independent random variables such that $X_i \in [0,1]$. For $X = \sum_{i=1}^{n} X_i$ and $\mu = \mathbb{E}[X]$, the following holds for any $0 \leq \delta \leq 1$:*

$$\mathbb{P}(|X - \mu| \geq \delta\mu) \leq 2\exp\left(\frac{-\mu\delta^2}{3}\right)$$

▶ **Lemma A.2** ([24]). *Let $I = \{1, \ldots, N\}$, $r \in [N]$ be a given parameter. If we sample a subset R without replacement, then the following holds for any $J \subset I$ and $\delta \in (0,1)$.*

(i) $\mathbb{P}\left(|J \cap R| \geq (1+\delta)\,|J|\,\frac{r}{N}\right) \leq \exp\left(-\frac{\delta^2|J|r}{3N}\right)$;

(ii) $\mathbb{P}\left(|J \cap R| \leq (1-\delta)\,|J|\,\frac{r}{N}\right) \leq \exp\left(-\frac{\delta^2|J|r}{3N}\right)$;

(iii) *Further more, we have the following if $|J| \leq k$, then the following holds.*

$$\mathbb{P}\left(|J \cap R| \geq (1+\delta)k\frac{r}{N}\right) \leq \exp\left(-\frac{\delta^2 kr}{3N}\right)$$

B Proof of Lemma 3.2

▶ **Lemma B.1** (Restatement of Lemma 3.2).

(i) *For each $j \in [t]$ with $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(\left||B_j \cap R| - \frac{|R||B_j|}{n}\right| \geq \frac{\varepsilon}{10}\frac{|R||B_j|}{n}\right) \leq \frac{1}{n^{10}}$.*

(ii) *For each $j \in [t]$ with $|B_j| < \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(|B_j \cap R| \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}$.*

(iii) *For each vertex v_i with $d_M(v_i) \geq \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(\left|\kappa_{v_i} - \frac{|R|d_M(v_i)}{n}\right| \geq \frac{\varepsilon}{10}\frac{|R|d_M(v_i)}{n}\right) \leq \frac{1}{n^{10}}$.*

(iv) *For each vertex v_i with $d_M(v_i) < \frac{\sqrt{\varepsilon T}}{10t}$, $\mathbb{P}\left(\kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\right) \leq \frac{1}{n^{10}}$.*

Proof. Let us take $N = n, r = |R| = \Gamma = \widetilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$, $I = \{v_1, \ldots, v_n\}$ in Lemma A.2.

(i) Setting $J = B_j$ and $\delta = \frac{\varepsilon}{10}$ in Lemma A.2 (i) and (ii), we have

$$\mathbb{P}\left(\left||B_j \cap R| - \frac{|R|\,|B_j|}{n}\right| \geq \frac{\varepsilon}{10}\frac{|R|\,|B_j|}{n}\right) \leq 2\exp\left(-\frac{(\varepsilon/10)^2\,|B_j|\,\Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$

The last inequality holds as $|B_j| \geq \frac{\sqrt{\varepsilon T}}{10t}$, $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil = \Theta\left(\frac{\log n}{\varepsilon}\right)$ and $\Gamma = \widetilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$.

(ii) Set $J = B_j$, $k = \frac{\sqrt{\varepsilon T}}{10t}$, $\delta = \frac{1}{4}$ in Lemma A.2 (iii). As $|B_j| \leq \frac{\sqrt{\varepsilon T}}{10t}$, $|J| \leq k$. Hence,

$$\mathbb{P}\left(|B_j \cap R| \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\right) \leq \exp\left(-\frac{(1/4)^2(\sqrt{\varepsilon T}/10t)\Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$

(iii) Setting J as the set of monochromatic neighbors of v_i in R and $\delta = \frac{\varepsilon}{10}$ in Lemma A.2 (i) and (ii), we get

$$\mathbb{P}\left(\left|\kappa_{v_i} - \frac{|R|\,d_M(v_i)}{n}\right| \geq \frac{\varepsilon}{10}\frac{|R|\,d_M(v_i)}{n}\right) \leq \exp\left(-\frac{(\varepsilon/10)^2|J|\Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$

The last inequality holds as $|J| = d_M(v_i) \geq \frac{\sqrt{\varepsilon T}}{10t}$, $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil = \Theta\left(\frac{\log n}{\varepsilon}\right)$ and $\Gamma = \widetilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$.

(iv) Set J as the set of monochromatic neighbors of v_i in R, $k = \frac{\sqrt{\varepsilon T}}{10t}$, $\delta = \frac{1}{4}$ in Lemma A.2 (iii). Note that $|J| = d_M(v_i) \leq \frac{\sqrt{\varepsilon T}}{10t} = k$. Hence,

$$\mathbb{P}\left(\kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\right) \leq \exp\left(-\frac{(1/4)^2(\sqrt{\varepsilon T}/10t)\Gamma}{3n}\right) \leq \frac{1}{n^{10}}.$$ ◀

C Communication Complexity

Communication Complexity [21] deals with finding the minimum amount of space that is needed to communicate in order to compute a function when the input to the function is distributed among multiple parties. For the purpose of our work, we are concerned with two player games with one-way communication protocol. The players are traditionally called Alice and Bob. Both of them have a n-bit input string and are unaware of each other's input. The goal is to minimize the bits Alice needs to communicate to Bob so that he can compute a function on both their inputs. No assumption is made on their computational powers and there is no restriction on the amount of time needed for computing the function.

C.1 INDEX problem in the communication complexity model

Lower bound results in the streaming model of computation are proved by reduction from a problem in communication complexity model. We determine our lower bounds by considering a reduction from the INDEX problem in the one-way communication protocol for two players to the specific problem in graphs in the VA model. The INDEX problem is defined as follows: There are two parties, Alice and Bob. Alice has a N-bit input string $X \in \{0,1\}^N$ and Bob has an integer $j \in [N]$. Both are unaware of each other's input and the goal is to compute X_j, the j^{th} bit of X. The lower bound for space complexity to solve the INDEX problem in the one-way deterministic communication model is $\Omega(N)$.

▶ **Lemma C.1** ([21]). *The deterministic communication complexity of INDEX is* $\Omega(N)$

D Algorithm for Conflict-Est in VArand model

- $\Gamma = \widetilde{\Theta}\left(\frac{n}{\sqrt{T}}\right)$; $v_1, \ldots, v_n$ be the random ordering in which vertices are revealed and $R = \{v_1, \ldots, v_\Gamma\}$;
- $\kappa_{v_i}, i \in [n]$, denotes the number of monochromatic neighbors of v_i in R,
- $\widehat{d_{v_i}}, i \in [n]$, denotes the (estimated) monochromatic neighbors of vertices in G.
- H denotes the set of *high* degree vertex in R, i.e., $H = \{v_i : \kappa_{v_i} \geq \frac{|R|}{n}\frac{\sqrt{\varepsilon T}}{8t}\}$ and $L = V \setminus H$; $L_R = L \cap R$ and $H_R = H \cap R$;
- The vertices in L are partitioned into t buckets as follows: $B_j = \{v_i \in L : \left(1+\frac{\varepsilon}{10}\right)^{j-1} \leq d_M(v_i) < \left(1+\frac{\varepsilon}{10}\right)^{j}\}$, where $j \in [t]$.

Algorithm 1 RANDOM-ORDER-EST(ε, T): CONFLICT-EST in VARAND model.

Input: $G = (V, E)$ and a coloring function f on V in the VARAND model, parameters T and ε.

Output: $\widehat{m}$, that is, a $(1 \pm \varepsilon)$ approximation to $|E_M|$.

Set $t = \lceil \log_{1+\frac{\varepsilon}{10}} n \rceil$. If $T < 63t^2$, then store all the vertices in G along with their colors. At the end, report the exact value of $|E_M|$. Otherwise, we proceed through via three building blocks described below and marked as (1),(2), (3) and (4). Refer to the notations described above this pseudocode.

(1) Processing the vertices in R, the first Γ vertices, in the stream:

for *(each vertex $v_i \in R$ exposed in the stream)* **do**
Store v_i as well as its color $f(v_i)$.
For each edge $(v_{i'}, v_i)$ that arrives in the stream, increase the values of $\kappa_{v_{i'}}$ and κ_{v_i}.
end

(2) Computation of some parameters based on vertices in R and their colors:

for *(each $v_i \in R$ with $\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$)* **do**
Add v_i to H_R, and set $\widehat{d_{v_i}} = \frac{n}{|R|}\kappa_{v_i}$.
end
$\widehat{m_h} = \sum_{v_i \in H} \widehat{d_{v_i}}$.
Let $L_R = R \setminus H_R$.
for *(each $v_i \in L_R$)* **do**
Set $\widehat{d_{v_i}} = \kappa_{v_i}$.
end

(3) Processing the vertices in $V(G) \setminus R$ in the stream:

for *(each vertex $v_i \notin R$ exposed in the stream)* **do**
Determine the value of κ_{v_i}. If $\kappa_{v_i} \geq \frac{|R|}{n} \frac{\sqrt{\varepsilon T}}{8t}$, find $\widehat{d_{v_i}} = \frac{n}{|R|}\kappa_{v_i}$ and add $\widehat{d_{v_i}}$ to the current $\widehat{m_h}$.
Also, for each $v_{i'} \in L_R$, increase the value of $\widehat{d_{v_{i'}}}$ if $(v_{i'}, v_i)$ is an edge.
end

(4) Post processing, after the stream ends, to return the output:

From the values of $\widehat{d_{v_i}}$ for all $v_i \in L_R$, determine the buckets for each vertex in L_R. Also, for each $j \in [t]$, find $|A_j| = |L_R \cap B_j|$. Then determine

$$\widehat{m_\ell} = \frac{n}{|R|} \sum_{j \in [t]} |A_j| \left(1 + \frac{\varepsilon}{10}\right)^j.$$

Report $\widehat{m} = \frac{\widehat{m_h} + \widehat{m_\ell}}{2}$ as the final OUTPUT.

E Algorithm for Conflict-Sep inVArand model

Algorithm 2 Algorithm: Conflict-Sep in Vertex Arrival in Random Order model

Input: $G = (V, E)$ and a coloring function f on V in the VArand model
Output: The algorithm verifies if f is ε-far from valid or not
Let S be the set of stored vertices and their colors. Initially, S is empty. **for** $i \leftarrow 1$ *to* $|V|$ **do**
  let u be the i^{th} vertex of the stream
  Store u and its color $f(u)$ in S with probability $\mathcal{O}\left(\frac{\log n}{\sqrt{m}}\right)$
  for *every vertex* v *in* S **do**
    Check if (v, u) is an edge and $f(v) = f(u)$
  end
end
Output f is valid if none of the edges sampled are conflicting, else output that f is ε -far from being valid.