# Approximating the Caro-Wei Bound for Independent Sets in Graph Streams*

Graham Cormode, Jacques Dark, and Christian Konrad

Department of Computer Science, Centre for Discrete Mathematics and its Applications (DIMAP), University of Warwick, Coventry, UK
{g.cormode,j.dark,c.konrad}@warwick.ac.uk

**Abstract.** The Caro-Wei bound states that every graph $G = (V, E)$ contains an independent set of size at least $\beta(G) := \sum_{v \in V} \frac{1}{\deg_G(v)+1}$, where $\deg_G(v)$ denotes the degree of vertex $v$. Halldórsson et al. [1] gave a randomized one-pass streaming algorithm that computes an independent set of expected size $\beta(G)$ using $\mathrm{O}(n \log n)$ space. In this paper, we give streaming algorithms and a lower bound for approximating the Caro-Wei bound itself.

In the edge arrival model, we present a one-pass $c$-approximation streaming algorithm that uses $\mathrm{O}(\overline{d} \log(n)/c^2)$ space, where $\overline{d}$ is the average degree of $G$. We further prove that space $\Omega(\overline{d}/c^2)$ is necessary, rendering our algorithm almost optimal. This lower bound holds even in the *vertex arrival model*, where vertices arrive one by one together with their incident edges that connect to vertices that have previously arrived. In order to obtain a poly-logarithmic space algorithm even for graphs with arbitrarily large average degree, we employ an alternative notion of approximation: We give a one-pass streaming algorithm with space $\mathrm{O}(\log^3 n)$ in the vertex arrival model that outputs a value that is at most a logarithmic factor below the true value of $\beta$ and no more than the maximum independent set size.

## 1 Introduction

For very large graphs, the model of streaming graph analysis, where edges are observed one by one, is a useful lens. Here, we assume that the graph of interest is too large to store in full, but some representative summary is maintained incrementally. We seek to understand how well different problems can be solved in this model, in terms of the size of the summary, the time taken to process each edge and answer a query, and the accuracy of any approximation obtained. Variants arise in the model depending on whether edges can be removed as well as added, and whether edges arrive grouped in some order, and so on.

**Independent Sets and the Caro-Wei Bound.** We study questions pertaining to *independent sets* within graphs. Independent sets play a fundamental role in graph theory,

* The work of GC is supported in part by European Research Council grant ERC-2014-CoG 647557, The Alan Turing Institute under EPSRC grant EP/N510129/1 the Yahoo Faculty Research and Engagement Program and a Royal Society Wolfson Research Merit Award; JD is supported by a Microsoft Research Studentship; and CK by EPSRC grant EP/N011163/1.

and have many applications in optimization and scheduling problems. Given a graph, an independent set is a set of nodes such that there is no edge between any pair. One important objective is to find a *maximum independent set*, i.e., an independent set of maximum cardinality. This is a challenging task even in the offline setting: Computing a maximum independent set is NP-hard on general graphs [2], and remains hard to approximate within a factor of $n^{1-\epsilon}$ for any $\epsilon > 0$ [3,4].

Despite this strong intractability result, there is substantial interest in computing independent sets of non-trivial sizes. The best polynomial time algorithm for maximum independent set was given by Feige and has an approximation factor of $\mathrm{O}(\frac{n \log^2(\log n)}{\log^3 n})$ [5]. Since no substantial improvements on this bound are possible, many works give approximation guarantees or absolute bounds on the size of an independent set in terms of the degrees of the vertices of the input graph. For example, it is known that the Greedy algorithm, which iteratively picks a node of minimum degree and then removes all neighbors from consideration, has an approximation factor of $(\Delta + 2)/3$, where $\Delta$ is the maximum degree of the input graph [6]. The Greedy algorithm also achieves the *Caro-Wei* bound [7,8], which is the focus of this paper: Caro [9] and Wei [7] independently proved that every graph $G$ contains an independent set of size

$$\beta(G) := \sum_{v \in V} \frac{1}{\deg_G(v) + 1} \,. \tag{1}$$

The quantity $\beta(G)$ is an attractive bound. It is known that it gives polylogarithmic approximation guarantees on graphs that are of *polynomially bounded-independence* [10], which means (informally) that the size of a maximum independent set in an $r$-neighborhood around a node is bounded in size by a polynomial in $r$. For example, on unit disc graphs, which are of polynomially bounded-independence, $\beta(G)$ is a $\mathrm{O}\left((\frac{\log n}{\log \log n})^2\right)$ approximation to the size of a maximum independent set. In distributed computing, the Caro-Wei bound is particularly interesting, since an independent set of size $\beta(G)$ can be computed in a single communication round [10]. Very relevant to the present work is a result by Halldórsson et al. [1], who showed that an independent set of expected size $\beta(G)$ can be computed space efficiently in the data streaming model.

**Independent Sets in the Streaming Model.** Due to the aforementioned computational hardness of the maximum independent set problem, every streaming algorithm that approximates a maximum independent set within a polynomial factor $n^{\delta}$, for any constant $\delta < 1$, requires exponential time, unless $\mathsf{P} = \mathsf{NP}$. By sampling a subset of vertices $V' \subseteq V$, storing all edges between vertices $V'$ while processing the stream, and outputting a maximum independent set in the subgraph induced by $V'$ (using an exponential time computation), it is possible to obtain a randomized one-pass $c$-approximation streaming algorithm for maximum independent set with $\tilde{\mathrm{O}}(\frac{n^2}{c^2})$ space[1]. Halldórsson et al. [11] showed that this is best possible: They proved that even for the seemingly simpler task of approximating the size of a maximum independent set, every $c$-approximation streaming algorithm requires $\tilde{\Omega}(\frac{n^2}{c^2})$ space. In order to circumvent

[1] We use the notation $\tilde{\mathrm{O}}(.), \tilde{\Theta}(.)$ and $\tilde{\Omega}(.)$, which correspond to $\mathrm{O}(.)$, $\Theta(.)$ and $\Omega(.)$, respectively, where all polylogarithmic factors are ignored.

both the large space lower bound and the exponential time computations required, in a different work, Halldórsson et al. [1] relaxed the desired quality guarantee and gave one-pass streaming algorithms for computing independent sets with expected sizes that match the Caro-Wei bound. These algorithms use $\mathrm{O}(n \log n)$ space and have constant update times.

**Approximating the Solution Size.** In this paper, we ask whether we can reduce the space requirements of $\mathrm{O}(n \log n)$ even further, if, instead of computing an independent set whose size is bounded by the Caro-Wei bound, we approximate the size of such an independent set, i.e., the Caro-Wei bound itself. This objective ties in with a recent trend in graph streaming algorithms: Since many combinatorial objects such as matchings or independent sets may be of size $\Omega(n)$, streaming algorithms that output such objects require at least this amount of space. Consequently, many recent papers ask whether the task of approximating the output size is easier than outputting the object itself. As previously mentioned, this is not the case for the maximum independent set problem, where the space complexity of both computing a $c$-approximate independent set and finding a $c$-approximation to the size of a maximum independent set is $\tilde{\Theta}(\frac{n^2}{c^2})$ [11]. For the maximum matching problem, it is known that space $\Omega(n/c)$ is needed for computing a $c$-approximation, but space $\tilde{\mathrm{O}}(n/c^2)$ is sufficient for outputting a $c$-approximation to the maximum matching size [12]. However, for graphs with arboricity $c$, the size of a maximum matching can even be approximated within a factor of $\mathrm{O}(c)$ using $\mathrm{O}(c \log^2 n)$ space [13]. Another example is a work by Cabello and Pérez-Lantero [14], which gives a polylogarithmic space streaming algorithm that approximates the maximum size of an independent set of intervals within a constant factor, while storing such a set would require $\Omega(n)$ space.

**Starting Point: Frequency Moments.** Approximating $\beta(G)$ is essentially the same as approximating the $-1$ (negative) frequency moment (or the harmonic mean) of a frequency vector derived from the graph stream. The $p$th frequency moment of a stream of $n$ different items where item $i$ appears $f_i$ times is defined by $F_p = \sum_i |f_i|^p$. Approximating the frequency moments is one of the most studied problems in the data streaming literature, starting in 1996 with the seminal work of Alon, Matias and Szegedy [15]. It is known that all finite positive frequency moments can be approximated with sublinear space (see Woodruff's article [16] for an overview of the problem). Braverman and Chestnut [17] studied the problem of approximating the negative frequency moments, which turn out to be much harder to approximate: Computing a $(1+\epsilon)$-aproximation to the harmonic mean in one pass requires $\Omega(n)$ space if the length of the input sequence is $\Omega(n^2)$. While this lower bound is designed for arbitrary frequency vectors, it can be embedded into a graph with $\Theta(n^2)$ edges so that frequencies correspond to vertex degrees. This implies we cannot find an algorithm to approximate the Caro-Wei bound within a factor of $1+\epsilon$ which guarantees that the space used will always be sublinear.

**Our Results.** Despite these lower bounds, we are able to provide upper and lower bounds that improve on those stated above. The key advance is that they incorporate a dependence on the target quantity, $\beta(G)$. This means when this quantity is suitably big (as is the case in many graphs of interest), we can in fact guarantee sublinear space. In more detail, we proceed as follows. Since in our setting the frequency vector is derived from the degrees of the vertices of the input graph, we can exploit the properties

of the underlying graph. In our first result, we relate the space complexity of our algorithm to a given lower bound $\gamma$ on $\beta(G)$. A meaningful lower bound $\gamma$ is easy to obtain: It is easy to see that the Turán bound [18] for independent sets, which shows that $n/(\overline{d}+1)$ is a lower bound on the size of a maximum independent set, is also a lower bound on $\beta(G)$, where $\overline{d}$ is the average degree of the input graph. Our first result is then a one-pass randomized streaming algorithm with space $\mathrm{O}(\frac{n \log n}{\gamma c^2})$ that approximates $\beta(G)$ within a factor of $c$ with high probability (**Theorem 1**). Using $\gamma = \frac{n}{\overline{d}+1}$, the space becomes $\mathrm{O}(\frac{\overline{d} \log n}{c^2})$, which is polylogarithmic for graphs of constant average degree such as planar graphs or bounded arboricity graphs. The algorithm can also give a $(1+\epsilon)$-approximation using $\mathrm{O}(\frac{n \log n}{\gamma \epsilon^2})$ space.

We prove that our algorithm is best possible (up to poly-log factors). Via a reduction from a hard problem in communication complexity, we show that every $p$-pass streaming algorithm for computing a $c$-approximation to $\beta(G)$ requires $\Omega(\frac{n}{\beta(G)c^2p})$ space (**Theorem 4**). This lower bound also holds in the *vertex arrival order*, where vertices arrive one by one together with those incident edges that connect to vertices that have previously arrived (see Section 2 for a more precise definition). Our lower bound is more general than the lower bound from Braverman and Chestnut [17], since their lower bound only holds for $(1+\epsilon)$-approximation algorithms and does not establish a dependency on the output quantity, i.e., the $-1$-negative frequency moment. Furthermore, their bound was not developed in the graphical setting where frequencies are derived from the vertex degrees.

Our lower bound shows that the promise that the input stream is in vertex arrival order is not helpful for approximating $\beta(G)$. However, if we regard the task of approximating $\beta(G)$ as obtaining a (hopefully large) lower bound on the size of a maximum independent set of the input graph, then any value sandwiched between $\beta(G)$ and the maximum independent set size would be equally suitable (or even superior). In the vertex arrival setting, we give a randomized one-pass streaming algorithm with space $\mathrm{O}(\log^3 n)$, which outputs a value $\beta'$ with $\beta' = \Omega(\beta(G)/\log n)$ and $\beta'$ is at most the maximum independent set size (**Theorem 2**). Since the Caro-Wei bound is a polylogarithmic approximation to the maximum independent set size in polynomially bounded-independence graphs, a corollary of our result is that the maximum independent set size can be approximated within a polylogarithmic factor in polylogarithmic space in polynomially bounded-independence graphs (e.g., the approximation factor obtained on unit disc graphs is $\mathrm{O}(\frac{\log^3 n}{(\log \log n)^2})$) .

Our focus is on streaming models where edges only arrive. We briefly comment on when our results generalize to models which allow deletions following each algorithm.

**Further Related Work.** There has been substantial interest in the topic of streaming algorithms for graphs in the last two decades. Indeed, the introduction of the streaming model focused on graph problems [19]. McGregor provides a survey that outlines key results on well-studied problems such as finding sparsifiers, identifying connectivity structure, and building spanning trees and matchings [20].

Our work is the first to consider the graph frequency moments (or degree moments) in the data streaming model. They have previously been considered in the property testing literature [21,22,23], where the input graph can only be queried a sublinear

number of times. There are important connections between the degree moments and network science and various other disciplines. For details we refer the reader to [22].

## 2 Preliminaries

The Independent Set problem is most naturally modeled as a problem over graphs $G = (V, E)$. A set $U \subseteq V$ is an independent set if for all pairs $u, w \in U$ we have $\{u, w\} \notin E$, i.e. there is no edge between $u$ and $w$. Let $\alpha(G)$ be the *independence number* of graph $G$, i.e., the size of a maximum independent set in $G$.

We consider graphs defined by streams of edges. That is, we observe a sequence of unordered pairs $\{u, w\}$ which collectively define the (current) edge set $E$. We do not require $V$ to be given explicitly, but take it to be defined implicitly as the union of all nodes observed in the stream. In the (arbitrary, possibly adversarial) edge arrival model, no further constraints are placed on the order in which the edges arrive. In the vertex arrival model, there is a total ordering on the vertices $\prec$ which is revealed incrementally. Given the final graph $G$, node $v$ "arrives" so that all edges $\{u, v\} \in E$ such that $u \prec v$ are presented sequentially before the next vertex arrives. We do not assume that there is any further ordering among this group of edges.

## 3 Algorithm in the Edge-arrival Model

In this section, we suppose that a lower bound $\gamma \leq \beta(G)$ is known. For example, $\gamma = \frac{n}{\overline{d}+1}$ is a suitable bound, where $\overline{d}$ is the average degree of the input graph. If no such bound is known, then the algorithm can be used with the trivial lower bound $\gamma = 1$.

We give an algorithm that computes an estimate $B$ which approximates $\beta(G)$ within a factor of $1 + \epsilon$ with probability at least $2/3$. By running $\Theta(\log n)$ copies of our algorithm and returning the median of the computed estimates, the success probability can be increased to $1 - \frac{1}{n^c}$, for any constant $c$.

The estimator $B$ is computed as follows: First, take a uniform random sample $S \subseteq V$ such that every vertex $v \in V$ is included in $S$ with probability $p = \frac{3}{\epsilon^2 \gamma}$. Then, while processing the stream, compute $\deg_G(v)$, for every vertex $v \in S$. Let $x_v \in \{0, 1\}$ be the indicator variable of the event $v \in S$. Then $B$ is computed by $B = \frac{1}{p} \sum_{v \in V} a_v x_v$, where $a_v := \frac{1}{\deg_G(v)+1}$.

We first show that $B$ is an unbiased estimator and we bound the variance of $B$.

**Lemma 1.** *Let $B$ be the estimate computed as above. Then:*

$$\mathbb{E}[B] = \beta(G)\,, \text{ and } \mathbb{V}[B] < \frac{1}{p} \sum_{v \in V} a_v^2 \leq \frac{1}{p} \beta(G)\,.$$

The proof is a fairly straightforward calculation of expectations, and is deferred to the appendix.

**Theorem 1.** *Let $\gamma \leq \beta(G)$ be a given lower bound on $\beta(G)$. Then, there is a randomized one-pass approximation streaming algorithm in the edge arrival model with space $\mathrm{O}\left(\frac{n \log n}{\gamma \epsilon^2}\right)$ that approximates $\beta(G)$ within a factor of $1 + \epsilon$, with high probability.*

**Algorithm 1** Algorithm DEGTEST$(d, \epsilon)$

**Require:** Degree bound $d$, $\epsilon$ for a $1+\epsilon$ approximation

$p \leftarrow 1$, $S \leftarrow \varnothing$, $m \leftarrow 0$, $\epsilon' \leftarrow \epsilon/2$, $c \leftarrow \frac{28}{\epsilon'^2}$
**while** stream not empty **do** {The current subgraph is $G_i$}
  $v \leftarrow$ next vertex in stream
  **if** COIN($p$) **then**
    $S \leftarrow S \cup \{v\}$ {Sample vertex with probability $p$}
  Update degrees of vertices in $S$, i.e., for every $u \in S$ adjacent to $v$, increment its degree {This ensures that for every $u \in S$ $\deg_{G_i}(u)$ is known}
  Remove every vertex $u \in S$ from $S$ if $\deg_{G_i}(u) > d$
  **if** $p = 1$ **then**
    $m \leftarrow \max\{m, |S|\}$
  **if** $|S| = c\log(n)$ **then**
    $m \leftarrow c\log(n)/p$
    Remove each element from $S$ with probability $\frac{1}{1+\epsilon'}$
    $p \leftarrow p/(1+\epsilon')$
**return** $m$

*Proof.* By Chebyshev's inequality, the error probability of our estimate is at most $1/3$, since (recall that $p = \frac{3}{\epsilon^2\gamma}$)

$$\mathbb{P}\left[|B - \beta(G)| \geq \epsilon\beta(G)\right] \leq \frac{\mathbb{V}[B]}{\epsilon^2\beta(G)^2} < \frac{1}{p\epsilon^2\beta(G)} \leq \frac{1}{3}\,.$$

By a standard Chernoff bounds argument, running $\Theta(\log n)$ copies of our algorithm and returning the median of the computed estimates allows us to obtain an error probability of $\mathrm{O}(n^{-c})$, for any constant $c$. □

*Remarks:* Observe that the previous theorem also holds for large values of $\epsilon$ (e.g. $\epsilon = n^{\delta}$, for some $\delta > 0$). The core of our algorithm is to sample nodes with a fixed probability and to count their degree. This can easily be achieved in the model where edges are also deleted (the turnstile streaming model) without any further data structures, so our results hold in that stream model also.

## 4 Algorithm in the Vertex-arrival Model

Let $v_1, \ldots, v_n$ be the order in which the vertices appear in the stream. Let $G_i = G[\{v_1, \ldots, v_i\}]$ be the subgraph induced by the first $i$ vertices. Let $n_{d,i} := |\{v \in V(G_i) : \deg_{G_i}(v) \leq d\}|$ be the number of vertices of degree at most $d$ in $G_i$, and let $n_d = \max_i n_{d,i}$.

We first give an algorithm, DEGTEST$(d, \epsilon)$, which with high probability returns a $(1+\epsilon)$-approximation of $n_d$ using $\mathrm{O}(\frac{1}{\epsilon^2}\log^2 n)$ bits of space. In the description of the algorithm, we suppose that we have a random function COIN: $[0,1] \to \{\texttt{false}, \texttt{true}\}$ such that COIN($p$) = `true` with probability $p$ and COIN($p$) = `false` with probability $1-p$. Furthermore, the outputs of repeated invocations of COIN are independent.

Algorithm DEGTEST$(d, \epsilon)$ maintains a sample $S$ of at most $c \log n$ vertices. It ensures that all vertices $v \in S$ have degree at most $d$ in the current graph $G_i$ (notice that $\deg_{G_i}(v) \leq \deg_{G_j}(v)$, for every $j \geq i$). Initially, $p = 1$, and all vertices of degree at most $d$ are stored in $S$. Whenever $S$ reaches the limiting size of $c \log n$, we downsample $S$ by removing every element of $S$ with probability $\frac{1}{1+\epsilon'}$ and update $p \leftarrow p/(1+\epsilon')$. This guarantees that throughout the algorithm $S$ constitutes a uniform random sample of all vertices of degree at most $d$ in $G_i$.

The algorithm outputs $m \leftarrow c \log(n)/p$ as the estimate for $n_d$, where $p$ is the smallest value of $p$ that occurs during the course of the algorithm. It is updated whenever $S$ reaches the size $c \log n$, since $S$ is large enough at this moment to be used as an accurate predictor for $n_{d,i}$, and hence also for $n_d$.

**Lemma 2.** *Let* $0 < \epsilon \leq 1$. DEGTEST$(d, \epsilon)$ *(Algorithm 1) approximates* $n_d$ *within a factor* $1 + \epsilon$ *with high probability, i.e.,*

$$\frac{n_d}{1+\epsilon} \leq \text{DEGTEST}(d, \epsilon) \leq (1+\epsilon) n_d \,,$$

*and uses* $\mathrm{O}(\frac{1}{\epsilon^2} \log^2 n)$ *bits of space.*

For space reasons, we defer the proof of this Lemma to the appendix and only give a brief outline here. We say that the algorithm is in phase $i$ if the current value of $p$ is $p = 1/(1+\epsilon')^i$. We focus on the key moments $(j_i)_{i \geq 0}$ in the algorithm, where $j_i$ is the smallest index $j$ such that $n_{d,j} \geq c \log n (1+\epsilon')^i (1+\epsilon'/2)$. The core of our proof is to show that after iteration $j_i$, the algorithm is in phase $i+1$ with high probability. For an intuitive justification of this claim, suppose that this is not true and the algorithm was in phase at most $i$ after iteration $j_i$. Then, since $S$ is a uniform sample, we expect the size of $S$ to be at least $n_{d,j_i}/(1+\epsilon')^i \geq c \log n (1+\epsilon'/2)$, which however would have triggered the downsampling step in Line 10 of the algorithm and would have transitioned the algorithm into the next phase. On the other hand, suppose that the algorithm was in phase at least $i+2$ after iteration $j_i$. In iteration $k$ when the algorithm transitioned into phase $i+2$, the number of nodes $n_{d,k}$ of degree at most $d$ was bounded by $n_{d,k} \leq n_{d,j_i}$. The transition from phase $i+1$ to $i+2$ would thus not have occurred, since the expected size of $S$ in iteration $k$ was at most $n_{d,j_i}/(1+\epsilon')^{i+1} \leq c \log n (1+\epsilon'/2)/(1+\epsilon')$. In our proof, we make this intuition formal and conduct an induction over the phases. Let $j_{\tilde{i}}$ be the largest occurring value of $j_i$. Then $n_{d,j_{\tilde{i}}}$ is a good approximation of $n_d$ and, as argued above, the algorithm is in phase $\tilde{i}+1$ after iteration $j_{\tilde{i}}$. Using the largest occurring value of $p$, we can thus estimate $n_d$.

Next, we run multiple copies of DEGTEST in order to obtain our main algorithm, Algorithm 2. This consists of making multiple parallel guesses of the parameter $d$ as powers of 2, and taking the guess which provides the maximum bound.

**Theorem 2.** *Let* $\gamma$ *be the output of Algorithm 2. Then, with high probability:*

1. $\gamma = \Omega(\frac{\beta(G)}{\log n})$, *and*
2. $\gamma \leq \alpha(G)$.

*Furthermore, the algorithm uses space* $\mathrm{O}(\log^3 n)$ *bits.*

**Algorithm 2** Algorithm in the Vertex-arrival Order

**for** every $i \in \{0, 1, \ldots, \lceil \log n \rceil\}$, run in parallel:
  $\tilde{n}_{2^i} = \textsc{DegTest}(2^i, 1/2)$
**end for**
**return** $\max\left\{ \frac{\tilde{n}_{2^i}}{2(2^i+1)} \,:\, i \in \{0, 1, \ldots, \lceil \log n \rceil\} \right\}$

*Proof.* For $0 \leq i < \lceil \log(n) \rceil$, let $V_i \subseteq V$ be the subset of vertices with $\deg_G(v) \in \{2^i, 2^{i+1} - 1\}$. Then,

$$\beta(G) = \sum_{v \in V} \frac{1}{\deg_G(v) + 1} = \sum_i \sum_{v \in V_i} \frac{1}{\deg_G(v) + 1} \leq \sum_i \frac{|V_i|}{2^i + 1}.$$

Let $i_{\max} := \arg\max_i \frac{|V_i|}{2^i+1}$. Then, we further simplify the previous inequality:

$$\beta(G) \leq \cdots \leq \sum_i \frac{|V_i|}{2^i + 1} \leq \lceil \log(n) \rceil \cdot \frac{|V_{i_{\max}}|}{2^{i_{\max}} + 1} \leq \lceil \log(n) \rceil \cdot \frac{|V_{\leq i_{\max}}|}{2^{i_{\max}} + 1}. \qquad (2)$$

where $V_{\leq i} = \cup_{j \leq i} V_j$. Let $d_{\max} = 2^{i_{\max}}$. Since $|V_{i_{\max}}| \leq n_{d_{\max}}$ and $\tilde{n}_{d_{\max}} = \textsc{DegTest}(d_{\max}, \frac{1}{2})$ is a 1.5-approximation to $n_{d_{\max}}$, we obtain $\gamma = \Omega(\frac{\beta(G)}{\log n})$, which proves Item 1.

Concerning Item 2, notice that for every $i$ and $d$, it holds

$$\begin{aligned} \alpha(G) \geq \alpha(G_i) \geq \beta(G_i) &= \sum_{v \in V(G_i)} \frac{1}{\deg_{G_i}(v) + 1} \\ &\geq \sum_{v \in V(G_i) : \deg_{G_i}(v) \leq d} \frac{1}{\deg_{G_i}(v) + 1} \geq \frac{n_{i,d}}{d+1}, \end{aligned}$$

and, in particular, the inequality holds for $n_{d_{\max}} = n_{i_{\max}, d_{\max}}$. Since the algorithm returns a value bounded by $\frac{\tilde{n}_{d_{\max}}}{2 \cdot (d_{\max}+1)}$, and $\tilde{n}_{d_{\max}}$ constitutes a 1.5-approximation of $n_{d_{\max}}$, Item 2 follows.

Concerning the space requirements, the algorithm runs $\mathrm{O}(\log n)$ copies of Algorithm 1 which itself requires $\mathrm{O}(\log^2 n)$ bits of space. □

*Remark:* On first glance, it may appear that our algorithm would translate to the turnstile model where edges can be deleted: the central step of sampling vertices at varying probabilities is reminiscent of steps from $L_0$ sampling algorithms [24]. However, there are a number of obstacles to achieving this. First, the algorithm computes a maximum over the estimate $\beta(G_i)$ for intermediate graphs $G_i$. This is correct when nodes and edges only arrive, but is not correct when a graph may be subject to deletions. We therefore leave the question of giving comparable bounds under the turnstile stream model as an open problem.

## 5 Space Lower Bound

Our lower bound follows from a reduction using a well-known hard problem from communication complexity. Let $\mathrm{DISJ}_n$ refer to the *two-party set disjointness problem* for

inputs of size $n$. In this problem we have two parties, Alice and Bob. Alice knows $X \subset [n]$, while Bob knows $Y \subset [n]$. Alice and Bob must exchange messages until they both know whether $X \cap Y = \emptyset$ or $X \cap Y \neq \emptyset$.

Using $R(\text{DISJ}_n)$ to refer to the randomised (bounded error probability) communication complexity of $\text{DISJ}_n$, the following theorem is known.

**Theorem 3 (Kalyanasundaram and Schnitger [25]).** $R(\text{DISJ}_n) \in \Omega(n)$ .

To get our lower bound, we will show a reduction from randomised set disjointness to randomised $c$-approximation of $\beta(G)$.

**Theorem 4.** *Every randomized constant error $p$-pass streaming algorithm that approximates $\beta(G)$ within a factor of $c$ uses space $\Omega\left(\frac{n}{\beta(G)c^2p}\right)$, even if the input stream is in vertex arrival order.*

*Proof.* Let $\text{ALG}_{c,n}$ be any streaming algorithm that performs $p$ passes over a vertex arrival stream of an $n$-vertex graph $G$ and returns a $c$-approximation of $\beta(G)$ with probability $\frac{2}{3}$. Suppose we are given an instance of $\text{DISJ}_k$. We will construct a graph $G$ from $X$ and $Y$ which we can use to tell whether $X \cap Y = \emptyset$ by checking a $c$-approximation of $\beta(G)$.

Let $z \geq 2$ be an arbitrary integer. Set $q = 2zc^2$ and $a = kq$. Let $G = (V, E)$, where $V$ is partitioned into disjoint subsets $A$, $B$, $C$, and $U_i$ for $i \in [k]$. These are of size $|A| = |B| = a$, $|C| = z$, and $|U_i| = q$. So $n := |V| = kq + 2a + z = 3kq + z = z(6kc^2 + 1)$. Thus, $k \in \Theta(\frac{n}{zc^2})$ holds.

First consider the set of edges $E_0$ consisting of all $\{u, v\}$ with $u, v \in A \cup B$, $u \neq v$. Setting $E = E_0$ makes $A \cup B$ a clique, while all other vertices remain isolated.

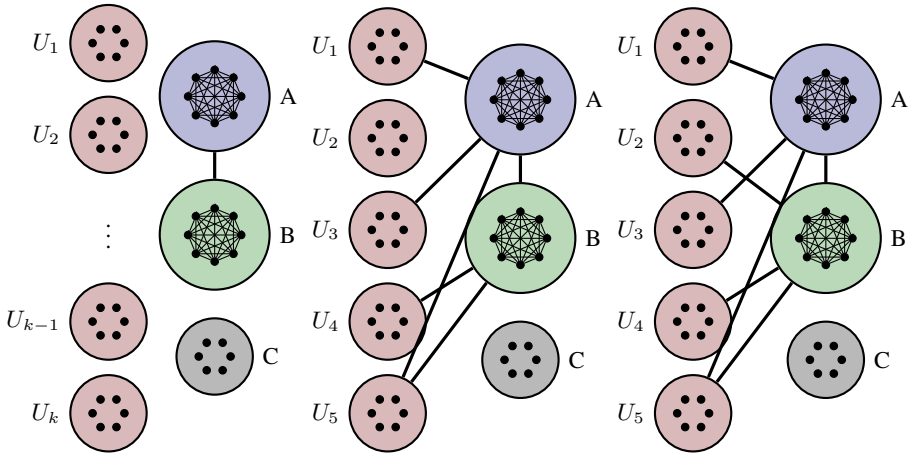
Figure 1a shows this initial configuration. For clarity, we represent the structure using super-nodes and super-edges. A super-node is a subset of $V$ (in this case we use $A$, $B$, $C$, and each $U_i$). Between the super-nodes, we have super-edges representing the existence of all possible edges between constituent vertices. So a super-edge between super-nodes $Z_1$ and $Z_2$ represents that $\{z_1, z_2\} \in E$ for every $z_1 \in Z_1$ and $z_2 \in Z_2$. The lack of a super-edge between $Z_1$ and $Z_2$ indicates that none of these $\{z_1, z_2\}$ are in $E$.

Next we add dependence on $X$ and $Y$. Let

$$E_X = \bigcup_{i \in [n] \setminus X} \left( \bigcup_{u \in U_i, v \in A} \{\{u, v\}\} \right) \text{ and } E_Y = \bigcup_{i \in [n] \setminus Y} \left( \bigcup_{u \in U_i, v \in B} \{\{u, v\}\} \right).$$

So $E_X$ contains all edges from vertices in $U_i$ to vertices in $A$ exactly when index $i$ is not in the set $X$. $E_Y$ similarly contains all edges from $U_i$ to $B$ when $i \notin Y$.

Now let $E = E_0 \cup E_X \cup E_Y$. Adding these edge sets corresponds to adding a super-edge to Figure 1a between $U_i$ and $A$ (or $B$) whenever $i$ is not in $X$ (or $Y$). Figures 1b and 1c illustrate this. In Figure 1b, the intersection is non-empty, which creates a set of isolated nodes that push up the value of $\beta(G)$. Meanwhile, there is no intersection in Figure 1c, so the only isolated nodes are those in $C$.




(a) Initial configuration.

(b) Example with $X = \{2, 4\}$ and $Y = \{1, 2, 3\}$.

(c) Example with $X = \{2, 4\}$ and $Y = \{1, 3\}$.

Fig. 1: Lower bound construction

Now, consider $\beta(G)$. In the case where $X \cap Y = \emptyset$, we will have a super-edge connecting each $U_i$ to at least one of $A$ and $B$, so the degree of each vertex in each $U_i$ is either $a$ or $2a$. Similarly, $A \cup B$ is a clique, so each vertex has degree at least $(2a - 1)$. There are $2a$ such vertices, so they contribute at most $\frac{2a}{(2a-1)+1} = 1$ to $\beta$. Vertices in $C$ are isolated and contribute exactly $z$ to $\beta$. Therefore, $z \leq \beta(G) \leq \frac{kq}{a} + 1 + z = z + 2$.

Now consider the case where $X \cap Y \neq \emptyset$. This means that there exists some $i \in X \cap Y$, and so $U_i$ will have no super-edges. So each vertex in $U_i$ is isolated, and contributes exactly 1 to $\beta$. There are $q$ such vertices, and also accounting for the contribution of vertices $C$, we obtain $\beta(G) \geq q + z = z(2c^2 + 1)$.

Since the minimum possible ratio of the $\beta$-values between graphs in the two cases is at least $\frac{z(2c^2+1)}{z+2} > c^2$ (using $z \geq 2$), a $c$-approximation algorithm for $\beta(G)$ would allow us to distinguish between the two cases.

Now, return to our instance of $\text{DISJ}_k$. We can have Alice initialise an instance of $\text{ALG}_{c,n}$ and have all vertices in $A$, $C$, and each $U_i$ arrive in any order. This only requires knowledge of $X$ because only edges in $E_0$ and $E_X$ are between these vertices and these are the only edges that will be added so far in the vertex arrival model. Alice then communicates the state of $\text{ALG}_{c,n}$ to Bob. Bob can now have all vertices in $B$ arrive in any order. This only requires knowledge of $Y$ because only edges in $E_0$ and $E_Y$ are still to be added. Bob then communicates the state of $\text{ALG}_{c,n}$ back to Alice (if $p \geq 2$). This process continues until the $p$ passes of the algorithm have been executed. Bob can then compute a $c$-approximation of $\beta(G)$ with probability at least $\frac{2}{3}$ from the final state of the algorithm, determining which case we are in and solving $\text{DISJ}_k$.

From Theorem 3, we know that Alice and Bob must have communicated at least $\Omega(k)$ bits. However, all they communicated was the state of $\text{ALG}_{c,n}$. Therefore, $\Omega(k/p) =$

$\Omega(\frac{n}{zc^2p})$ bits was being used by $\mathrm{ALG}_{c,n}$ at the time, since the algorithm runs in $p$ passes.

Consider again the graph $G$. The above argument shows that in order to compute a $c$-approximation to $\beta(G)$, space $\Omega(\frac{n}{zc^2p})$ is needed. Since $\beta(G) \geq z$ in both cases, we obtain the space bound $\Omega(\frac{n}{\beta(G)c^2p})$. Last, recall that $z$ and thus $\beta(G)$ can be chosen arbitrarily. The theorem hence holds for any value of $\beta(G)$. □

*Remark:* The vertices of set $C$ of the construction employed in the previous proof are isolated. This property may be considered undesirable – for example, it may be relatively easy to identify and separately count isolated vertices. However, this structure in the hard instances can be entirely circumvented by, for example, replacing each of these vertices $u \in C$ with a pair of nodes $u_1, u_2$, which are connected by an edge. We also note that, of course, the problem is no easier when deletions are allowed, and so the lower bound also holds for such models.

## 6 Conclusion

In this paper, we gave an optimal one-pass $c$-approximation streaming algorithm with space $\mathrm{O}(\frac{n \operatorname{polylog} n}{c^2\gamma})$ for approximating the Caro-Wei bound $\beta(G)$ in graph streams, where $\gamma \leq \beta(G)$ is a given lower bound. If the input stream is in vertex arrival order, then we showed that a quantity $\beta'$ can be computed, which is at most a logarithmic factor below $\beta(G)$ and at most $\alpha(G)$, the maximum independent set size of the input graph.

From a technical perspective, we leverage this problem to advance the study of the degree moments in the streaming model. The fact that the frequencies are derived from the degrees of the input graph adds an additional dimension to the frequency moments problem, since, as illustrated by our two algorithms, the arrival order of edges can now be exploited. Furthermore, it seems plausible that exploiting additional graph structure could reduce the space complexity even further. For example, it is known that in claw-free graphs, it holds $\sum_{u \in \Gamma(v)} \frac{1}{\deg(u)} = \mathrm{O}(1)$, for every vertex $v$ [10]. It remains to be investigated whether such properties can give additional space improvements. Last, one of the objectives of this work was the popularization of the Caro-Wei bound, and we thus only addressed the $-1$-negative frequency moment. Our algorithm for the edge arrival model can in fact also be used for approximating any other negative degree moment $\sum_{v \in V} (\frac{1}{\deg_G(v)})^p$, for every $p < 0$, since the analysis only requires that the contribution of a vertex to the degree moment is at most 1, which is the case for all negative moments (it holds $(\frac{1}{\deg_G(v)})^p \leq 1$, for every $v \in V$ and $p < 0$). Generalizing our approach to positive frequency moments is left for future work.

**Acknowledgements.** We thank an anonymous reviewer whose comments helped us simplify Theorem 1. The work of GC is supported in part by European Research Council grant ERC-2014-CoG 647557; JD is supported by a Microsoft EMEA scholarship and the Alan Turing Institute under the EPSRC grant EP/N510129/1; CK is supported by EPSRC grant EP/N011163/1.

## References


1. Halldórsson, B.V., Halldórsson, M.M., Losievskaja, E., Szegedy, M.: Streaming algorithms for independent sets in sparse hypergraphs. Algorithmica **76**(2) (2016) 490–501
2. Karp, R.M.: Reducibility among combinatorial problems. In Miller, R.E., Thatcher, J.W., eds.: Complexity of Computer Computations. Plenum Press (1972) 85–103
3. Håstad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. Acta Mathematica **182**(1) (1999) 105–142
4. Zuckerman, D.: Linear degree extractors and the inapproximability of max clique and chromatic number. Theory of Computing **3**(1) (2007) 103–128
5. Feige, U.: Approximating maximum clique by removing subgraphs. SIAM J. Discret. Math. **18**(2) (February 2005) 219–225
6. Halldórsson, M., Radhakrishnan, J.: Greed is good: Approximating independent sets in sparse and bounded-degree graphs. In: STOC. (1994) 439–448
7. Wei, V.: A lower bound on the stability number of a simple graph. Technical report, Bell Labs (1981)
8. Griggs, J.R.: Lower bounds on the independence number in terms of the degrees. Journal of Combinatorial Theory, Series B **34**(1) (1983) 22 – 39
9. Caro, Y.: New results on the independence number. Technical report, Tel Aviv University (1979)
10. Halldórsson, M.M., Konrad, C.: Distributed large independent sets in one round on bounded-independence graphs. In: Distributed Computing. (2015) 559–572
11. Halldórsson, M.M., Sun, X., Szegedy, M., Wang, C.: Streaming and communication complexity of clique approximation. In: International Colloquium on Automata, Languages, and Programming. (2012) 449–460
12. Assadi, S., Khanna, S., Li, Y.: On estimating maximum matching size in graph streams. In: ACM-SIAM Symposium on Discrete Algorithms. (2017) 1723–1742
13. Cormode, G., Jowhari, H., Monemizadeh, M., Muthukrishnan, S.: The sparse awakens: Streaming algorithms for matching size estimation in sparse graphs. In: ESA. (2017)
14. Cabello, S., Pérez-Lantero, P. In: Interval Selection in the Streaming Model. Springer International Publishing, Cham (2015) 127–139
15. Alon, N., Matias, Y., Szegedy, M.: The space complexity of approximating the frequency moments. Journal of Computer and System Sciences **58**(1) (1999) 137 – 147
16. Woodruff, D.P.: Frequency moments. In: Encyclopedia of Database Systems. Springer (2009) 1169–1170
17. Braverman, V., Chestnut, S.R.: Universal sketches for the frequency negative moments and other decreasing streaming sums. In: APPROX/RANDOM. (2015) 591–605
18. Turán, P.: On an extremal problem in graph theory. Mat. Fiz. Lapok **48**(436-452) (1941) 137
19. Henzinger, M., Raghavan, P., Rajagopalan, S.: Computing on data streams. Technical Report SRC 1998-011, DEC Systems Research Centre (1998)
20. McGregor, A.: Graph stream algorithms: a survey. SIGMOD Record **43**(1) (2014) 9–20
21. Gonen, M., Ron, D., Shavitt, Y.: Counting stars and other small subgraphs in sublinear-time. SIAM J. Discrete Math. **25**(3) (2011) 1365–1411
22. Eden, T., Ron, D., Seshadhri, C.: Sublinear time estimation of degree distribution moments: The arboricity connection. CoRR **abs/1604.03661** (2016)
23. Aliakbarpour, M., Biswas, A.S., Gouleakis, T., Peebles, J., Rubinfeld, R., Yodpinyanee, A.: Sublinear-time algorithms for counting star subgraphs with applications to join selectivity estimation. CoRR **abs/1601.04233** (2016)
24. Jowhari, H., Sağlam, M., Tardos, G.: Tight bounds for lp samplers, finding duplicates in streams, and related problems. In: ACM Principles of Database Systems. (2011)

25. Kalyanasundaram, B., Schnitger, G.: The probabilistic communication complexity of set intersection. SIAM Journal on Discrete Mathematics **5**(4) (1992) 545–557

## A Deferred Proofs

*Proof (Proof of Lemma 1).* Concerning the expected value, we have:

$$\mathbb{E}[B] = \mathbb{E}\left[\frac{1}{p}\sum_{v\in V} a_v x_v\right] = \frac{1}{p}\sum_{v\in V} a_v \mathbb{E}[x_v] = \sum_{v\in V} a_v = \beta(G)\ .$$

To bound the variance, we use the equality $\mathbb{V}[B] = \mathbb{E}\left[B^2\right] - \mathbb{E}[B]^2$ and bound both terms separately. We have

$$B^2 = \left(\frac{1}{p}\sum_{v\in V} a_v x_v\right)^2 = \frac{1}{p^2}\left(\sum_{u,v\in V, u\neq v} a_u a_v x_u x_v + \sum_{v\in V} a_v^2 x_v^2\right)\ ,$$

and since $x_u$ and $x_v$ are independent if $u \neq v$, we obtain:

$$\mathbb{E}\left[B^2\right] = \frac{1}{p^2}\left(\sum_{u,v\in V, u\neq v} a_u a_v p^2 + \sum_{v\in V} a_v^2 p\right) = \sum_{u,v\in V, u\neq v} a_u a_v + \frac{1}{p}\sum_{v\in V} a_v^2\ . \quad (3)$$

Next, we bound $\mathbb{E}[B]^2$ as follows:

$$\mathbb{E}[B]^2 = \beta(G)^2 = (\sum_{v\in V} a_v)^2 = \sum_{u,v\in V} a_u a_v > \sum_{u,v\in V, u\neq v} a_u a_v\ . \quad (4)$$

Combining Equality 3 and Inequality 4 gives $\mathbb{V}[B] < \frac{1}{p}\sum_{v\in V} a_v^2$. Since $a_v \leq 1$, we further have $\frac{1}{p}\sum_{v\in V} a_v^2 \leq \frac{1}{p}\beta(G)$, which completes the proof. □

*Proof (Proof of Lemma 2).* First, suppose that $n_d < c\log n$. Then the algorithm never downsamples the set $S$ and computes $n_d$ exactly (and makes no error).

Assume now that $n_d \geq c\log n$. For $i \geq 0$, let $j_i$ be the smallest index $j$ such that $n_{d,j} \geq c\log n(1+\epsilon')^i(1+\epsilon'/2)$. We say that the algorithm is in phase $i$, if $p = 1/(1+\epsilon')^i$.

First, for any $i$, we argue that in iteration $k \leq j_i$, the algorithm is in a phase at most $i+1$ w.h.p. Let $E_{k,i}$ be the event that the transition from phase $i+1$ to $i+2$ occurs in iteration $k \leq j_i$, and let $E$ be the event that at least one of the events $E_{k,i}$, for every $k$ and $i$, occurs. For $E_{k,i}$ to happen, it is necessary that the algorithm is in phase $i+1$ in iteration $k$. Assume that this is the case. Then, since $n_{d,k} \leq n_{d,j_i}$, the expected size of $S$ in iteration $k$ is

$$\mathbb{E}[S] = \frac{n_{d,k}}{p} \leq \frac{c\log(n)(1+\epsilon')^i(1+\epsilon'/2)}{(1+\epsilon')^{i+1}} = \frac{c\log(n)(1+\epsilon'/2)}{1+\epsilon'},$$

and thus, by a Chernoff bound,

$$\mathbb{P}[|S| \geq c\log n] \leq \exp\left(-\frac{(\frac{1+\epsilon'}{1+\epsilon'/2})^2}{2+\frac{1+\epsilon'}{1+\epsilon'/2}}\cdot\frac{c\log(n)(1+\epsilon'/2)}{1+\epsilon'}\right) = \exp\left(-\frac{\frac{1+\epsilon'}{1+\epsilon'/2}c\log(n)}{2+\frac{1+\epsilon'}{1+\epsilon'/2}}\right)$$

$$= \exp\left(-\frac{(1+\epsilon')c\log(n)}{3+2\epsilon'}\right) \leq \exp\left(-\frac{c\log(n)}{3}\right) \leq n^{-3},$$

for $c \geq 21$. Thus, by the union bound, the probability that $E$ occurs is at most $n^{-2}$.

We assume from now on that $E$ does not occur. Let $F_i$ be the event that at the end of iteration $j_i$, the algorithm is in phase $i+1$. We prove now by induction that all $F_i$ occur with high probability. Consider first $F_0$. Conditioned on $\neg E$, the algorithm is in phase 0 or 1 after iteration $j_0$. We argue that with high probability, the algorithm is in phase 1 after iteration $j_0$. Suppose that the algorithm is in phase 0 in the beginning of iteration $j_0$. Then, $\mathbb{E}[S] = \frac{n_{d,j_0}}{p} = n_{d,j_0} = c\log n(1+\epsilon'/2)$. Thus, by a Chernoff bound,

$$\mathbb{P}[|S| \leq c\log n] \leq \exp\left(-c\log n(1+\epsilon'/2)\left(\frac{\epsilon'}{2+\epsilon'}\right)^2\right) = \exp\left(-c\log n\frac{\epsilon'^2}{4+2\epsilon'}\right) \leq n^{-2},$$

for $c \geq \frac{28}{\epsilon'^2}$, and hence, if the algorithm was in phase 0 at the beginning of iteration $j_0$, then, with high probability, the transition to phase 1 would occur.

Assume now that both $\neg E$ and $F_i$ hold. Then, the algorithm is in phase $i+1$ or $i+2$ at the end of iteration $j_{i+1}$. Suppose we are in phase $i+1$ at the beginning of iteration $j_{i+1}$. Then, $\mathbb{E}[S] = \frac{n_{j_0,d}}{p} = n_{j_0,d} = c\log n(1+\epsilon'/2)$, and by the same Chernoff bound as above, the transition to phase $i+2$ would take place with high probability, which implies that $F_{i+1}$ holds.

Let $j_{\max}$ be the largest $j$ such that $c\log n(1+\epsilon'/2)(1+\epsilon')^j \leq n_d$. As proved above, when the algorithm terminates, then the output $m$ is either $c\log n(1+\epsilon')^{j_{\max}}$ or $c\log n(1+\epsilon')^{j_{\max}+1}$ with high probability. Suppose first that the output is $m = c\log n(1+\epsilon')^{j_{\max}}$. Since $m(1+\epsilon'/2)(1+\epsilon') \geq n_d$, the algorithm computes a $(1+\epsilon'/2)(1+\epsilon') \leq (1+2\epsilon')$-approximation. Suppose now that the output is $m = c\log n(1+\epsilon')^{j_{\max}+1}$. Since $m(1+\epsilon'/2)/(1+\epsilon') \leq n_d$, we equally obtain a $(1+2\epsilon')$-approximation. Since $\epsilon = 2\epsilon'$, the algorithm returns a $(1+\epsilon)$-approximation.

Concerning the space requirements of the algorithm, at most $c\log n$ vertex degrees are stored, which requires $O(\frac{1}{\epsilon^2}\log^2 n)$ bits of space. □