Factorial Lower Bounds for (Almost) Random Order Streams

Ashish Chiplunkar
IIT Delhi

John Kallaugher
Sandia National Labs

Michael Kapralov
EPFL

Eric Price
UT Austin

Abstract

In this paper we introduce and study the StreamingCycles problem, a random order streaming version of the Boolean Hidden Hypermatching problem that has been instrumental in streaming lower bounds over the past decade. In this problem the edges of a graph G, comprising n/ℓ disjoint length-ℓ cycles on n vertices, are partitioned randomly among n players. Every edge is annotated with an independent uniformly random bit, and the players' task is to output, for some cycle in G, the sum (modulo 2) of the bits on its edges, after one round of sequential communication.

Our main result is an $\ell^{\Omega(\ell)}$ lower bound on the communication complexity of StreamingCycles, which is tight up to constant factors in the exponent. Applications of our lower bound for StreamingCycles include an essentially tight lower bound for component collection in (almost) random order graph streams, making progress towards a conjecture of Peng and Sohler [SODA'18] and the first exponential space lower bounds for random walk generation.

Contents

1 Introduction

The streaming model of computation has been a core model for small space algorithms that process large datasets since the foundational paper of [AMS96] showed how to (approximately) compute basic statistics of large datasets using space only polylogarithmic in the size of the input. While many such stream statistics are tractable to calculate in this model—including frequency moments, number of distinct elements, and heavy hitters—for graphs of n edges even some basic problems are known to require space $\Omega(n)$ in streams in which the edge arrival order is chosen adversarially. At the same time, a recent line of work on testing graph properties [KKS14, CJMM17, MMPS17, PS18, KMNT20, CFPS20] shows that when the edges of the input graph are presented in a uniformly random order, one can approximate many fundamental graph properties (e.g., matching size, number of connected components, constant query testable properties in bounded degree graphs) in polylogarithmic or even constant space.

These algorithms make use of small space graph exploration primitives, including:

Component collection. Given a graph $G = (V, E)$ presented as a random order stream and a budget k, collect the connected components of a representative sample of vertices of G assuming that many vertices in G belong to components of size bounded by k[1]. The work of [PS18] designs a component collection primitive that uses $k^{O(k^3)}$ space, and then uses it to obtain an additive εn-approximation to the number of connected components in G.

Random walk generation. Given a graph $G = (V, E)$ presented as a random order stream, a target walk length k and a budget s, generate a sample of (close to) independent random walks of length k from s vertices in G selected uniformly at random. The work of [KKP22] designs a primitive that outputs such walks (with constant TVD distance to the desired distribution, say), using space $2^{O(k^2)}s$.

k-disc estimation. Given a bounded degree graph $G = (V, E)$ presented as a random order stream, an integer k, estimate k-disk frequencies[2]. The work of [MMPS17] designs a primitive for estimating k-disc frequencies and then uses it to show that any constant query testable property of bounded degree graphs is random order streamable.

The above primitives perform depth-k exploration in random order graph streams using space exponential in k. Our work is motivated by the natural question:

Does depth-k exploration in random order streams require space exponential in k?

This question was originally raised by [PS18], who wrote

[1] We note that our definition of component collection here is not formal by design. This is because our lower bound applies to a very weak formal version of component collection, in which one is promised that a constant fraction of vertices of the input graph belong to components of size at most k, and the task is to output one such vertex (see Definition 4.2 in Section 4). Our algorithmic results, on the other hand, solve a stronger version of the problem, that of outputting a 'representative sample' of vertices of the graph together with components that they belong to (as long as those are of size bounded by k), in any graph. In particular, our algorithmic primitive naturally leads to an algorithm that additively approximates the number of connected components in the input graph, as in [PS18]—see Section 5.

[2] A k-disc is the subgraph induced by vertices at shortest path distance at most k from a given vertex, and the set of k-disc frequencies corresponds to the numbers of occurrences of all such graphs, up to isomorphism.

...it will also be interesting to obtain lower bounds for random order streams. It seems to be plausible to conjecture that approximating the number of connected components requires space exponential in $1/\varepsilon$. It would be nice to have lower bounds that confirm this conjecture.

In this paper we make progress towards this conjecture, showing that this dependence is indeed necessary, at least in graph streams that admit some amount of correlation. Our lower bound is based on a new communication problem that we refer to as the StreamingCycles problem, a relative of the well-studied Boolean Hidden Hypermatching problem. We introduce the StreamingCycles problem next, then give reductions from component collection and random walk generation. Our reduction generates instances of the component collection and random walk generation problem that are not quite random order streams, but rather allow for small *batches* of edges as opposed to edges themselves to arrive in a random order. We argue that this is in fact a very natural *robust* analog of the idealized random order streaming model, and show that corresponding random order streaming algorithms extend to the batch random order setting. Finally, we give an overview of our lower bound, which is the main technical contribution of the paper.

The StreamingCycles problem. Our main technical contribution is a tight lower bound for the StreamingCycles problem, which we now define. In an instance of StreamingCycles(n, ℓ) a graph $G = (V, E)$ made up of n/ℓ length-ℓ cycles is received as a stream of edges e with bit labels x_e. There are n players, indexed by the edges of G. Upon arrival of an edge e the corresponding bit label x_e is given to the corresponding player as private input, together with a message from the previous player. The edges e are posted on a common board as the edges arrive. The last player must return a vertex $v \in V$ and the *parity* of the cycle $C \subseteq E$ containing v, i.e. $\sum_{e \in C} x_e$. We consider a distributional version of the problem, in which the bits x_e are chosen independently and uniformly at random, and the ordering of the edges in the stream is uniformly random.

A naïve protocol for the StreamingCycles problem is for the players to track the connected component of a vertex $u \in G$. This strategy succeeds if and only if the edges of the component arrive "in order", which happens with probability $\ell^{-\Theta(\ell)}$. Thus, it suffices to track $\ell^{O(\ell)}$ vertices, which results in an $\ell^{O(\ell)}$ communication per player protocol. Our main result shows that this is essentially best possible:

Theorem 1.1 (Main result; informal version of Theorem 4.1)**.** *Any protocol for the* StreamingCycles(n, ℓ) problem that succeeds with $2/3$ probability requires $\min(\ell^{\Omega(\ell)}, n^{0.99})$ bits of communication from some player.

Relation to the Boolean Hidden Hypermatching problem. We note that this is related to the search version of the Boolean Hidden Hypermatching problem, in which a vector $x \in \{0,1\}^n$ is given to Alice, who sends a single message to Bob. Bob, in addition to the message from Alice, is given a perfect hypermatching with hyperedges of size ℓ and must output the parity of x on one of the hyperedges. The Boolean Hidden Hypermatching problem admits a protocol with $O(n^{1-1/\ell})$ communication (Alice simply sends Bob the values of x on a uniformly random subset of coordinates of size $n^{1-1/\ell}$), and this bound is tight. Note that in the StreamingCycles problem the bits of x are associated with edges in the cycles, and therefore every cycle can naturally be associated with a hyperedge in Boolean Hidden Hypermatching problem. In contrast to the Boolean Hidden Hypermatching problem, in which the bits are presented first and then the hyperedges are revealed, in the StreamingCycles problem the bits and the identities of the hyperedges are

gradually revealed to the algorithm. Similarly to the Boolean Hidden Hypermatching problem, a 'sampling' protocol turns out to be nearly optimal. The naïve protocol mentioned above, and considered in more detail in Section 3, solves StreamingCycles(n, ℓ) using $O(\ell!)$ samples, and we prove a nearly matching lower bound of $\min\{\ell^{\Omega(\ell)}, n^{0.99}\}$ bits.

Applications to component collection and random walk generation. We now give a natural way for the players to produce a graph stream based on their inputs to the communication game:

- Define the vertex set $V' = V \times \{0, 1\}$.
- On receiving the t^{th} edge (uv, b_{uv}), insert edges $(u, 0)(v, b_{uv})$ and $(u, 1)(v, \overline{b_{uv}})$ into the stream.

In other words, every edge of the graph G in the StreamingCycles problem becomes a pair of edges in G'. The order in which edges of G' are presented is not quite random as *pairs of edges* as opposed to individual edges arrive in a uniformly random order.

We refer to such streams, in which *batches* of edges arrive uniformly at random in the stream as opposed to edges themselves, as *(hidden-)batch* random order streams. Note that the reduction above generates a stream with batches of size two, corresponding to the pairs of edges arriving at the same time. We argue in Section 3.1 below that the batched model, in which arrival times of edges could be correlated, but the correlations are restricted by bounded size batches, is a very natural *robust* analog of the idealized uniformly random streaming model. In particular, we show that, surprisingly, some existing random order streaming algorithms for estimating graph properties are quite robust, and can be made to work even when the structure of the batches is not known to the algorithm, i.e. in the *hidden-batch* random order model.

Using the reduction above together with Theorem 1.1, we get

Theorem 1.2 (Component collection lower bound; informal version of Theorem 4.3)**.** *Component collection requires $k^{\Omega(k)}$ bits of space in hidden-batch random order streams.*

Proof. If the component collection algorithm returns a vertex (v, b) together with a component of size ℓ that contains (v, b), return v and parity $= 0$. If it returns (v, b) and a component of size 2ℓ containing (v, b), return v and parity $= 1$. Otherwise fail. □

The bound provided by Theorem 1.2 is **tight up to constant factors in the exponent**. We give an algorithm with $k^{O(k)}$ space complexity in Section 5:

Theorem 1.3 (Component collection upper bound; informal version of Theorem 5.2)**.** *There exists a component collection algorithm in (hidden-batch) random order streams with space complexity $k^{O(k)}$ (words).*

Similarly to the work of [PS18], our component collection algorithm can be used to estimate the number of connected components to additive precision εn. The space complexity of our estimation algorithm is $(1/\varepsilon)^{O(1/\varepsilon)}$, similarly improving upon on [PS18]. The details are provided in Section 5.

Similarly, we obtain exponential lower bounds for the random walk generation problem:

Theorem 1.4 (Random walk generation lower bound; informal version of Theorem 4.7)**.** *Generation of a random walk of length k started at any vertex in a graph given as a hidden-batch random order stream requires $k^{\Omega(\sqrt{k})}$ space. Generation of $C \cdot 4^k$ random walks for a sufficiently large constant $C > 0$ requires $k^{\Omega(k)}$ space.*

Proof. For the first lower bound, let $\ell = \sqrt{k/C}$ for a sufficiently large absolute constant C, so that $k = C\ell^2$. Generate a walk of length k with precision $\varepsilon = 1/10$ in total variation distance. The walk loops around the cycle that it starts in with probability at least $2/3$. Let (v, b) denote the starting vertex. If the cycle is of length ℓ, output v and parity $= 0$. If the cycle is of length 2ℓ, output v and parity $= 1$. Thus, random walk generation requires at least $\ell^{\Omega(\ell)} = k^{\Omega(\sqrt{k})}$ space.

For the second bound, let $\ell = k/2$ and run $C4^k = C2^{2\ell}$ random walks of length k started at uniformly random vertices, with precision $1/10$ in total variation distance (for the joint distribution), for a sufficiently large constant $C > 0$. With probability at least $2/3$ at least one of the walks will loop around the cycle that it started in. Let (v, b) denote the starting vertex. If the cycle is of length ℓ, output v and parity $= 0$. If the cycle is of length 2ℓ, output v and parity $= 1$. □

Boolean Fourier Analysis for Many-Player Games Our lower bound for StreamingCycles is based on the techniques of *Boolean Fourier analysis*. The application of these techniques to one-way communication complexity goes back to [GKK+07], but previous applications have either involved two players or at most a small number relative to the size of the input, meaning that they can afford to lose factors polynomial or even exponential in the player count. Our application involves n players for an $\widetilde{O}(n)$-sized input, requiring a careful consideration of how the Fourier coefficients associated with the players' messages evolve as each player passes to the next. We give an overview of these techniques in Sections 2 and 3.

1.1 Related work

The random order streaming model has seen a lot of attention recently. Besides the aforementioned work of [PS18] that gives small space algorithms for component counting, small space approximations to matching size have been given in [KKS14, CJMM17, MMPS17, KMNT20] (naturally, the problem has also attracted significant attention in adversarial streams, but the space complexity of known algorithms in this model is significantly higher than in random order streams [EHL+15, BS15, AKL17, MV18, BGM+19, MV16, CCE+16, EHM16]). The work of [MMPS17] shows that constant query testable graph properties can be tested in constant space in random order streams in bounded degree graphs.

The Fourier-analytic approach to proving communication complexity lower bounds pioneered by [GKK+07] has been instrumental in lower bounds for many graph problems, including cycle counting [VY11], estimating MAX-CUT value [KKS15, KKSV17, KK19] more general CSPs [GVV17, GT19, CGV20], and subgraph counting [KKP18]. Communication problems inspired by the Boolean hidden matching problem (and therefore related to the StreamingCycles problem that forms the basis of our lower bound) have also been recently used to obtain lower bounds for multipass algorithms for several fundamental graph streaming problems [AKL17, AKSY20, AN21]. Lower bounds for statistical estimation problems (e.g., distinct elements, frequency moments and quantile estimation) in random order streams were given in [CCM08, CJP08].

2 Warm-up: Boolean Hidden Hypermatching with interleaving

Let $\mathbf{X} \in \{0,1\}^n$ be uniformly random, and revealed one bit at a time to our algorithm. Let our algorithm's state at time t be $\mathbf{M}_t$, which is at most c bits long. The Fourier-analytic lower bound approach studies quantities corresponding to the following question: what is known about the *parity* of each set of bits at time t? For the indicator $z \in \{0,1\}^t$ of any subset of bits that arrive before time t, define

$$\widetilde{\mathbf{F}}_t(z) \coloneqq \mathop{\mathbb{E}}_{x|\mathbf{M}_t}[(-1)^{z \cdot x}] \tag{1}$$

which is ± 1 if the corresponding parity is specified by the message and 0 if it is completely unknown. One can think of $\widetilde{\mathbf{F}}_t(z)^2 \in [0,1]$ as an estimate of how well the parity z is remembered at time t. A method that stores c individual bits would have

$$\forall k \le c, \sum_{|z|=k} \widetilde{\mathbf{F}}_t(z)^2 = \binom{c}{k},$$

where $|z|$ denotes the Hamming weight of z, because it remembers exactly each subset of those bits. The foundation of the Fourier-analytic approach is that a similar bound typically holds for *any* protocol that generates c-bit messages $\mathbf{M}_t$:

$$\forall k \le c, \sum_{|z|=k} \widetilde{\mathbf{F}}_t(z)^2 \le \binom{O(c)}{k} \tag{2}$$

with very good probability over $\mathbf{M}_t$. This inequality (Lemma 3 in [GKK+07]) is a consequence of the hypercontractive inequality (see Lemma 3.4 in [KKL88]).

In Boolean Hidden Hypermatching, one first receives the bits x and then receives $\Theta(n/\ell)$ "important" sets $z^{(i)}$, each of size ℓ. Since the sets are uniform and independent of the message $\mathbf{M}_t$ (and so, of $\widetilde{\mathbf{F}}_t$), the expected amount known about them is

$$\mathop{\mathbb{E}}_{z^{(i)}}\left[\sum_{i \in [n/\ell]} \widetilde{\mathbf{F}}_t(z^{(i)})^2\right] = (n/\ell)\frac{1}{\binom{n}{\ell}} \sum_{|z|=\ell} \widetilde{\mathbf{F}}_t(z)^2 \le (n/\ell)\frac{\binom{O(c)}{\ell}}{\binom{n}{\ell}} = (n/\ell)\left(\frac{O(c)}{n}\right)^{\ell}. \tag{3}$$

If $c \ll n^{1-1/\ell}$, this is $o(1)$ so the algorithm probably does not remember any of the important parities.

The challenge we face in adapting this approach is that our important sets (the components of the graph) are revealed over time, interleaved with the bits of x rather than at the end.

To see how this can be an issue, consider a two-stage version of Boolean Hidden Hypermatching: the first $n/2$ bits of x are given, then at time $s = n/2$ we receive $z^{(i)}_{\le s}$ (elements $z^{(i)}$ with indices at most s) for each i, then the rest of x, and finally at time $t = n$ we receive the rest of the important indices $z^{(i)}_{[s+1:t]}$ (elements $z^{(i)}$ with indices between $s+1$ and t). For simplicity, suppose $\left|z^{(i)}_{\le s}\right| = \left|z^{(i)}_{[s+1:t]}\right| = \ell/2$ always. The algorithm that stores a random subset of bits still needs $c \gtrsim n^{1-1/\ell}$. Solving either half of the stream (determining the parity of one of the half-sets $z^{(i)}_{\le s}$, $z^{(i)}_{[s+1:t]}$) requires only $n^{1-2/\ell}$ space in general, but how can we get a tight $n^{1-1/\ell}$ bound?

The problem is that (2) does not give strong enough control over the higher-order moments to show (3). The sets $z^{(i)}$ at the end are no longer independent of $\widetilde{\mathbf{F}}_t$, because the algorithm's behavior

in the second half can depend on the $z_{\le s}$. One could instead apply (2) to each half of the stream and take the product, getting

$$\sum_z \widetilde{\mathbf{F}}_s(z_{\le s})^2 \widetilde{\mathbf{F}}_t(0^s z_{[s+1:t]})^2 = \left(\sum_{|z_{\le s}|=\ell/2} \widetilde{\mathbf{F}}_s(z_{\le s})^2\right)\left(\sum_{|z_{[s+1:t]}|=\ell/2} \widetilde{\mathbf{F}}_t(0^s z_{[s+1:t]})^2\right) \le \binom{O(c)}{\ell/2}^2$$

and so, on average over z,

$$\sum_{i\in[n/\ell]} \widetilde{\mathbf{F}}_s(z^{(i)}_{\le s})^2 \widetilde{\mathbf{F}}_t(0^s z^{(i)}_{[s+1:t]})^2 \le (n/\ell)\left(\frac{O(c)}{n}\right)^\ell.$$

For algorithms that store individual bits this implies (3), since in that case

$$\widetilde{\mathbf{F}}_t(z) = \widetilde{\mathbf{F}}_t(z_{\le s}0^{t-s})\widetilde{\mathbf{F}}_t(0^s z_{[s+1:t]})$$

and

$$\widetilde{\mathbf{F}}_t(z_{\le s}0^{t-s})^2 \le \widetilde{\mathbf{F}}_s(z_{\le s})^2.$$

However, for general algorithms,

$$\widetilde{\mathbf{F}}_t(z) \ne \widetilde{\mathbf{F}}_t(z_{\le s}0^{t-s})\widetilde{\mathbf{F}}_t(0^s z_{[s+1:t]}).$$

To solve this, we need to relate $\widetilde{\mathbf{F}}_t(z)$ to bounds involving $z_{\le s}$ and $z_{[s+1:t]}$ individually. We define

$$\widetilde{\mathbf{r}}_{s,t}(z_{[s+1:t]}) \coloneqq \mathop{\mathbb{E}}_{x|\mathbf{M}_s,\mathbf{M}_t,\mathbf{B}_t}\left[(-1)^{z_{[s+1:t]}\cdot x_{[}s+1:t]}\right]$$

as a "double-ended" version of (1): it asks about the knowledge of H given the states before and after H arrives, as well as the "board" $\mathbf{B}_t$ at time t (which is the information about important sets revealed by time t, namely the $z^{(i)}_{\le s}$). Since x is independent of $\mathbf{B}_t$, this is specified by $2c$ bits ($\mathbf{M}_s$ and $\mathbf{M}_t$), so it also satisfies (2). Our key observation, Lemma 3.5, is that

$$\widetilde{\mathbf{F}}_t(z) = \mathop{\mathbb{E}}_{\mathbf{M}_s|\mathbf{M}_t,\mathbf{B}_t}\left[\widetilde{\mathbf{F}}_s(z_{\le s})\widetilde{\mathbf{r}}_{s,t}(z_{[s+1:t]})\right]. \tag{4}$$

This lets us relate $\widetilde{\mathbf{F}}_t(z)$ to $\widetilde{\mathbf{F}}_s(z_{\le s})$ and $\widetilde{\mathbf{r}}_{s,t}(z_{[s+1:t]})$, each of which are bounded by (2).

Specifically, for any fixed index i, the average amount that is remembered about $z^{(i)}$ is:

$$\begin{aligned}
\mathop{\mathbb{E}}_{z^{(i)},\mathbf{M}_s,\mathbf{M}_t}\left[\widetilde{\mathbf{F}}_t(z^{(i)})^2\right] &\le \mathop{\mathbb{E}}_{\mathbf{M}_s,\mathbf{M}_t}\left[\mathop{\mathbb{E}}_{z^{(i)}}\left[\widetilde{\mathbf{F}}_s(z^{(i)}_{\le s})^2\widetilde{\mathbf{r}}_{s,t}(z^{(i)}_{[s+1:t]})^2\right]\right] \\
&= \mathop{\mathbb{E}}_{\mathbf{M}_s,\mathbf{M}_t}\left[\mathop{\mathbb{E}}_{z^{(i)}_{\le s}}\left[\widetilde{\mathbf{F}}_s(z^{(i)}_{\le s})^2 \mathop{\mathbb{E}}_{z^{(i)}_{[s+1:t]}}\left[\widetilde{\mathbf{r}}_{s,t}(z^{(i)}_{[s+1:t]})^2\right]\right]\right] \\
&\le \frac{\binom{O(c)}{\ell/2}}{\binom{n/2}{\ell/2}}\cdot\frac{\binom{O(2c)}{\ell/2}}{\binom{n/2}{\ell/2}} = \left(\frac{O(c)}{n}\right)^\ell.
\end{aligned}$$

Thus, for the two-stage version of Boolean Hidden Hypermatching, we need $c \gtrsim n^{1-1/\ell}$ to remember one of the n/ℓ important sets on average.

These equations, (4) and (2), form the Fourier-analytic basis of our lower bound. The rest of the challenge for our setting comes from random order streams having much more complicated combinatorics than two-stage hypermatching, spread across n stages. As edges arrive, components appear, extend, and merge to eventually form the final cycles.

(a) Single-seed cycle (b) Multi-seed cycle

Figure 1: Illustration of two possible edge arrival orders. Lighter edges arrive later.

3 Technical Overview

A basic "sampling" protocol. Suppose we only permit ourselves to remember one parity (so using one bit of space, in addition to whatever space we need to know which parity this is). We want to eventually learn the parity of one cycle in the stream, and so the natural strategy is as follows:

1. Arbitrarily choose some edge to start with, and record its parity.
2. Whenever we see a new edge that is incident to the parity we are storing, add that edge to the parity, and hope our parity eventually grows to encompass an entire cycle.

This strategy will succeed with probability $\ell^{-\Theta(\ell)}$, as it only works if no edge of the cycle arrives before a path to it from the first edge has already arrived (we refer to such a cycle as a "single-seed" cycle—see Fig. 1a for an illustration). So we would have to repeat this process $\ell^{\Theta(\ell)}$ times in order to achieve a constant probability of success.

Can we hope to do better by not considering the parities independently? Suppose we maintain the parities of c paths at a time, which may merge with each other as we process the stream. We now have some chance of finding "multi-seed" cycles, i.e. cycles in which several disjoint paths arrive before eventually being merged by later edge arrivals—see Fig. 1b for an illustration. If we happen to have remembered the parity of each of the components that eventually merged into a given multi-seed cycle, we will find the parity of the cycle. The chance that any k of them are from the same cycle is $\sim\binom{c}{k}(\ell/n)^{k-1}$, and the chance of any given cycle having only k seeds is $(\ell/k)^{-\Theta(\ell)}$, as it requires k paths of average length ℓ/k to arrive in order. Until c is $n^{1-\Theta(1/\ell)}$, the probability of finding the parity of a cycle will therefore by dominated by the single-seed case.

However, so far we have assumed we can only store individual parities. To extend these arguments to algorithms that maintain arbitrary state, we make use of the tools of *Boolean Fourier analysis*.

Fourier-analytic lower bound for StreamingCycles. We construct a hard instance for the problem in which the bit labels $\mathbf{X}$ for the edges are chosen uniformly at random, and in order to simplify the analysis we allow the algorithm to remember which edges it has seen for free (although not the bit labels). We say that these edges are posted on the "board" $\mathbf{B}$. The state of the board at time t, i.e. after receiving t edges, is denoted by $\mathbf{B}_t$.

For any subset $z \in \{0,1\}^t$ of the edges that have arrived so far (given by the appropriate bit

mask), we can associate the expectation of the parity of z given $\mathbf{M}_t$ with the *normalized Fourier coefficient*

$$\widetilde{\mathbf{F}}_t(z) \coloneqq \mathop{\mathbb{E}}_{x|\mathbf{M}_t}[(-1)^{z\cdot x}].$$

Note that if the algorithm returns v, and C is the cycle containing v (written as an element of $\{0,1\}^n$), the algorithm's best guess for the parity of C will be 1 if $\widetilde{\mathbf{F}}_n(C) > 0$ and -1 otherwise. Moreover, the probability that this guess will be correct is $\frac{1+|\widetilde{\mathbf{F}}_n(C)|}{2}$. Therefore, for a lower bound, it will suffice to prove that with good probability

$$\left|\widetilde{\mathbf{F}}_n(C)\right| = \mathrm{o}(1).$$

Fourier mass on collections of component types. Writing $\mathbb{Z}_+^{\{\}}$ for the set of multisets of integers, for $\beta \in \mathbb{Z}_+^{\{\}}$ and a $z \in \{0,1\}^t$ we define

$$z \sim_t \beta$$

to be true iff z corresponds to a set of edges which contains $\beta[a]$ components (i.e., paths or cycles) of length a for each a (here for $\beta \in \mathbb{Z}_+^{\{\}}$ and an integer a we write $\beta[a]$ to denote the number of occurrences of a in β). Our main object of study in the lower bound proof is

$$\mathbf{H}_\beta^t \coloneqq \sum_{z\in\{0,1\}^t, z\sim_t\beta} \widetilde{\mathbf{F}}_t(z)^2,$$

which can be viewed as the amount of certainty that the algorithm has about parities of unions of components whose sizes are prescribed by β.

In order to build some intuition, we consider our prototypical sampling protocol from the start of Section 3. Let c be the number of edges sampled at the beginning and for every $j \geq 1$ let Y_j^t denote the number of components of size j (i.e., with j edges) that the sampling algorithm was able to construct at time t. Then for every multiset β and every t one simply has

$$\mathbf{H}_\beta^t = \prod_a \binom{Y_a^t}{\beta[a]}.$$

and

$$\mathbf{H}_{\{1\}}^c = c,$$

where the last equality holds because the sampling algorithm grows components out of the first c arriving edges, and for every j we have that $\mathbf{H}_{\{j\}}^t$ is the number of components of size j that the algorithm knows the parity of at time t. In order to prove that the sampling protocol does not succeed, we need to prove that $Y_\ell^n = 0$, and for general protocols we need to prove

Lemma 3.1. *For all $\varepsilon > 0$, there is a $D > 0$ depending only on ε such that, if $c < \min(\ell^{\ell/D}, n^{1-\varepsilon})$ and $D < \ell < D^{-1}\log n$,*

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\mathbf{H}_{\{\ell\}}^n\right] \leq \varepsilon.$$

In order to establish Lemma 3.1, we bound the expected evolution of $\mathbb{E}\left[\mathbf{H}_\beta^t\right]$ as a function of t. In what follows we first analyze the evolution of $\mathbf{H}_\beta^t$ for the simple "sampling" protocol, and then present the main ideas of our analysis.

Evolution of Fourier coefficients of the "sampling" protocol. Fix a component of size j. At time t the probability that it gets extended is about $\frac{2}{n-t}$, and therefore

$$\mathbb{E}[Y_j^t|\mathbf{B}_{t-1},\mathbf{F}_{t-1}] \approx Y_j^{t-1} + \frac{2}{n-t}\cdot Y_{j-1}^{t-1} + (\text{contribution from merges of smaller components}).$$

In order to derive the asymptotics of Y_j^t, we first ignore the contribution of merges, and later verify that they do not affect the result significantly. In particular, ignoring the contribution of merges, we get

$$\mathbb{E}[Y_j^t|\mathbf{B}_{t-1},\mathbf{F}_{t-1}] \approx Y_j^{t-1} + \frac{2}{n-t}\cdot Y_{j-1}^{t-1}.$$

We assume for intuition that $t \le n/2$, i.e .we are only looking at the first half of the stream. Since the initial conditions are (essentially) $Y_1^1 = c$ and $Y_j^1 = 0$ for $j > 1$, because the algorithm can remember c single edges at the beginning of the stream, and no larger components (since they typically do not form at the very beginning of the stream). This now yields that

$$Y_j^t \le c\cdot 4^{j-1}(t/n)^{j-1}/(j-1)!$$

for $t \le n/2$ and all $j \ge 1$. This is because $Y_j^1 = c$ as required, and for $j \ge 2$

$$\begin{aligned}
\mathbb{E}[Y_j^t] &\le \sum_{s=1}^{t-1}\frac{2}{n-s}Y_{j-1}^s \\
&\le (c\cdot 4^{j-1}/(j-2)!)\frac{1}{n}\sum_{s=1}^{t-1}(t/n)^{j-2} \\
&\approx c\cdot 4^{j-1}/(j-2)!\cdot\int_0^{t/n}x^{j-2}dx \\
&= c\cdot 4^{j-1}(t/n)^{j-1}/(j-1)!
\end{aligned} \tag{5}$$

This in particular implies that $Y_\ell^{t/2} \ll 1$ if $c = \ell^{o(\ell)}$, and in general that for the sampling protocol we have, at least for $t \le n/2$,

$$\begin{aligned}
\mathbb{E}[\mathbf{H}_\beta^t] &= \mathbb{E}\left[\prod_{j\in\beta}Y_j^t\right] \\
&\approx \prod_{j\in\beta} c\cdot 4^{j-1}(t/n)^{j-1}/(j-1)! \\
&\lesssim \left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot\left(\frac{t}{n}\right)^{|\beta|_*-|\beta|}\cdot c^{|\beta|},
\end{aligned}$$

where Q is an absolute constant, $|\beta|_* = \sum_{i\in\beta} i$ and $|\beta|$ is the number of elements in the multiset β (counting multiplicities).

Evolution of Fourier coefficients of a general protocol. The outline of the simple "sampling" protocol above provides a good model for our general proof. Specifically, in Lemma 4.37 (see Section 4.8) we show that there exists a constant $Q > 0$ such that for (almost) all β and t (the

near-end of the stream and going from $\beta = \{\ell - 1\}$ to $\beta = \{\ell\}$ require some special treatment) one has

$$\mathbb{E}\left[\mathbf{H}_\beta^t\right] \lesssim \left(\prod_{j \in \beta} \frac{1}{j!}\right) \cdot Q^{|\beta|_*} \cdot \left(\frac{t}{n}\right)^{|\beta|_* - \nu(\beta)} \cdot c^{|\beta|}, \tag{6}$$

where $|\beta|_* = \sum_{i \in \beta} i$ and $\nu(\beta) = \sum_{i \in \beta} \lceil \frac{i}{2} \rceil$. Lemma 3.1 then follows by essentially summing the above bound over all component types (the proof is presented in Section 4.8).

The result of Lemma 3.1 can then be seen to imply that the chance of successfully guessing the parity of any cycle is o(1) whenever $c = \ell^{o(\ell)}$, and specifically that the StreamingCycles problem requires $\ell^{\Omega(\ell)}$ space.

To establish (6), we bound the (expected) evolution of $\mathbf{H}_\beta^t$ as a function of $t \in [n]$. Specifically, we show in Section 4.7 that for every $t \in [n]$ the expectation of $\mathbf{H}_\beta^t$ can be upper bounded in terms of expectations of $\mathbf{H}_\alpha^s$ for $s < t$ and α corresponding to "subsets" of β (see Section 4.4 for the formal definitions). This is a natural extension of our analysis of the "sampling" protocol above. In full generality, however, this requires showing that if the algorithm has limited information about parities of collections of type α at time s (i.e., $\mathbf{H}_\alpha^s$ is small), then it is unlikely to know too much about collections of type β obtained as a result of merging several components in α or extending them by edges arrived between s and t.

Crucially, the probability that a collection of components of type α grows into a collection of components of type β at any given time depend only on the *collection type* (i.e., the multisets α and β). Specifically, for $s \in [n]$, a pair of collection types $\alpha, \beta \in \mathbb{Z}_+^{\{\}}$ such that $\alpha[1] = \beta[1]$ (the number of single edge components in α and β is the same) and a realization B_s of the board $\mathbf{B}_s$ at time s we write

$$p_s(\alpha, \beta, B_s) = \Pr_{\mathbf{B}_{s+1}}\left[z \cdot 1 \sim_{s+1} \beta | \mathbf{B}_s = B_s\right]$$

for any $z \in \{0,1\}^s$ such that $z \sim_s \alpha$ to denote the probability that a collection of type α at time s becomes a collection of type β at time $s+1$ through one of the following "growth events":

Extension An edge arrives at time $s+1$ that is incident to exactly one component in the collection.

Merge An edge arrives at time $s+1$ that is incident to two components in the collection.

We will use

Lemma 3.2 (Informal version of Lemma 4.18)**.** *With high probability over the board state $\mathbf{B}_s$, for s not too close to n one has for (almost) every α, β*

$$p_s(\alpha, \beta, \mathbf{B}_s) \leq \begin{cases} \frac{O(\alpha[a])}{n} & \text{if } \alpha \to \beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a] \cdot \alpha[b])}{(n-s)^2} & \text{if } \alpha \to \beta \text{ is a merge of paths of size } a \text{ and } b. \end{cases}$$

We will also need

Definition 3.3 (Down set of $\beta \in \mathbb{Z}_+^{\{\}}$)**.** *For $\alpha, \beta \in \mathbb{Z}_+^{\{\}}$ we write $\alpha \in \beta - 1$ if β can be obtained from α by either an* extension *or a* merge *followed by possibly adding an arbitrary number of 1's to α.*

For $\beta \in \mathbb{Z}_+^{\{\}}$ and $\alpha \in \beta - 1$ we write $|\beta - \alpha|$ to denote the number of ones that need to be added to α after a merge or extension to obtain β.

Equipped with the above, and writing T for the set of all edge arrival times, we can state our main bound on the evolution of Fourier coefficients:

Lemma 3.4. *For every $t \in T$ one has for $\beta \in \mathbb{Z}_+^{\{\}}$ that contain at least one component of size more than 1 and have $|\beta|_* \leq \ell - 2$,*

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\beta^t\right] \leq \sum_{s=1}^{t-1} \sum_{\alpha \in \beta - 1} q(|\beta - \alpha|) \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}[\mathbf{H}_\alpha^s \cdot p(\alpha, \beta, \mathbf{B}_s)]$$

and for β with all components of size 1

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\beta^t\right] \leq q(|\beta|),$$

where

$$q(k) = \begin{cases} \left(\frac{8c}{k}\right)^k & \text{if } k \leq c \\ 2^c & \text{otherwise.} \end{cases}$$

The function $q(k)$ comes from hypercontractivity (as in (2), [KKL88]), and bounds the total amount of Fourier mass a c-bit message can place on size-k parities; when $\alpha \to \beta$, $q(|\beta - \alpha|)$ appears because the term involves remembering $|\beta - \alpha|$ of the isolated edges that arrive between s and t.

The proof of Lemma 3.4 is based on a function $\mathbf{r}$ that functions similarly to the $\mathbf{r}_{s,t}$ used in the Section 2 warm-up. Lemma 3.4 allows us to bound the Fourier mass on various collections of components as a function of their evolution in the stream. We consider two prototypical examples now.

Example 1: single-seed components. To obtain some intuition for Lemma 3.4, we first consider a simplified setting where $p(\alpha, \beta, \mathbf{B}_s) = 0$ unless $\alpha \to \beta$ is an extension, i.e., we ignore the effect of merges. Without merges, since we eventually care about the single-element set $\{\ell\}$, we only need to track the mass on other single-element sets $\beta = \{a\}$, so $\beta - 1 = \{a - 1\}$. Then

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\beta^t\right] &\leq \sum_{s=1}^{t-1} \sum_{\substack{\alpha \in \beta - 1 \\ \alpha \to \beta \text{ is an extension}}} q(|\beta - \alpha|) \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}[\mathbf{H}_\alpha^s \cdot p(\alpha, \beta, \mathbf{B}_s)] \\
&= \sum_{s=1}^{t-1} q(0) \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}_{\{a-1\}}^s \cdot p(\{a-1\}, \{a\}, \mathbf{B}_s)\right] \\
&\leq \sum_{s=1}^{t-1} 1 \cdot \frac{O(1)}{n} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}_{\{a-1\}}^s\right].
\end{aligned}$$

where the last step uses the first bound from Lemma 3.2. One observes that the above recurrence is quite similar to (5) and can be upper bounded similarly by $c\frac{2^{O(a)}}{a!}(t/n)^a$—as needed for (6).

Example 2: multi-seed components. To illustrate the way Lemma 3.4 handles merges, suppose we want to bound $H_{\{4\}}$. There are three paths from $\{4\}$ via down-set relations to our base cases:

$$\{4\} \to \{3\} \to \{2\} \to \{1\}$$
$$\{4\} \to \{3\} \to \{1,1\}$$
$$\{4\} \to \{2,1\} \to \{1\}$$

The first is a series of extensions, so bounded by about $c/a!$ according to Example 1; the other two involve merges, and we show give negligible contribution. We show this for the $\{4\} \to \{2,1\} \to \{1\}$ path here.

The only path to $\beta = \{2,1\}$ is an extension from $\{1\}$, so Lemma 3.4 and Lemma 3.2 show

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}^t_\beta\right] &\le \sum_{s=1}^{t-1} \sum_{\alpha \in \beta - 1} q(|\beta - \alpha|) \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}[\mathbf{H}^s_\alpha \cdot p(\alpha, \beta, \mathbf{B}_s)] \\
&= \sum_{s=1}^{t-1} q(1) \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}^s_{\{1\}} \cdot p(\{1\}, \{2,1\}, \mathbf{B}_s)\right] \\
&\le \sum_{s=1}^{t-1} 8c \cdot \frac{O(1)}{n} \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}^s_{\{1\}}\right] \\
&\lesssim c^2(t/n).
\end{aligned}$$

Then the contribution to $\mathbb{E}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}^t_\beta\right]$ for $\beta = \{4\}$ from the $\{4\} \to \{2,1\} \to \{1\}$ path is at most

$$\begin{aligned}
\sum_{s=1}^{t-1} q(|\{4\} - \{2,1\}|) \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}^s_{\{2,1\}} \cdot p(\{2,1\}, \{4\}, \mathbf{B}_s)\right] &\le \sum_{s=1}^{t-1} 1 \cdot O(c^2(s/n)) \cdot \frac{O(1)}{(n-s)^2} \\
&\lesssim c^2 \sum_{s=1}^{t-1} \frac{1}{(n-s)^2} \\
&\eqsim \frac{c^2}{n-t}
\end{aligned}$$

For almost the entire stream, say $t < n - n^{0.99} < n - \omega(c)$, this term is much less than the $\Theta(c)$ contribution from extensions. The fact that merges are about a c/n factor less likely can be used in general to show that the evolution is ultimately dominated by extensions and establish (6), for $t < n - n^{0.99}$.

Handling the end of the stream. Our bound (6) gives roughly a $\frac{c}{a!}$ bound on $H^t_{\{a\}}$, but only up to time $t = n - n^{0.99}$. By this time, however, probably every single cycle will be missing only $O(1)$ edges, and so have at most $O(1)$ components. Thus each cycle will have an $\Omega(\ell)$-long component that has arrived, and (6) gives a $\frac{c^{O(1)}}{\Omega(\ell)!} \ll \varepsilon/\operatorname{poly}(\ell)$ bound for the Fourier mass of its collection type at t. Summing over the $\operatorname{poly}(\ell)$ possible types leads to Lemma 3.1.

We now discuss a key technical insight that allows us to establish Lemma 3.4. It is the one illustrated in the warm-up example of Section 2.

Key tool in proving Lemma 3.4: decomposition of a typical message. In order to establish Lemma 3.4, we need an approach to expressing the Fourier transform of the typical message $\mathbf{F}_t$ at time t in terms the Fourier transform of the typical messages $\mathbf{F}_s$ for $s < t$. This is achieved by Lemma 3.5 below. Intuitively, this lemma allows us to exploit communication bottlenecks arising at every $s < t$ that preclude various Fourier coefficients from becoming large.

Let $\mathbf{r}(x_{[s+1:t]}; F_s, F_t)$ denote the indicator function for $x_{[s+1:t]}$ taking F_s to F_t. Note that $\mathbf{r}$ is a random function depending only on $\mathbf{B}_t$ (F_s is a function on s bits and F_t on t bits, so its dependence on $\mathbf{B}_t$ is given implicitly by its arguments). The decomposition of typical messages is given by

Lemma 3.5. *For every $s, t \in T$ with $s < t$, and any $z_{\leq t}$, we have:*

$$\widetilde{\mathbf{F}}_t(z_{\leq t}) = \mathop{\mathbb{E}}_{\mathbf{F}_s}\left[\widetilde{\mathbf{F}}_s(z_{\leq s}) \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)\middle|\mathbf{F}_t, \mathbf{B}_t\right].$$

Since $\mathbf{r}$ is a function partitioning its input based on at most $2c$ bits, its normalized Fourier transform $\widetilde{\mathbf{r}}$ is subject to a bound similar to (2), which enables us to establish Lemma 3.4.

3.1 Hidden-Batch Random Order Streams

We introduce a new random order streaming model that allows for (limited) correlations whose structure is unknown to the algorithm (the hidden-batch random order streaming model, Definition 3.6 below). We give new algorithms for estimating local graph structure using small space in this model, and show that existing results in this space translate to our model with only a very mild loss in parameters.

Definition 3.6 (Hidden-batch random order stream model; informal)**.** *In the (b, w)-hidden-batch random order stream model the edges of the input graph $G = (V, E)$ are partitioned adversarially into batches of size bounded by b, after which every batch is presented to the algorithm in a time window of length $w \geq 0$ starting at a uniformly distributed time in the interval $[0, 1]$.*

To motivate this model consider observing, say, a network traffic stream or a stream of friendings in a social network. In each case, there are many events (say, a login attempt, or a group of people meeting each other at a party) that will trigger a bounded number of updates (the back and forward of packets in a login protocol, or people adding friends they met) that might have very complicated temporal correlations with each other, but that occur over a bounded period and are mostly independent of other events being observed in the stream.

To simulate this, we think of the division of observations into events (our batches) being adversarial but limited by a maximum batch size b, while the times of the events are chosen at random but the observations associated with the events are adversarially distributed about the event time, subject to the event duration limit w. Note that this means that observations from multiple events may (and often will be) interleaved—more than one person may be logging onto the same network at the same time and more than one party may be taking place at once.

It is worth stressing that the partitioning of edges into batches is unknown to the algorithm (consequently, we refer to our model as the *hidden* batch model).

Some existing random order streaming algorithm can be readily ported to the hidden-batch random order streaming model.

Theorem 3.7 (Component Collection; informal version of Theorem 5.2)**.** *There is a (b, w)-hidden batch streaming algorithm that, if at least a $\Omega(1)$ fraction of the vertices of G are in components of*

size at most ℓ, returns a vertex in G and the component containing it with probability $9/10$ over its internal randomness and the order of the stream, using $\ell^{O(\ell)}(b+wm)\operatorname{polylog} n$ bits of space.

We show that results of [PS18] on counting connected components in random graph streams can be easily extended to our hidden-batch random order model with only a mild loss in parameters. Specifically, in [PS18] it was shown that the number of connected components $c(G)$ in a graph G can be approximated up to an εn additive term using $(1/\varepsilon)^{O(1/\varepsilon^3)}$ words of space. We show that this can be improved to $(1/\varepsilon)^{O(1/\varepsilon)}$ and that (up to log factors) it can be extended to hidden-batch streaming with only linear loss in the parameters.

Theorem 3.8 (Counting Components; informal version of Theorem 5.1)**.** *For all $\varepsilon \in (0,1)$, there is a (b,w)-hidden batch streaming algorithm that achieves an εn additive approximation to $c(G)$ with $9/10$ probability, using $(1/\varepsilon)^{O(1/\varepsilon)}(b+wm)\operatorname{polylog}(n)$ bits of space.*

We note that the wm term in the space complexity corresponds to the expected number of edges arriving in a given time window of length w. Since the arrival times of edges are adversarially chosen in a window of length w started at the arrival time of the corresponding batch, it is natural to expect the algorithm to store these edges.

Our algorithm above, similarly to the approach of [PS18], proceeds by first sampling a few nodes in the graph uniformly at random, and then constructing connected components incident on those nodes explicitly. Such a sample of component sizes for a few vertices selected uniformly at random from the vertex set of the input graph can then be used to obtain an estimator for the number of connected components. The details are given in Section 5.

4 Lower Bounds

In this section we prove our main result, which is a lower bound for StreamingCycles, even when the edge arrival order and bit labels are chosen uniformly at random.

Theorem 4.1 (StreamingCycles Lower Bound)**.** *For all constants $\varepsilon > 0$, solving the distributional version of the* StreamingCycles(n,ℓ) problem with probability at least $2/3$ requires at least one player to send a message of size at least $\min\left(\ell^{\Omega(\ell)}, n^{1-\varepsilon}\right)$.

We use this to prove a lower bound for the component collection problem.

Definition 4.2 (Component Collection)**.** *In the (β,ℓ) component collection problem, we are given a graph G as a stream of edges, with at least $\beta|V(G)|$ of its vertices in components of size at most ℓ, and we must return a vertex $v \in V(G)$ and the size of the component containing v.*

Our lower bound is given by Theorem 4.3:

Theorem 4.3 (Component Collection Lower Bound)**.** *For all constants $C, \varepsilon > 0$, solving the $(1,\ell)$ component estimation problem in the $(2,0)$-batch random order streaming model with probability at least $2/3$ requires at least $\min\left(\ell^{\Omega(\ell)}, n^{1-\varepsilon}\right)$ space.*

The theorem gives a tight lower bound for the $(1,\ell)$ component collection problem in $(2,0)$-hidden batch streams.

We also prove a lower bound for the random walk generation problem in random streams, which we define formally first. Our definition matches the one in [KKP22].

Definition 4.4 (Pointwise ε-closeness of distributions)**.** *We say that a distribution $p \in \mathbb{R}_+^{\mathcal{U}}$ is ε-close pointwise to a distribution $q \in \mathbb{R}_+^{\mathcal{U}}$ if for every $u \in \mathcal{U}$ one has*

$$p(u) \in [1-\varepsilon, 1+\varepsilon] \cdot q(u).$$

We now define the notion of an ε-approximate sample of a k-step random walk:

Definition 4.5 (ε-approximate sample)**.** *Given $G = (V, E)$ and a vertex $u \in V$ we say that $(X_0, X_1, \ldots, X_k)$ is an* ε-approximate sample *of the k-step random walk started at u if the distribution of $(X_0, X_1, \ldots, X_k)$ is ε-close pointwise to the distribution of the k-step walk started at u (see Definition 4.4).*

Definition 4.6 (Random walk generation)**.** *In the $(k, s, \varepsilon, \delta)$-random walk generation problem one must, given a graph $G = (V, E)$ presented as a stream, generate s independent ε-approximate samples of the walk of length k in G started at a uniformly random vertex, with error bounded by δ in the total variation distance.*

The work of [KKP22] designs a primitive that outputs such walks using space $(1/\varepsilon)^{O(k)} 2^{O(k^2)} s$, with $\delta = 1/10$, say.

Theorem 4.7 (Random Walk Generation Lower Bound)**.** *There exists an absolute constant $C > 1$ such that for sufficiently large $k \geq 1$,* **(1)** *solving the $(k, 1, 1/10, 1/10)$ component estimation problem in the $(2, 0)$-batch random order streaming model requires at least $\min\left(k^{\Omega(\sqrt{k})}, n^{0.99}\right)$ space and* **(2)** *solving the $(k, C4^k, 1/10, 1/10)$ random walk generation problem in the $(2, 0)$-batch random order streaming model requires at least $\min\left(k^{\Omega(k)}, n^{0.99}\right)$ space.*

We start by setting up basic notation in Section 4.1, then define our communication problem, namely the StreamingCycles problem, in Section 4.2. Notation relating the underlying graph in the StreamingCycles problem to the observable graph (the graph that the players see on their common board) together with basic results on this relation is presented in Section 4.3. Definitions relating to properties of collections of components observed on the board, such as the definition of extension and merge events, are presented in Section 4.4. Technical lemmas on the probabilities of these events are presented in Section 4.5. A key lemma on the decomposition of the typical message of a player in the StreamingCycles problem is presented in Section 4.6. The main recurrence on the evolution of Fourier coefficients throughout the stream is obtained in Section 4.7, and solved in Section 4.8, where we prove of our main lemma (Lemma 3.1). We use this to prove Theorem 4.1 in Section 4.9. Finally, the proof of Theorem 4.3 is presented in Section 4.10, and the proof of Theorem 4.7 is presented in Section 4.11.

4.1 Notation

We use the notation $[a] = \{1, 2, \ldots, n\}$ for positive integers a. For strings x, y, we use $x \cdot y$ to denote the concatenation of x and y. For $x \in \{0,1\}^n$ and $s, t \in [n]$, we write $x_{[s:t]}$ for the substring of x starting at index s and ending at index t and $x_{\leq t}$ for $x_{[1:t]}$.

We will use standard permutation notation, including S_n to denote the set of permutations of $[n]$. For any set A and permutations $\pi, \phi : A \to A$ we will use $\pi\phi : A \to A$ to denote their composition. For any such permutation π and a tuple $(u, v) \in A \times A$ or set $B \subset A$, we will write

$$\pi((u, v)) = (\pi(u), \pi(v))$$
$$\pi(B) = \{\pi(B) : a \in A\}$$

For any set A, we will use $\mathcal{U}(A)$ to denote the uniform distribution on A. For any predicate p, we will use $\mathbb{1}(p)$ to denote the variable that is 1 if p is true and 0 otherwise.

4.2 The StreamingCycles problem

The StreamingCycles(n, ℓ) problem is an n-player one way communication problem defined as follows. Informally, in this game every player is presented with an edge $e = (u, v) \in \binom{V}{2}$ together with x_u (here $x \in \{0,1\}^V$), with the promise that the union of the edges can be partitioned into n/ℓ disjoint cycles of length ℓ. The n^{th} player must output a cycle C in the graph together with the parity of the cycle, namely $\sum_{u \in C} x_u$.

We consider a distributional version of this problem in which the bit string x is chosen uniformly at random, the edges are assigned to the players uniformly at random, and the partitioning of the edge set into cycles is not known to the players in advance (otherwise the problem becomes trivial). The edges given to the players are public input (they are posted on a board visible to all players), whereas the bits x_u are private inputs. For every $t \in [n]$ the t^{th} player receives a message of c bits from the $(t-1)^{\text{th}}$ player and sends a message of c bits to the $(t+1)^{\text{th}}$ player. The message from the 0^{th} player is the zero string of length c. We define the problem formally in what follows.

Underlying graph G. Let the *underlying graph* $G = (V, E)$ with $V = [n]$ be defined as follows. We first define next : $V \to V$ as

$$\text{next}(j) = \begin{cases} j+1 & \text{if } j \not\equiv 0 \bmod \ell \\ j - \ell + 1 & \text{otherwise.} \end{cases}$$

Then for each vertex $v \in V$ define an edge

$$e_v = (v, \text{next}(v)) \tag{7}$$

and set

$$E = \{e_v\}_{v \in V}$$

The graph G will be revealed to the players over n time steps. We will use $T = [n]$ to denote the set of these time steps. The labels of the vertices of G are permuted before they are presented to the players. We define the permutation now.

Permuting labels of vertices. Choose two permutations $\boldsymbol{\pi} : T \to V$, $\boldsymbol{\sigma} : V \to V$ uniformly at random. These two permutations will determine, for each $t \in T$, the edge that is written on the board at time t (we will also say the edge "arrives" at time t, and that it is "present" at all $t' \geq t$). Specifically, the edge arriving at time t is

$$\mathbf{b}_t = \boldsymbol{\sigma}(e_{\boldsymbol{\pi}(t)}).$$

Definition 4.8 (Board state $\mathbf{B}_t$)**.** *We define the "board" $\mathbf{B}_t$ to be the sequence of all edge arrivals up to time t, i.e.*

$$\mathbf{B}_t = (\mathbf{b}_s)_{s=1}^t.$$

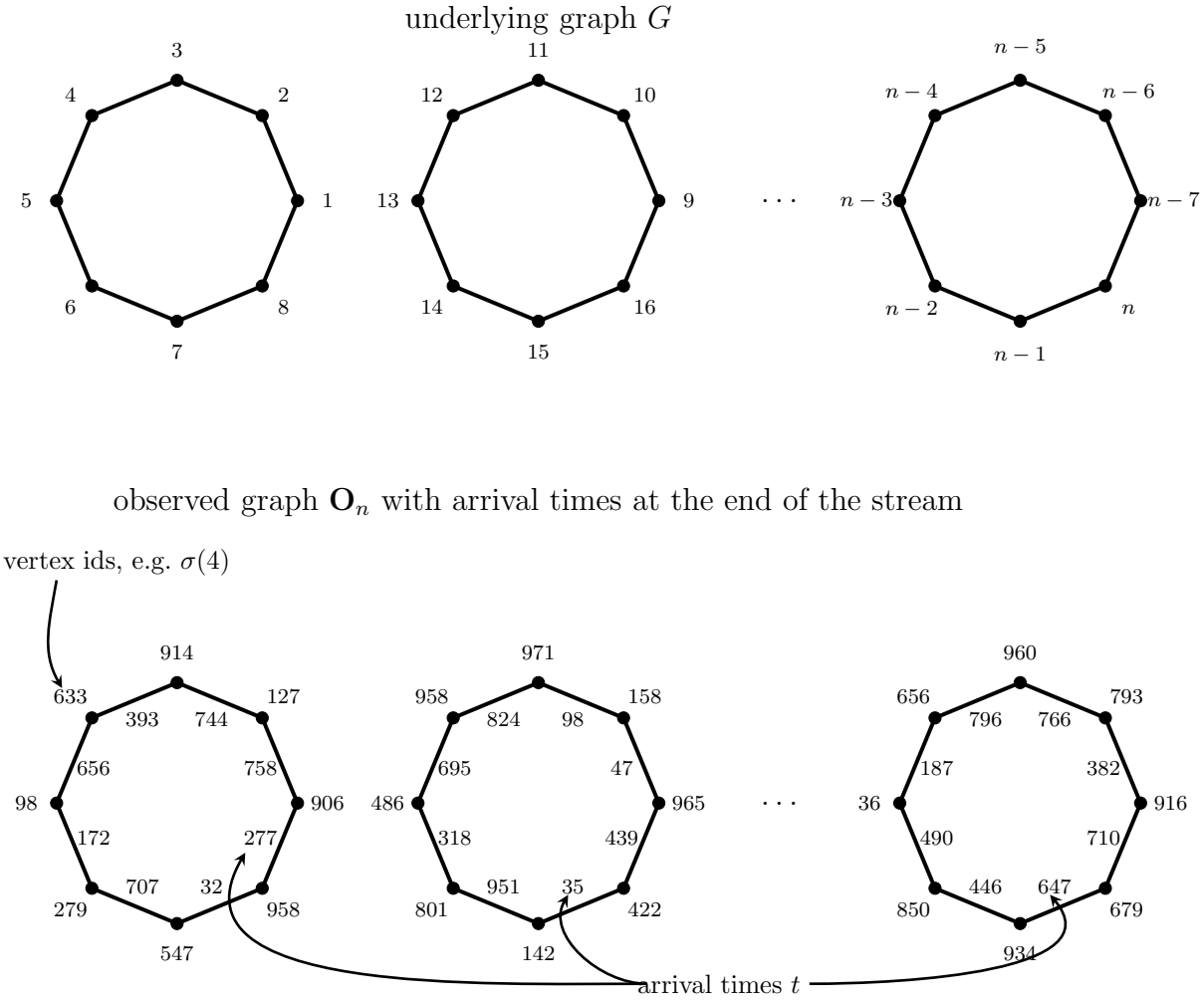

Figure 2: Illustration of the underlying graph G (top) and the observed graph (bottom). In this example $\pi^{-1}(1) = 758, \pi^{-1}(2) = 744, \pi^{-1}(3) = 393$ etc.

We will occasionally use $\mathbf{B}_{s:t}$ to refer to $(\mathbf{b}_i)_{i=s}^t$. This then defines the *observed graph* $\mathbf{O}_t = (V, \{\mathbf{b}_s\}_{s=1}^t)$ at time t. Note that $\mathbf{O}_t$ is isomorphic to $(V, \{e_v\}_{\boldsymbol{\pi}(v)\leq t})$, with $\boldsymbol{\sigma}$ giving an isomorphism.

Each subset $\mathbf{S}$ of $\{\mathbf{b}_s\}_{s=1}^t$ (or equivalently, subgraph of $\mathbf{O}_t$) can be associated with a t-bit binary string $x^{\mathbf{S}}$ given by

$$x_s = \begin{cases} 1 & \text{if } \mathbf{b}_s \in \mathbf{S} \\ 0 & \text{otherwise.} \end{cases}$$

In cases where it is unambiguous, we will use $\mathbf{S}$ to refer to $x^{\mathbf{S}}$ directly. Conversely, for each $x \in \{0,1\}^t$, we will write $\mathbf{O}_t^x$ for the subgraph of $\mathbf{O}_t$ such that $x^{\mathbf{O}_t^x} = x$.

Note that as $\mathbf{O}_t$ is isomorphic to a subgraph of a union of length-ℓ cycles, each of its components is either a length-ℓ cycle of a path of length $< \ell$, and the same holds for all subgraphs S.

Player input. Choose $\mathbf{X} \sim \mathcal{U}(\{0,1\}^T)$. At each time step $t \in T$, the t^{th} player receives three pieces of input: **(1)** the bit $\mathbf{X}_t \in \{0,1\}$, **(2)** a message $\mathbf{M}_{t-1} \in \{0,1\}^c$ from player $t-1$ (we let $\mathbf{M}_0 = \mathbf{0}^c$ for convenience), and **(3)** the contents of the board $\mathbf{B}_t$. Then, if $t < n$, player t sends $\mathbf{M}_t$ to player $t+1$.

Objective of the game. Player n, after timestep n, must output $v \in V$ and $\mathbf{X} \cdot \mathbf{C}_v$, where $\mathbf{C}_v$ is the component of $\mathbf{O}_n$ containing v (which, as G is a union of length-ℓ cycles, will necessarily be a length-ℓ cycle).

As the input distribution of the problem is fixed, we will by Yao's principle assume that the players are deterministic from now on.

4.3 The Underlying and the Observable Graph

In this section we introduce terminology and some necessary lemmas for understanding the relationship between the observed and the underlying graph.

Definition 4.9. *For each $t \in [n]$, $\boldsymbol{\Pi}_t$ is the set of possibilities for $\boldsymbol{\pi}$ that are compatible with $\mathbf{B}_t$ (see Definition 4.8), i.e.*

$$\boldsymbol{\Pi}_t = \left\{\pi \in S_n : \exists \sigma \in S_n, (\sigma(e_{\pi(s)}))_{s=1}^t = \mathbf{B}_t\right\}$$

Lemma 4.10. *For every $t \in [n]$, conditioned on $\mathbf{B}_t$, one has $\boldsymbol{\pi} \sim \mathcal{U}(\boldsymbol{\Pi}_t)$.*

Proof. Unconditionally, $(\boldsymbol{\pi}, \boldsymbol{\sigma})$ are uniformly distributed on $S_n \times S_n$, so conditioned on $\mathbf{B}_t$, they are distributed uniformly on

$$\{(\pi, \sigma) \in S_n \times S_n : (\sigma(e_{\pi(s)}))_{s=1}^t = \mathbf{B}_t\}$$

and so it will suffice to prove that for every $\pi \in \boldsymbol{\Pi}_t$, the number of $\sigma \in S_n$ such that

$$(\sigma(e_{\pi(s)}))_{s=1}^t = \mathbf{B}_t$$

is the same. For any pair $\pi_1, \pi_2 \in \boldsymbol{\Pi}_t$ We will give an injection from σ_1' such that

$$(\sigma_1'(e_{\pi_1(s)}))_{s=1}^t = \mathbf{B}_t$$

to σ_2' such that

$$(\sigma_2'(e_{\pi_2(s)}))_{s=1}^t = \mathbf{B}_t$$

implying that there at least as many such σ_2' as there are such σ_1', and so by symmetry there are the same number of each.

Since $\sigma_1, \sigma_2 \in \Pi_t$ by assumption, there exist σ_1, σ_2 such that

$$(\sigma_1(e_{\pi_1(s)}))_{s=1}^t = \mathbf{B}_t = (\sigma_2(e_{\pi_2(s)}))_{s=1}^t.$$

Fix a choice of such σ_1 and σ_2. The injection is defined by setting $\sigma_2' = \sigma_1'\sigma_1^{-1}\sigma_2$. For any fixing of σ_1, σ_2, this is an injective function of σ_1', as $\sigma_1^{-1}\sigma_2$ is a permutation, and

$$\begin{aligned}(\sigma_2'(e_{\pi_2(s)}))_{s=1}^t &= (\sigma_1'\sigma_1^{-1}\sigma_2(e_{\pi_2(s)}))_{s=1}^t \\ &= (\sigma_1'\sigma_1^{-1}\sigma_1(e_{\pi_1(s)}))_{s=1}^t \\ &= (\sigma_1'(e_{\pi_1(s)}))_{s=1}^t \\ &= \mathbf{B}_s\end{aligned}$$

completing the proof. □

observed graph $\mathbf{O}_t$ with arrival times at time $t = 500$

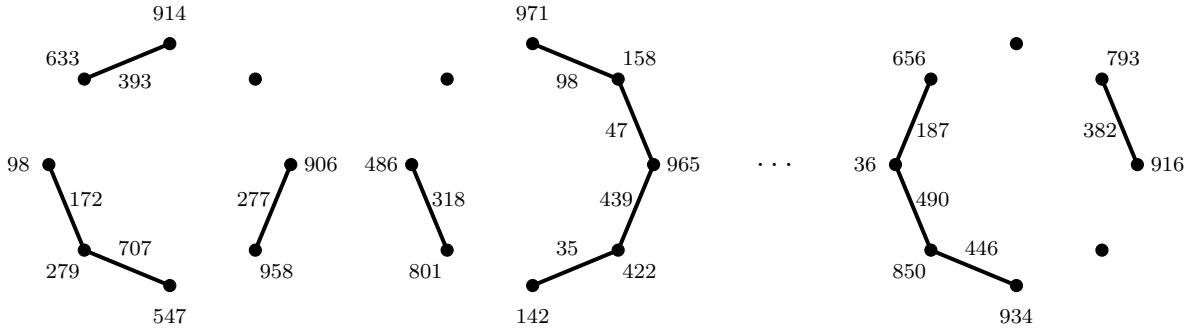

Figure 3: Illustration of the state of the board (i.e., the observed graph annotated with arrival times) at an intermediate point $t = 500$ in the stream.

4.4 Basic properties of collections of components

We write $\mathbb{Z}_+^{\{\}}$ to denote the set of multisets of non-negative integers. We will refer to each such multiset as a *collection type*. For such a multiset α we will write $\alpha[i]$ for the number of times i appears in α.

Definition 4.11. *For $t \in T$, $z \in \{0,1\}^t$ and $\alpha \in \mathbb{Z}_+^{\{\}}$ we write $z \sim_t \alpha$ if $\mathbf{O}_t^z$ is a union of components in $\mathbf{O}_t$, and for each $i \in [\ell]$, the number of i-edge components in $\mathbf{O}_t^z$ is $\alpha[i]$.*

Definition 4.12 (Weight and size of $\alpha \in \mathbb{Z}_+^{\{\}}$)**.** *For $\alpha \in \mathbb{Z}_+^{\{\}}$ we let $|\alpha|_* := \sum_{i \in \alpha} i$ denote the* weight *of α and let $|\alpha|$ denote the number of elements in α, which we refer to as the* size *of α.*

Definition 4.13 (Extensions and Merges)**.** *For any $\alpha \in \mathbb{Z}_+^{\{\}}$ we define an extension or merge as follows:*

[Extension] *Increment one of the elements in α by 1;*

[Merge] *Replace two elements $a, b \in \alpha$ with $a + b + 1$.*

Similarly, for any $t \in T$, $z \in \{0,1\}^t$, we say $\mathbf{O}_t^z$ experiences an extension or merge at time $t+1$ if:

[Extension] $\mathbf{b}_{t+1}$ *is incident to exactly one component in* $\mathbf{O}_t^z$.

[Merge] $\mathbf{b}_{t+1}$ *connects two components in* $\mathbf{O}_t^z$.

Note that $\mathbf{O}_t^z$ experiences an extension or merge at time $t+1$ iff $\mathbf{O}_t \sim_t \alpha$, $\mathbf{O}_{t+1} \sim_{t+1} \beta$ for β obtained by an extension or merge on α, respectively.

Definition 4.14 (Down set of $\beta \in \mathbb{Z}_+^{\{\}}$)**.** *For $\alpha, \beta \in \mathbb{Z}_+^{\{\}}$ we write $\alpha \in \beta - 1$ if β can be obtained from α by either an* extension *or a* merge *followed by possibly adding an arbitrary number of 1's to α.*

For $\beta \in \mathbb{Z}_+^{\{\}}$ and $\alpha \in \beta - 1$ we write $|\beta - \alpha|$ to denote the number of ones that need to be added to α after a merge or extension to obtain β.

Definition 4.15 (Growth event $\mathbf{Grow}(z,s,t)$)**.** *For every $t \in T$, every $s \in [t]$ and every $z \in \{0,1\}^t$, let $\boldsymbol{Grow}(z,s,t) = 1$ if $\mathbf{b}_{s+1}$ extends one or merges two components of $\mathbf{O}_s^{z_{\leq s}}$ and only single edge components of $\mathbf{O}_t^z$ arrive between $s+1$ and t, and let $\boldsymbol{Grow}(z,s,t) = 0$ otherwise.*

Lemma 4.16. *For every $t \in T$, B_t in the support of $\mathbf{B}_t$, every $r, r' \in \{0,1\}^t$ such that $r \sim_t \alpha$ and $r' \sim_t \alpha$ for some $\alpha \in \mathbb{Z}_+^{\{\}}$ when $\mathbf{B}_t = B_t$, then for all $\beta \in \mathbb{Z}_+^{\{\}}$*

$$\Pr_{\mathbf{B}_{t+1}}[r \cdot 1 \sim_{t+1} \beta | \mathbf{B}_t = B_t] = \Pr_{\mathbf{B}_{t+1}}\left[r' \cdot 1 \sim_{t+1} \beta | \mathbf{B}_t = B_t\right].$$

Proof. Recall that the sequence of edges $\mathbf{B}_t$ is given by

$$(\boldsymbol{\sigma}(e_{\boldsymbol{\pi}(s)}))_{s=1}^t$$

where $\boldsymbol{\sigma}$ and $\boldsymbol{\pi}$ are uniformly chosen permutations from $[n]$ to $[n]$, while the edge that arrives at time $t+1$ is given by

$$\boldsymbol{\sigma}(e_{\boldsymbol{\pi}(t+1)})$$

Recall also that $\mathbf{B}_t$ does *not* include the identity of $\boldsymbol{\sigma}, \boldsymbol{\pi}$ themselves. Now, as $r \sim_t \alpha$ and $r' \sim_t \alpha$, we can construct a graph automorphism $\phi : V \to V$ of $\mathbf{O}_t$, depending only on $\mathbf{B}_t$, such that ϕ swaps $\mathbf{O}_t^r$ with $\mathbf{O}_t^{r'}$ and is the identity everywhere else. Then, as $\mathbf{B}_t$ includes the arrival time of the edges, we can construct a permutation $\psi : T \to T$, also depending only on $\mathbf{B}_t$, such that for all $t \in T$

$$\phi(\mathbf{b}_{\psi(t)}) = \mathbf{b}_t$$

and ψ is the identity everywhere except the support of r and r'.

Recalling that $\mathbf{b}_s = \boldsymbol{\sigma}(e_{\boldsymbol{\pi}(s)})$, this means that, as the distribution of $\boldsymbol{\sigma}, \boldsymbol{\pi}$ conditional on $\mathbf{B}_t$ is uniform on

$$\{(\sigma, \pi) \in S_n \times S_n : (\sigma(e_{\pi(s)}))_{s=1}^t = (\mathbf{b}_s)_{s=1}^t\}$$

we have, for each $\sigma, \pi \in S_n \times S_n$,

$$\Pr_{\boldsymbol{\sigma},\boldsymbol{\pi}}[(\boldsymbol{\sigma}, \boldsymbol{\pi}) = (\sigma, \pi)] = \Pr_{\boldsymbol{\sigma},\boldsymbol{\pi}}[(\boldsymbol{\sigma}, \boldsymbol{\pi}) = (\phi\sigma, \psi\pi)].$$

Now, as ϕ is a graph automorphism swapping $\mathbf{O}_t^z$ and $\mathbf{O}_t^{z'}$, and $\psi(t+1) = t+1$, $\mathbf{b}_{t+1}$ will extend or merge $\mathbf{O}_t^z$ iff $\phi(\mathbf{b}_{\psi(t+1)})$ causes the same extension or merge in $\mathbf{O}_t^{z'}$, and so the above equation gives us the result. □

This allows us to define the probability of growth events using only the *collection type* of a collection of paths.

Definition 4.17. *For $s \in [n]$, a pair of collection types $\alpha, \beta \in \mathbb{Z}_+^{\{\}}$ such that $\alpha[1] = \beta[1]$ and an element B_s of the support of $\mathbf{B}_s$ we write*

$$p_s(\alpha, \beta, B_s) = \Pr_{\mathbf{B}_{s+1}}[z \cdot 1 \sim_{s+1} \beta | \mathbf{B}_s = B_s]$$

for any $z \in \{0,1\}^s$ such that $z \sim_s \alpha$. Furthermore, if $\beta[1] > \alpha[1]$, we write

$$p_s(\alpha, \beta, B_s) = p(\alpha, \beta', B_s)$$

where β' is β with $\beta[1] - \alpha[1]$ copies of 1 removed, whereas if $\beta[1] < \alpha[1]$, we set $p(\alpha, \beta, B_s) = 0$.

4.5 Extension and Merge Probabilities

In this section, we will prove the following lemma, bounding the probability that a merge or extension event will occur at a given time.

Lemma 4.18. *For some absolute constant $C > 0$, for each $t \le n - C2^{2\ell} n^{5/6} \log n$, there is an event $\mathcal{E}_t$ over $\mathbf{B}_t$ such that $\Pr_{\mathbf{B}_t}[\mathcal{E}_t] \ge 1 - 1/n^{\ell+1}$, and for any $B_t \in \mathcal{E}_t$ and for every α, β such that $|\alpha|_* \le \ell - 3$ (recall Definition 4.12)*

$$p_t(\alpha, \beta, B_t) \le \begin{cases} \frac{O(\alpha[a])}{n} & \text{if } \alpha \to \beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a] \cdot \alpha[b])}{(n-t)^2} & \text{if } \alpha \to \beta \text{ is a merge of paths of size } a \text{ and } b. \end{cases}$$

We will prove four cases of the lemma, depending on t and whether we are considering an extension or a merge.

We start by proving the lemma when $t \le n/8$ and β is reached from α by an extension.

Claim 4.19. *For each $t \le n/8$, every B_t in the support of $\mathbf{B}_t$, and for every α, β such that $\alpha \to \beta$ is an extension of a path of length a,*

$$p_t(\alpha, \beta, B_t) \le \frac{3\alpha[a]}{n}.$$

Proof. Let O_t be the graph corresponding to the edges in the sequence B_t. Let H be any subgraph of B_t with component sizes corresponding to α. It will suffice to bound the probability that a length-a component of H is extended at time $t+1$ when $\mathbf{B}_t = B_t$.

There are $2\alpha[a]$ distinct end vertices of components of length a in H. Call this set S. One of these paths is extended iff $\boldsymbol{\sigma}(\boldsymbol{\pi}(s+1)) \in S$ or $\boldsymbol{\sigma}(\text{next}(\boldsymbol{\pi}(s+1))) \in S$. For any σ, suppose $\boldsymbol{\sigma} = \sigma$. Now for each element of S, there is exactly one possibility for $\boldsymbol{\pi}(s+1)$ that will cause one of $\boldsymbol{\sigma}(\boldsymbol{\pi}(s+1)) \in S$ or $\boldsymbol{\sigma}(\text{next}(\boldsymbol{\pi}(s+1))) \in S$ to hold, and conditioned on $\mathbf{B}_t = B_t$, $\boldsymbol{\sigma} = \sigma$, $\boldsymbol{\pi}(t+1)$ is uniformly distributed on a set of size $n - t \ge 7n/8$ (as fixing $\mathbf{B}_t$ and $\boldsymbol{\sigma}$ fixes $(\boldsymbol{\pi}(s))_{s=1}^t$). So the result follows by taking a union bound. □

Next, we prove it for merges when $t \le n/8$. To do this, we will first introduce a new concept, the swap graph $\mathcal{G}_t^z$.

The swap graph $\mathcal{G}_t^z$. For any $t \in T$, $z \in \{0,1\}^t$, the vertex set of this (undirected) graph will be $\mathbf{\Pi}_t$ as defined in Definition 4.9, the set of permutations in the support of $\boldsymbol{\pi}$ that are consistent with the observed board $\mathbf{B}_t$.

We now define the edge set of $\mathcal{G}_t^z$. If $\mathbf{O}_t^z$ is *not* a single component of size at most $\ell - 2$ in $\mathbf{O}_t$, $\mathcal{G}_t^z$ is the complete graph for convenience.

Now suppose that $\mathbf{O}_t^z$ is a single component of size at most $\ell - 2$. We now define edges incident on a permutation $\pi \in \mathbf{\Pi}_t$ in $\mathcal{G}_t^z$. To define these edges, first note that the set of vertices

$$\pi(\{s \in [t] : z_s = 1\})$$

induces a path P in the underlying graph G. Let k denote the length of this path. Write

$$P = (u_i)_{i=1}^k, u_i \in V, \tag{8}$$

where for every $i \in [k-1]$ one has $u_{i+1} = \text{next}(u_i)$. In particular, we have, as per (7), that $e_{u_i} = (u_i, u_{i+1})$ for each $i \in [k-1]$. For every path

$$P' = (u_i')_{i=1}^k, u_i' \in V \tag{9}$$

in G of the same length first define

$$\psi : V \to V \tag{10}$$

to swap u_i and u_i' for each $i \in [k-1]$ while being the identity everywhere else, and then add an edge (π, π'), where

$$\pi' = \psi\pi, \tag{11}$$

to $\mathcal{G}_t^z$ if $\pi' \in \mathbf{\Pi}_t$. We have that π' is consistent with $\mathbf{B}_t$ on every edge except, possibly, the edges immediately before and after P' or P. In particular, $\pi' \in \mathbf{\Pi}_t$ if none of those edges arrive in $\mathbf{B}_t$.

Lemma 4.20. *If $t \leq n/8$, the minimum degree of $\mathcal{G}_t^z$ is at least $n/8$.*

Proof. The result is trivial if $\mathbf{O}_t^z$ is not a single component of size at most $\ell - 2$ in $\mathbf{O}_t$, so we will consider only the case where it is.

We will to show that, for any $\pi \in \mathbf{\Pi}_t$, there are at least $n/12 - 1$ ways of choosing π' by the process defined above such that π' will still be in $\mathbf{\Pi}_t$. Let $P, P', \psi, \pi' = \psi\pi$ be as in (8), (9), (10) and (11) above. Note that π' will be in $\mathbf{\Pi}_t$ if there is some choice of permutation $\sigma' : V \to V$ such that $(\sigma'(e_{\pi'(s)}))_{s=1}^t = \mathbf{B}_t$. As there is a permutation σ such that $(\sigma(e_{\pi(s)}))_{s=1}^t = \mathbf{B}_t$, choosing

$$\sigma' = \psi\sigma$$

will guarantee that $\sigma'(e_{\pi'(s)}) = \mathbf{b}_s$ for each $s \in [t]$. So $(\sigma'(e_{\pi'(s)}))_{s=1}^t = \mathbf{B}_t$ will hold provided

$$\sigma'(e_{\pi'(s)}) = \sigma(e_{\pi'(s)}) = \sigma(e_{\pi(s)}) \tag{12}$$

for all s such that $z_s = 0$. The condition in (12) will be satisfied when there is no $s \in [t]$ such that $z_s = 0$ and $e_{\pi(s)}$ is incident on either P or P'. This will happen iff there is no $s \in [t]$ such that either $\text{next}(\pi(s)) = u_1'$ or $\pi(s) = u_k'$.

Therefore, the number of valid choices for the new path, and therefore the degree of π in $\mathcal{G}_t^z$, is given by the number of pairs w_0, w_k in V such that **(a)** $\pi(w_0), \pi(w_k) > t$, and **(b)** there is a path $(w_i)_{i=0}^k$ such that $w_i = \text{next}(w_{i-1})$ for each $i \in [k]$. We lower bound the number of such pairs now.

We will start by lower bounding the number of disjoint pairs w_0, w_k in V that satisfy **(b)**. We may assume without loss of generality that $k \le \ell/2$, as there is a one-to-one correspondence between paths $(w_0)_{i=1}^{k}$ and paths $(w'_i)_{i=0}^{\ell-k}$ such that $w'_0 = w_k$ and $w'_{\ell-k} = w_0$.

We will consider two cases to lower bound this number of disjoint pairs:

$k \le \ell/4$: We may divide each cycle in the underlying graph G into $\left\lfloor \frac{\ell}{2k} \right\rfloor$ disjoint blocks of $2k$ consecutive vertices. In each such block we may fit k disjoint pairs satisfying **(b)**. Therefore each cycle contains

$$\left\lfloor \frac{\ell}{2k} \right\rfloor \cdot k \ge \left(\frac{\ell}{2k} - 1 \right) \cdot k \ge \ell/4$$

disjoint pairs satisfying **(b)**, and so G contains at least $n/4$ of them.

$k > \ell/4$ For any cycle in G, let v be a vertex in the cycle. Then for each $i < k$, $(\text{next}^i(v), \text{next}^{i+k}(v))$ is a pair satisfying **(b)**, and these pairs are all disjoint. So there are at least $\min(k, \ell-k) > \ell/4$ such disjoint pairs in the cycle, and therefore at least $n/4$ in G.

Therefore, as $t \le n/8$, there are at most $n/8$ pairs (w_0, w_k) that do not satisfy **(a)**. Combining with the bounds above, we get that there are at least $n/8$ pairs (w_0, w_k) that satisfy both **(a)** and **(b)**. This completes the proof. □

We are now ready to prove the result for merges when $t \le n/8$.

Claim 4.21. *For each $t \le n/8$, every B_t in the support of $\mathbf{B}_t$, and for every α, β such that $\alpha \to \beta$ is a merge of paths of length a, b,*

$$p_t(\alpha, \beta, B_s) \le \frac{14\alpha[a] \cdot \alpha[b]}{n^2}.$$

Proof. Fix $\mathbf{B}_t = B_t$ and let O_t be the corresponding value of $\mathbf{O}_t$. For any pair of components P_1, P_2 of lengths a, b in O_t, they will merge at time $t+1$ iff two criteria are satisfied:

1. The two paths $\mathbf{Q}_1 = \boldsymbol{\sigma}^{-1}(P_1)$, $\mathbf{Q}_2 = \boldsymbol{\sigma}^{-1}(P_2)$ in the underlying graph G need only one edge to be connected.
2. That edge arrives at time $t+1$.

If the first criterion is satisfied, the second will be with probability $\frac{1}{n-t} \le \frac{8}{7n}$, so we will seek to bound the first. Let Π be the value of $\mathbf{\Pi}_t$ (recall Definition 4.9) given $\mathbf{B}_t = B_t$, and let Π^* be the set of $\pi \in \Pi$ that would lead to the first criterion being satisfied if $\boldsymbol{\pi} = \pi$. We will prove that $|\Pi^*| \le \frac{8}{n}|\Pi|$, which by Lemma 4.10 suffices to prove that the probability of $\boldsymbol{\pi}$ satisfying the first criterion is at most $8/n$.

Let $z \in \{0,1\}^t$ be given by $z_s = 1$ if an edge in P_1 arrived at time s, and $z_s = 0$ otherwise. Let $\mathcal{G}_t^z$ be the swap graph $\mathcal{G}_t^z$ when $\mathbf{B}_t = B_t$. Each vertex of this graph corresponds to a choice of π. Note that, given $\mathbf{B}_t = B_t$, the values of $\boldsymbol{\sigma}^{-1}$ on non-isolated vertices of $\mathbf{O}_t$ and the values of $\boldsymbol{\pi}$ on $[t]$ each uniquely determine the other, so in particular such a vertex determines a choice of $\boldsymbol{\sigma}^{-1}$. Thus, the choice of a vertex in $\mathcal{G}_t^z$ uniquely determines a choice Q_1, Q_2 of the length a, b paths $\mathbf{Q}_1$, $\mathbf{Q}_2$. Moving along an edge of $\mathcal{G}_t^z$ corresponds to choosing a different Q_1 while leaving Q_2 the same.

We observe that for each $\pi \in \Pi$, there are at most 2 neighbors of π in Π^*, as when fixing Q_2 there is at most two choices of Q_1 that lead to them being separated by exactly one edge. By

Lemma 4.20, the minimum degree of $\mathcal{G}_t^z$ is at least $n/8$. Therefore, each vertex in Π^* is a neighbor of at least $n/8$ vertices of Π, each of which neighbors at most 2 vertices of Π^*, and so

$$|\Pi| \geq \frac{n}{16}|\Pi^*|.$$

Therefore the probability that $\boldsymbol{\pi}$ satisfied the first criterion given $\mathbf{B}_t = B_t$ is at most $n/16$. We have that the probability of any given pair of components of length a, b merging is at most

$$\frac{16}{n} \cdot \frac{7}{8n} = \frac{14}{n^2}$$

and so the result follows. □

For the remaining cases we will use a new random variable $\mathbf{U}_s$, that gives the final board state, up to re-orderings of the edge arrivals from time $s+1$ to n. As conditioning on $\mathbf{U}_s$ means that Lemma 4.16 no longer holds, we will introduce new notation for the *average* growth probability.

Definition 4.22 (Average growth probabilities)**.** *For each $s \in [n]$, the random variable $\mathbf{U}_s$ is given by $\mathbf{B}_s$ and $(\boldsymbol{\pi}(i))_{i=1}^s$. For each $\alpha, \beta \in \mathbb{Z}_+^{\{\}}$ such that $\alpha[1] = \beta[1]$,*

$$\upsilon_s(\alpha, \beta, \mathbf{U}_s) = \frac{1}{|z \in \{0,1\}^s : z \sim_s \alpha|} \sum_{\substack{z \in \{0,1\}^s \\ z \sim_s \alpha}} \Pr_{\mathbf{B}_{s+1}}[z \cdot 1 \sim_{s+1} \beta | \mathbf{U}_s]$$

while if $\beta[1] > \alpha[1]$, we write

$$\upsilon_s(\alpha, \beta, B_s) = \upsilon_s(\alpha, \beta', B_s)$$

where β' is β with $\beta[1] - \alpha[1]$ copies of 1 removed, whereas if $\beta[1] < \alpha[1]$, we set $\upsilon(\alpha, \beta, B_s) = 0$.

For extensions and merges we will define high probability events $\mathcal{E}_s^e$, $\mathcal{E}_s^m$ respectively, over $\mathbf{U}_s$, such that conditioned on these events the quantity $\upsilon(\alpha, \beta, U_t^e)$ satisfies the natural analog of bound in Lemma 4.18. Lemma 4.23 below shows that such events imply Lemma 4.18. The rest of this section is devoted to defining such events.

Lemma 4.23. *For any $s \in [n]$, let $\mathcal{E}_s^e$, $\mathcal{E}_s^m$ be events such that for all $(U_s^e, U_s^m) \in \mathcal{E}_s^e \times \mathcal{E}_s^m$, and any α, β such that $|\alpha|_* \leq \ell - 3$,*

$$\upsilon(\alpha, \beta, U_s^e) \leq \frac{O(\alpha[a])}{n} \qquad \text{if } \alpha \to \beta \text{ is an extension of a path of size } a$$
$$\upsilon(\alpha, \beta, U_s^m) \leq \frac{O(\alpha[a] \cdot \alpha[b])}{(n-s)^2} \qquad \text{if } \alpha \to \beta \text{ is a merge of paths of size } a \text{ and } b$$

and each event occurs with probability at least $1 - 1/(2n^{\ell+3})$ over $\mathbf{U}_s$. Then there is an event $\mathcal{E}_s$ such that for any $B_s \in \mathcal{E}_s$ and for every α, β such that $|\alpha|_* \leq \ell - 3$

$$p_s(\alpha, \beta, B_s) \leq \begin{cases} \frac{O(\alpha[a])}{n} & \text{if } \alpha \to \beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a] \cdot \alpha[b])}{(n-s)^2} & \text{if } \alpha \to \beta \text{ is a merge of paths of size } a \text{ and } b. \end{cases}$$

and $\Pr_{\mathbf{B}_s}[\mathcal{E}_s] \geq 1 - 1/n^{\ell+1}$.

Proof. Define $\mathcal{E}_s$ to be the event that $\Pr_{\mathbf{U}_s}[\mathcal{E}_s^e \cap \mathcal{E}_s^m | \mathbf{B}_s] \geq 1 - 1/n^2$. Then

$$\begin{aligned} 1 - \Pr_{\mathbf{U}_s}[\mathcal{E}_s^e \cap \mathcal{E}_s^m] &= \mathop{\mathbb{E}}_{B_s}\left[1 - \Pr_{\mathbf{U}_s}[\mathcal{E}_s^e \cap \mathcal{E}_s^m | \mathbf{B}_s]\right] \\ &\geq \frac{1}{n^2}(1 - \Pr_{\mathbf{B}_s}[\mathcal{E}_s]) \end{aligned}$$

and so

$$\Pr_{\mathbf{B}_s}[\mathcal{E}_s] \geq 1 - 1/n^{\ell+1}.$$

For any $B_s \in \mathcal{E}_s$, $z \sim_s \alpha$, by applying Lemma 4.16, we get

$$\begin{aligned} \Pr_{\mathbf{B}_{s+1}}[z \cdot 1 \sim_{s+1} \beta | \mathbf{B}_s = B_s] &= \frac{1}{|z \in \{0,1\}^s : z \sim_s \alpha|} \sum_{\substack{z \in \{0,1\}^s \\ z \sim_s \alpha}} \Pr_{\mathbf{B}_{s+1}}[z \cdot 1 \sim_{s+1} \beta | \mathbf{B}_s = B_s] \\ &= \mathop{\mathbb{E}}_{\mathbf{U}_s}\left[\frac{1}{|z \in \{0,1\}^s : z \sim_s \alpha|} \sum_{\substack{z \in \{0,1\}^s \\ z \sim_s \alpha}} \Pr_{\mathbf{B}_{s+1}}[z \cdot 1 \sim_{s+1} \beta | \mathbf{U}_s] | \mathbf{B}_s = B_s\right] \\ &= \mathop{\mathbb{E}}_{\mathbf{U}_s}[\upsilon(\alpha, \beta, \mathbf{U}_s) | \mathbf{B}_s = B_s] \\ &\leq \mathop{\mathbb{E}}_{\mathbf{U}_s}[\upsilon(\alpha, \beta, \mathbf{U}_s) | \mathbf{B}_s = B_s, \mathcal{E}_s^e \cap \mathcal{E}_s^m] + \frac{1}{n^2} \\ &\leq \begin{cases} \frac{O(\alpha[a])}{n} + \frac{1}{n^2} & \text{if } \alpha \to \beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a] \cdot \alpha[b])}{(n-s)^2} + \frac{1}{n^2} & \text{if } \alpha \to \beta \text{ is a merge of paths of size } a \text{ and } b \end{cases} \end{aligned}$$

This gives the result.

□

$\mathbf{U}_s$ tells us how edges visible in $\mathbf{B}_s$ correspond to edges in the underlying graph. We will use the following lemma to obtain tight bounds on how often $\mathbf{U}_s$ contains certain "patterns" of edges. Specifically, we will be interested in the following types of patterns:

(1) a path P with a edges in the underlying graph G such that all of the edges of P have arrived by time t, but neither of the adjacent edges have (see Fig. 4, (a))

(2) a path P with a edges in the underlying graph G such that all of the edges of P have arrived by time t, neither of the adjacent edges have arrived, and the **next** vertex in the canonical order (defined by the next function) does not have any incident edges at time t (see Fig. 4, (b))

(3) a path P with a edges in the underlying graph G such that all of the edges of P have arrived by time t, neither of the adjacent edges have arrived, and the **previous** vertex in the canonical order (defined by the next function) does not have any incident edges at time t (see Fig. 4, (c))

(4) a path P with $a + b$ edges in the underlying graph G such that the first a and last b of the edges of P have arrived by time t, the $(a+1)^{\text{th}}$ edge has not arrived, and neither of the edges adjacent to P have arrived (see Fig. 5).

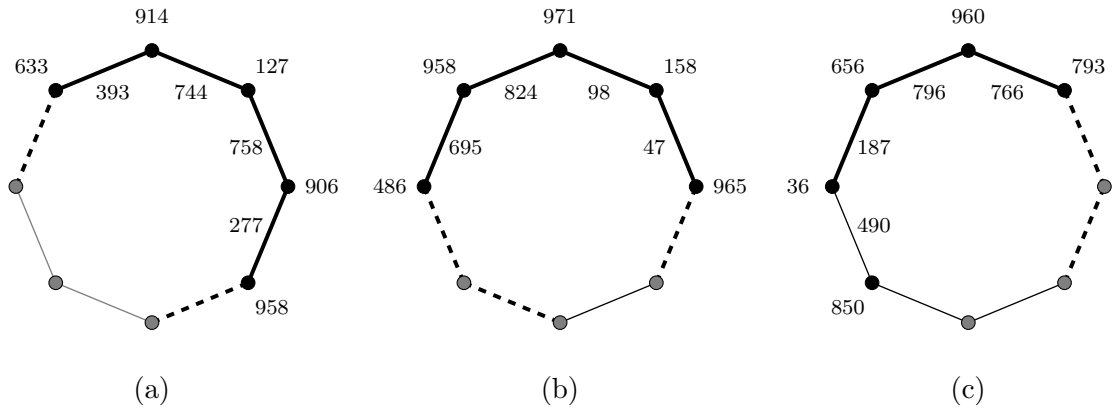

Figure 4: Illustration of patterns **(1), (2)** and **(3)**. Solid edges must be present at time t, dashed edges must not be present at time t,

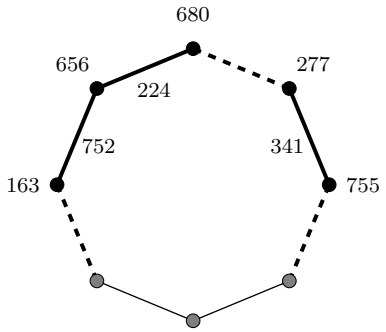

Figure 5: Illustration of pattern **(4)**. Solid edges must be present at time t, dashed edges must not be present at time t,

Bounds on the number of occurrences of the first three types of patterns are then used to bound the average extension probability in Claim 4.25 below. The first and the forth are then used to bound the average merge probability in Claim 4.26 below.

The following lemma bounds the number of "patterns" of the above types simultaneously. In order to cover all patterns above, we prove a more general lemma that applies to all possible patterns as opposed to just the ones above. For an integer $k \geq 1$ we encode such patterns by a binary string y of length k, where for $i \in [k]$ we have $y_i = 1$ if the i^{th} edge in a consecutive segment of k edges in the underlying graph must be present, and $y_i = 0$ if the i^{th} edge must be absent.

Lemma 4.24. *For any $t \in T$, let $y \in \{0,1\}^k$ for some $k \in [\ell]$. Suppose $|\overline{y}| \in \{2,3\}$. For each $v \in V$, let $(v_i)_{i=1}^{k+1}$ be the path in the underlying graph G such that $v_1 = v$ and $v_{i+1} = next(v_i)$ for each $i \in [k]$. Let $\mathbf{Y}_v = 1$ if for all $i \in [k]$, $\mathbb{1}(\boldsymbol{\pi}^{-1}(v_i) \leq t) = y_i$. Let $\mathbf{Y} = \sum_{v \in V} \mathbf{Y}_v$.*

Then, for every constant $C > 0$ there is a $D > 0$ depending only on C such that for every

$$t \in \left[n/8, n - D2^{2\ell} n^{5/6} \log n\right]$$

one has

$$\mathbf{Y} = \Theta\left(n \cdot \left(\frac{t}{n}\right)^{|y|} \cdot \left(\frac{n-t}{n}\right)^{|\overline{y}|}\right)$$

with probability at least $1 - n^{-C\ell}$ *over* $\mathbf{U}_t$.

Proof. Let $\boldsymbol{\beta} : V \to \{0,1\}$ be given by $\boldsymbol{\beta}(v) = \mathbb{1}(\boldsymbol{\pi}^{-1}(v) \leq t)$. Then for all $v \in V$, $\mathbb{E}_{\mathbf{U}_t}[\boldsymbol{\beta}(v)] = t/n$, but the values $\boldsymbol{\beta}(V)$ are not independent, as $\boldsymbol{\beta}$ is fixed to have exactly t values equal to 1, and $n-t$ equal to 0.

Now, let $\boldsymbol{\beta}'$ be defined as follows:

1. Draw $\mathbf{t}' \sim \mathrm{Bi}(n, t/n)$.
2. "Correct" $\boldsymbol{\beta}$ to have exactly $\mathbf{t}'$ values equal to 1, by either flipping $\mathbf{t}' - t$ randomly chosen zeroes to 1, or flipping $t - \mathbf{t}'$ randomly chosen ones to 0, as necessary.

Now, $\boldsymbol{\beta}'$ will have $\mathbf{t}'$ ones, and which values of $\boldsymbol{\beta}'$ are 1 will be chosen uniformly at random, so we have both $\mathbb{E}_{\mathbf{U}_t, \mathbf{t}'}[\boldsymbol{\beta}'(v)] = t/n$ for all $v \in V$ and the values $\boldsymbol{\beta}'(V)$ being independent.

Now, for each $v \in V$, let $(v_i)_{i=1}^{k+1}$ be as defined in the lemma statement, and let $\mathbf{Y}'_v$ be 1 if for all $i \in [k]$, $\mathbb{1}(\boldsymbol{\beta}'(v)) = y_i$. Let $\mathbf{Y}' = \sum_{v \in V} \mathbf{Y}'_v$. Now, since for every $v \in V$ the value of $\boldsymbol{\beta}(v)$ (resp. $\boldsymbol{\beta}'(v)$) affects at most $k \leq \ell$ values of $\mathbf{Y}(u), u \in V$ (resp. $\mathbf{Y}'(u), u \in V$), we have

$$|\mathbf{Y}' - \mathbf{Y}| \leq \left|\{v \in V : \boldsymbol{\beta}(v) \neq \boldsymbol{\beta}'(v\}\right| \leq \ell |\mathbf{t}' - t|$$

and so by applying the Chernoff bounds to $\mathbf{t}'$, with probability $1 - n^{-C\ell}/2$ over $\mathbf{t}'$,

$$\begin{aligned} |\mathbf{Y}' - \mathbf{Y}| &= O(\sqrt{\ell \cdot \mathbb{E}[\mathbf{t}'] \log n}) \\ &= O(\sqrt{\ell t \log n}) \\ &= O(\sqrt{\ell n \log n}) \end{aligned}$$

and so we may proceed by bounding the distribution of $\mathbf{Y}'$.

The events $\{\mathbf{Y}'_v : v \in V\}$ are not independent. However for each v, $\mathbf{Y}'_v$ is independent of the set of events that do not depend on $\boldsymbol{\beta}'(v_i)$ for any $i \in [k]$. This is all but $2k - 2 < 2\ell - 1$ of the other random variables in $\{\mathbf{Y}'_v : v \in V\}$. So we may construct a 2ℓ set partition of V, denoted by $[V_i : i \in [2\ell]]$, such that, for each $i \in [2\ell]$, the events $\{\mathbf{Y}'_v : v \in V_i\}$ are independent.

Now, for each $v \in V$, $\mathbf{Y}'_v = 1$ with probability $\left(\frac{t}{n}\right)^{|y|}\left(\frac{n-t}{n}\right)^{|\overline{y}|}$, and 0 otherwise. Now, for each $i \in [2\ell]$, let $\mathbf{Y}'^{(i)} = \sum_{v \in V_i} \mathbf{Y}'_v$. By the Chernoff bounds, with probability $1 - n^{-C\ell}/4\ell$ over $\mathbf{U}_t$ and $\mathbf{t}'$,

$$\left|\mathbf{Y}'^{(i)} - \mathop{\mathbb{E}}_{\mathbf{U}_t, \mathbf{t}'}\left[\mathbf{Y}'^{(i)}\right]\right| = O\left(\sqrt{\ell \cdot \mathbb{E}\left[\mathbf{Y}'^{(i)}\right] \log n \log \ell}\right)$$

and so by taking a union bound over $\left\{\mathbf{Y}'^{(i)} : i \in [2\ell]\right\}$, with probability $1 - n^{-C\ell}/2$ over $\mathbf{U}_t$ and $\mathbf{t}'$,

$$\left|\mathbf{Y}' - \mathop{\mathbb{E}}_{\mathbf{U}_t, \mathbf{t}'}\left[\mathbf{Y}'\right]\right| = O\left(\ell^{3/2}\sqrt{\mathbb{E}[\mathbf{Y}'] \log n \log \ell}\right).$$

Combining this with our bound on $|\mathbf{Y}' - \mathbf{Y}|$ above, with probability $1 - n^{-C\ell}$ over $\mathbf{U}_t$ (as $\mathbf{Y}$ is independent of $\mathbf{t}'$),

$$\left|\mathbf{Y} - \mathop{\mathbb{E}}_{\mathbf{U}_t,\mathbf{t}'}\left[\mathbf{Y}'\right]\right| = O\left(\sqrt{\ell n \log n} + \ell^{3/2}\sqrt{\mathbb{E}[\mathbf{Y}']\log n \log \ell}\right).$$

Since

$$\mathop{\mathbb{E}}_{U_t,\mathbf{t}'}\left[\mathbf{Y}'\right] = n\left(\frac{t}{n}\right)^{|y|}\left(\frac{n-t}{n}\right)^{|\overline{y}|}$$

we have

$$\begin{aligned}\sqrt{\ell n \log n} + \ell^{3/2}\sqrt{\mathbb{E}[\mathbf{Y}']\log n \log \ell} &= \sqrt{\ell n \log n} + \ell^{3/2}\sqrt{n \log n \log \ell}\left(\frac{t}{n}\right)^{|y|/2}\left(\frac{n-t}{n}\right)^{|\overline{y}/2|} \\ &\leq 2\ell^{3/2}\sqrt{n \log n \log \ell}\end{aligned}$$

and so, using the fact that $t \geq n/8$,

$$\mathbf{Y} = n\left(\frac{t}{n}\right)^{|y|}\left(\frac{n-t}{n}\right)^{|\overline{y}|} \cdot (1+\gamma), \tag{13}$$

where

$$\gamma = O\left(\ell^{3/2}\sqrt{\log n \log \ell}\left(\frac{t}{n}\right)^{-|y|}\left(\frac{n-t}{n}\right)^{-|\overline{y}|}\right)/\sqrt{n}.$$

We now note that

$$\begin{aligned}|\gamma| &= O\left(\ell^{3/2}\sqrt{\log n \log \ell}2^{3\ell}\left(\frac{n}{n-t}\right)^3\right)/\sqrt{n} && \left(\text{since } \left(\frac{t}{n}\right)^{-|y|} \leq 2^{3\ell} \text{ since } t \geq n/8\right) \\ &\leq O\left(\sqrt{\log n}2^{5\ell}\left(\frac{n}{n-t}\right)^3\right)/\sqrt{n} && (\text{since } \ell^{3/2} \leq 2^{2\ell} \text{ for } \ell \geq 2) \\ &\leq O\left(\left(\frac{2^{2\ell}n^{5/6}\log n}{n-t}\right)^3\right) \\ &\leq O(1/D^3) && (\text{since } t \geq n - D2^{2\ell}n^{5/6}\log n) \\ &\leq 1/2\end{aligned}$$

as long as D is larger than an absolute constant. Thus, we get that $|\gamma| \leq 1/2$ and the lemma follows by (13). □

We now use this result to bound $\upsilon(\alpha, \beta, U_t^e)$ in the case where $t \geq n/8$ and $\alpha \to \beta$ is an extension.

Claim 4.25. *For some absolute constant $C > 0$, for each $t \in \left[n/8, n - C2^{2\ell}n^{5/6}\log n\right]$, there is an event $\mathcal{E}_t^e$ such that for all $U_t^e \in \mathcal{E}_t^e$, and any α, β such that $|\alpha|_* \leq \ell - 3$ ($|\alpha|_*$ is the weight of α as per Definition 4.12) and $\alpha \to \beta$ is an extension of a length-a path, one has*

$$\upsilon(\alpha, \beta, U_t^e) \leq \frac{O(\alpha[a])}{n}.$$

The event $\mathcal{E}_t^e$ occurs with probability at least $1 - 1/2n^{\ell+3}$ over $\mathbf{U}_t$.

Proof. Given $\mathbf{U}_t$, we naturally associate with each a-edge component in $\mathbf{O}_t$ an $(a+2)$-edge path $(v_i)_{i=1}^{a+3}$ that consists of the component present at time t together with the two adjacent edges in the underlying graph G that have not arrived yet (see Fig. 6). Specifically, we have $v_{i+1} = \text{next}(v_i)$ for each $i \in [a+2]$, and $\boldsymbol{\pi}^{-1}(v_1) > t$, $\boldsymbol{\pi}^{-1}(\{v_i : 1 < i \leq a+1\}) \subseteq [t]$, $\boldsymbol{\pi}^{-1}(v_{a+2}) > t$ (pattern of type **(1)** above, see Fig. 4). Call the number of such paths $\mathbf{Y}_1$.

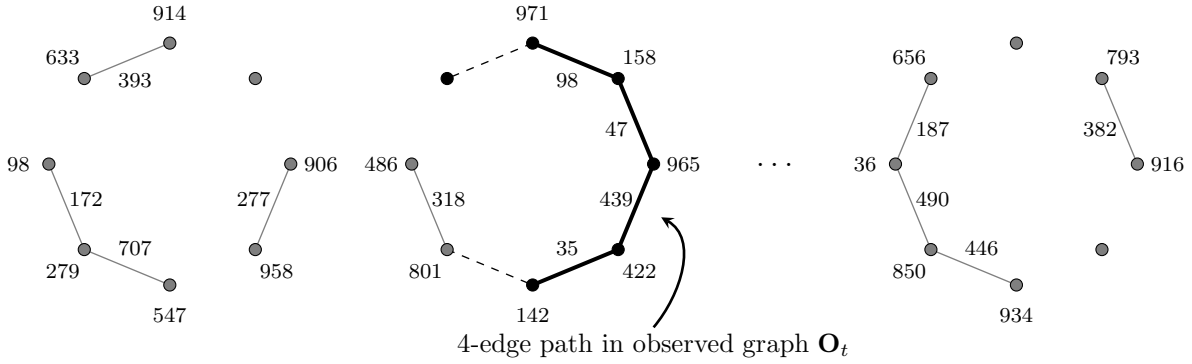

Figure 6: Path with $a = 4$ edges in the observed graph and the corresponding $(a+2)$-edge path in the underlying graph (incident edges that have not arrived yet are shown as dashed)

Such a component will be extended at time $t+1$ iff the edge that arrives at this time is incident to it and not to any other component, that is iff either:

(1) $\boldsymbol{\pi}(t+1) = v_1$ and $\boldsymbol{\pi}^{-1}(\text{next}^{-1}(v_1)) > t$ (i.e., the other endpoint of the arriving edge does not have any incident edges in the observed graph $\mathbf{O}_t$)

or

(2) $\boldsymbol{\pi}(t+1) = v_{a+2}$ and $\boldsymbol{\pi}^{-1}(\text{next}(v_{a+2})) > t$ (i.e., the other endpoint of the arriving edge does not have any incident edges in the observed graph $\mathbf{O}_t$).

For each of these, the second criterion is determined by $\mathbf{U}_t$ (recall Definition 4.22), and the first will be satisfied with probability $\frac{1}{n-t}$ over $\boldsymbol{\pi}(t+1)$. So conditioned on $\mathbf{U}_t$, the average probability that an a-edge component in $\mathbf{O}_t$ is extended is $\frac{1}{n-t} \cdot \frac{\mathbf{Y}_2+\mathbf{Y}_3}{\mathbf{Y}_1}$, where $\mathbf{Y}_2$ is the number of such paths where $\boldsymbol{\pi}^{-1}(\text{next}^{-1}(v_1)) > t$ and $\mathbf{Y}_3$ is the number where $\boldsymbol{\pi}^{-1}(\text{next}(v_{a+2})) > t$ (patterns of type **(2)** and **(3)** above, see Fig. 4).

By applying Lemma 4.24 with $C = 3$ (as ℓ must be at least 3 and so n must be as well, and thus $n^{C\ell} > 6n^{\ell+3}$) and taking a union bound,

$$\mathbf{Y}_2 + \mathbf{Y}_3 = \Theta\left(n\left(\frac{t}{n}\right)^a\left(\frac{n-t}{n}\right)^3\right)$$
$$\mathbf{Y}_1 = \Theta\left(n\left(\frac{t}{n}\right)^a\left(\frac{n-t}{n}\right)^2\right)$$

with probability $1 - 1/2n^{\ell+3}$ over $\mathbf{U}_t$, provided

$$n/8 \leq t \leq n - D2^{2\ell}n^{5/6}\log n$$

for a sufficiently large constant C. So let $\mathcal{E}_t^e$ be the event that this holds, then $\Pr_{\mathbf{U}_t}[\mathcal{E}_t^e] \geq 1-1/2n^{\ell+3}$ and for all $U_t \in \mathcal{E}_s^e$,

$$\begin{aligned} \upsilon(\alpha,\beta,U_t) &= \frac{\alpha[a]}{n-t} \cdot \Theta\left(\frac{n\left(\frac{t}{n}\right)^a\left(\frac{n-t}{n}\right)^3}{n\left(\frac{t}{n}\right)^a\left(\frac{n-t}{n}\right)^2}\right) \\ &= \frac{O(\alpha[a])}{n} \end{aligned}$$

completing the proof. □

Finally, we consider the case where $t \geq n/8$ and $\alpha \to \beta$ is a merge.

Claim 4.26. *For some absolute constant $C > 0$, for each $t \in \left[n/8, n - C2^{2\ell}n^{5/6}\log n\right]$, there is an event $\mathcal{E}_t^e$ such that for all $U_t^e \in \mathcal{E}_t^e$, and any α, β such that $|\alpha|_* \leq \ell - 3$ and $\alpha \to \beta$ is an merge of a length-a and length-b path,*

$$\upsilon(\alpha,\beta,U_t^e) \leq \frac{O(\alpha[a]\cdot\alpha[b])}{(n-t)^2}$$

$\mathcal{E}_t^e$ occurs with probability at least $1 - 1/2n^{\ell+3}$ over $\mathbf{U}_t$.

Proof. Given $\mathbf{U}_t$, we naturally associate with each pair of an a-edge and a b-edge component in $\mathbf{O}_t$ an $(a+2)$-edge path $(u_i)_{i=1}^{a+3}$ and a $(b+2)$-edge path $(v_i)_{i=1}^{b+3}$ in the underlying graph G that consist of the edges of the two components together with the not yet arrived adjacent edges (see Fig. 7). Specifically, $u_{i+1} = \text{next}(u_i)$ for each $i \in [a+2]$, $v_{i+1} = \text{next}(v_i)$ for each $i \in [b+2]$, and $\boldsymbol{\pi}^{-1}(u_1) > t, \boldsymbol{\pi}^{-1}(v_1) > t$,

$$\boldsymbol{\pi}^{-1}(\{u_i : 1 < i \leq a+1\} \cup \{v_i : 1 < i \leq b+1\})$$

is contained in $[t]$, $\boldsymbol{\pi}^{-1}(u_{a+2}) > t, \boldsymbol{\pi}^{-1}(v_{b+2}) > t$. Both correspond to patterns of type **(1)** above, see Fig. 4. Call the number of such paths $\mathbf{Y}_1$, $\mathbf{Y}_2$, respectively.

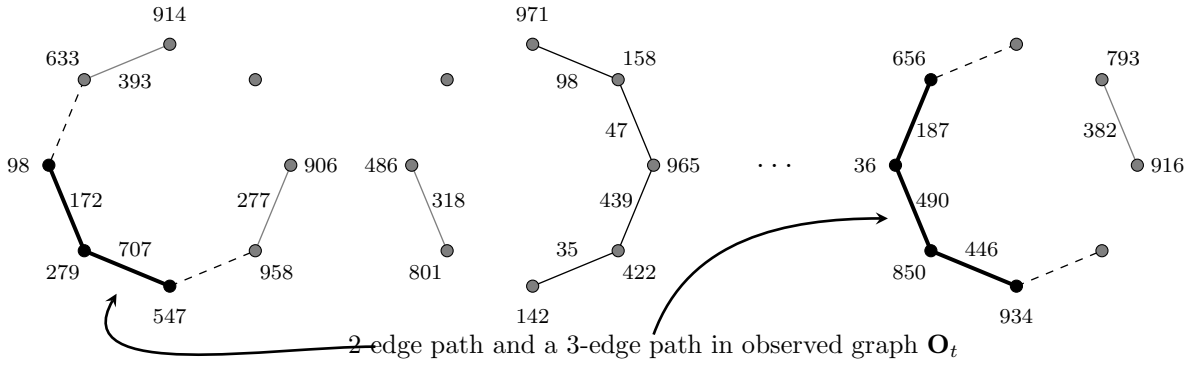

Figure 7: A pair of paths with $a = 2$ and $b = 3$ edges in the observed graph respectively, together with the corresponding $(a+2)$-edge and $(b+2)$-edge paths in the underlying graph (incident edges that have not arrived yet are shown as dashed)

Such a component will be merged at time $t+1$ iff the edge that arrives at this time is incident to both of them, that is iff either:

1. $\boldsymbol{\pi}(t+1) = v_1$ and $u_{a+2} = v_1$.

2. $\boldsymbol{\pi}(t+1) = u_1$ and $v_{b+2} = u_1$.

So conditioned on $\mathbf{U}_t$, the average probability that a pair of an a-edge and a b-edge component in $\mathbf{O}_t$ is $\frac{1}{n-t}$ times the fraction of those pairs such that either $u_{a+2} = v_1$ or $v_{b+2} = u_1$.

Let $\mathbf{Y}_3$ be the number of paths $(w_i)_{i=1}^{a+b+4}$ such that $w_{i+1} = \text{next}(w_i)$ for each $i \in [a+b+3]$, $\boldsymbol{\pi}^{-1}(w_1) > t$, $\boldsymbol{\pi}^{-1}(w_{a+2}) > t$, $\boldsymbol{\pi}^{-1}(w_{a+b+3}) > t$, and

$$\boldsymbol{\pi}^{-1}(\{w_i : 1 < i < a+2 \vee a+2 < i < a+b+3\})$$

is contained in $[t]$. Let $\mathbf{Y}_4$ be the number of paths $(w_i)_{i=1}^{a+b+4}$ such that $w_{i+1} = \text{next}(w_i)$ for each $i \in [a+b+3]$, $\boldsymbol{\pi}^{-1}(w_1) > t$, $\boldsymbol{\pi}^{-1}(w_{b+2}) > t$, $\boldsymbol{\pi}^{-1}(w_{a+b+3}) > t$, and

$$\boldsymbol{\pi}^{-1}(\{w_i : 1 < i < b+2 \vee b+2 < i < a+b+3\})$$

is contained in $[t]$. Both correspond to patterns of type **(4)** above, see Fig. 5.

Then the fraction of these pairs such that $u_{a+2} = v_1$ or $v_{b+2} = u_1$ is either $\frac{\mathbf{Y}_3+\mathbf{Y}_4}{\mathbf{Y}_1\mathbf{Y}_2}$ (if $a \neq b$) or $\frac{\mathbf{Y}_3}{\binom{\mathbf{Y}_1}{2}}$ (if $a = b$). If $a = b$ and $\mathbf{Y}_1 \leq 1$ the lemma holds trivially, so we may assume $\mathbf{Y}_1 > 1$ and thus both of these are bounded by

$$2\frac{\mathbf{Y}_3 + \mathbf{Y}_4}{\mathbf{Y}_1\mathbf{Y}_2}$$

and by applying Lemma 4.24 with $C = 3$ (as ℓ must be at least 3 and so n must be as well, and thus $n^{C\ell} > 8n^{\ell+3}$) and taking a union bound,

$$\mathbf{Y}_1 + \mathbf{Y}_2 = \Theta\left(n\left(\frac{t}{n}\right)^{a+b}\left(\frac{n-t}{n}\right)^3\right)$$
$$\mathbf{Y}_3 = \Theta\left(n\left(\frac{t}{n}\right)^{a}\left(\frac{n-t}{n}\right)^2\right)$$
$$\mathbf{Y}_4 = \Theta\left(n\left(\frac{t}{n}\right)^{b}\left(\frac{n-t}{n}\right)^2\right)$$

with probability $1 - 1/2n^{\ell+3}$ over $\mathbf{U}_t$, provided

$$n/8 \leq t \leq n - C2^{10\ell}n^{5/6}\log n$$

for a sufficiently large constant C. So let $\mathcal{E}_t^m$ be the event that this holds, then $\Pr_{\mathbf{U}_t}[\mathcal{E}_t^m] \geq 1 - 1/2n^{\ell+3}$ and for all $U_t \in \mathcal{E}_s^m$,

$$\begin{aligned}\upsilon(\alpha, \beta, U_t) &= \Theta\left(\frac{n\left(\frac{t}{n}\right)^{a+b}\left(\frac{n-t}{n}\right)^3}{n\left(\frac{t}{n}\right)^{a}\left(\frac{n-t}{n}\right)^2 n\left(\frac{t}{n}\right)^{b}\left(\frac{n-t}{n}\right)^2}\right) \cdot \frac{\alpha[a]\cdot\alpha[b]}{n-t} \\ &= \frac{O(\alpha[a]\cdot\alpha[b])}{(n-t)^2}\end{aligned}$$

completing the proof. □

By combining these four claims, Lemma 4.18 is proved.

4.6 A decomposition of a typical message

The main result of this section is Lemma 3.5, a central tool in our analysis. For every $t \in T$ and $1 \leq s < t$ the lemma gives a decomposition of a typical message sent by player t (equivalently, the state of the streaming algorithm after processing the t^{th} part of the input) in terms of the message sent by player s and an auxiliary function that represents the mapping from the message of player t to the message of player t.

As the players are deterministic, we may assume that the t^{th} player sends $\mathbf{M}_t := \mathbf{g}_t(\mathbf{X}_t; \mathbf{M}_{t-1})$ for some function

$$\mathbf{g}_t(\cdot,\cdot) : \{0,1\} \times \{0,1\}^c \to \{0,1\}^c, \tag{14}$$

depending only on $\mathbf{B}_t$, where c is the size of the messages that the players exchange. We let $\mathbf{F}_t : \{0,1\}^t \to \{0,1\}$ denote the *indicator function of the message* $\mathbf{M}_t$ sent by the t^{th} player, defined as

$$\mathbf{F}_t(x) = \begin{cases} 1 & \text{if, conditioned on } \mathbf{B}_t,\ \mathbf{X}_t = x \text{ would cause player } t \text{ to send } \mathbf{M}_t \\ 0 & \text{otherwise.} \end{cases}$$

For convenience we let $\mathbf{F}_0$ denote the trivial function from $\emptyset$ to $\{0,1\}$, and $\mathbf{M}_0 = 0^c$. Note that $\mathbf{F}_t, \mathbf{M}_t, \mathbf{g}_t$ above are random variables depending on $\mathbf{X}_{\leq t}$ and $\mathbf{B}_t$; we let F_t denote a realization of $\mathbf{F}_t$, and let m_t denote a realization of $\mathbf{M}_t$ (as usual, we use boldface for random variables and regular font to represent their realizations). We note that indicator functions F_t are in one to one correspondence with messages m_t .

Let $\mathcal{F}_t$ denote the collection of possible indicator functions at time t given $\mathbf{B}_t$, and note that for every t such messages partition $\{0,1\}^t$:

$$\sum_{F_t \in \mathcal{F}_t} F_t(x) = 1.$$

Since F_t's are in one to one correspondence with possible messages of the t^{th} player, the function $\mathbf{g}_t$ from (14) naturally extends to map $\{0,1\} \times \mathcal{F}_{t-1}$ to $\mathcal{F}_t$:

$$\mathbf{g}_t(\cdot,\cdot) : \{0,1\} \times \mathcal{F}_{t-1} \to \mathcal{F}_t. \tag{15}$$

The proof of Lemma 3.5 relies on the observation that for every $t \in T$, $F_t \in \mathcal{F}_t$, and every $x_{\leq t}$ one has

$$F_t(x_{\leq t}) = \sum_{F_s \in \mathcal{F}_s} F_s(x_{\leq s}) \cdot \mathbf{r}_t(x_{[s+1:t]}; F_s, F_t), \tag{16}$$

where $\mathbf{r}_t(x_{[s+1:t]}; F_s, F_t)$ is the indicator function for $x_{[s+1:t]}$ taking F_s to F_t, i.e.

$$\mathbf{r}(x_{[s+1:t]}; F_s, F_t) = \sum_{(F_{s+1},\ldots,F_{t-1}) \in \mathcal{F}_{s+1} \times \ldots \times \mathcal{F}_{t-1}} \mathbf{I}\left[\bigwedge_{j=s+1}^{t} \mathbf{g}_j(F_{j-1}, x_j) = F_j \right]. \tag{17}$$

Note that $\mathbf{r}$ is a random function depending only on $\mathbf{B}_t$. The function $\mathbf{r}$ satisfies

Claim 4.27. *For every $1 \leq s < t \leq n$, every $F_s \in \mathcal{F}_s$ one has for every $x_{[s+1:t]}$ that*

$$\mathbf{r}(x_{[s+1:t]}; F_s, F_t) \in \{0,1\}$$

for every $F_t \in \mathcal{F}_t$ and

$$\sum_{F_t \in \mathcal{F}_t} \mathbf{r}(x_{[s+1:t]}; F_s, F_t) = 1.$$

We now consider the random variable $\mathbf{X} \in \mathcal{U}(\{0,1\}^n)$, and corresponding random variables $\mathbf{F}_t : \{0,1\}^t \to \{0,1\}$ for each $t \in [n]$ to be the indicator of the message that results from $\mathbf{X}_{\leq t}$. Because $\mathbf{X}$ is uniform and the messages are deterministic, $(\mathbf{X} \mid \mathbf{F}_t = F_t)$ is uniform over the support of F_t. Therefore for each $t \in [n]$ and F_t if we define

$$\widetilde{F}_t(z) := \frac{2^s}{||F_t||_1} \widehat{F}_t(z).$$

(with $\widetilde{F}_0$ being the trivial function again) this satisfies

$$\widetilde{F}_t(z) = \mathop{\mathbb{E}}_{\mathbf{X}}\big[(-1)^{z \cdot \mathbf{X}_{\leq t}} \big| \mathbf{F}_t = F_t, \mathbf{B}_t\big].$$

Now consider the distribution of $(\mathbf{X} \mid \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t)$ for any $s < t$. $\mathbf{X} = x$ is consistent with $(\mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t)$ if and only if $\mathbf{F}_s(x_{\leq s}) = 1$ and $\mathbf{r}(x_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t) = 1$. Since $\mathbf{X}$ is uniform, $(\mathbf{X} \mid \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t)$ is uniform over those x consistent with $(\mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t)$; and since the $\mathbf{F}_s$ and $\mathbf{r}$ constraints are on disjoint sets of coordinates, this means that $\mathbf{X}_{\leq s}$ and $\mathbf{X}_{[s+1:t]}$ are independent conditioned on $(\mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t)$, with $(\mathbf{X}_{\leq s} \mid \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t) = (\mathbf{X}_{\leq s} \mid \mathbf{F}_s, \mathbf{B}_t)$ and $(\mathbf{X}_{[s+1:t]} \mid \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t)$ being uniform over the support of $\mathbf{r}(\cdot; \mathbf{F}_s, \mathbf{F}_t)$. Therefore if we define

$$\widetilde{\mathbf{r}}(z; F_s, F_t) := \frac{2^{t-s}}{||\mathbf{r}(\cdot; F_s, F_t)||_1} \widehat{\mathbf{r}}(z; F_s, F_t)$$

where the Fourier transform $\widehat{\mathbf{r}}$ is with respect to its first argument, this satisfies

$$\widetilde{\mathbf{r}}(z; F_s, F_t) = \mathop{\mathbb{E}}_{\mathbf{X}}\big[(-1)^{z \cdot \mathbf{X}_{[s+1:t]}} \big| \mathbf{F}_s = F_s, \mathbf{F}_t = F_t, \mathbf{B}_t\big].$$

These definitions lead to the following:

Lemma 3.5. *For every $s, t \in T$ with $s < t$, and any $z_{\leq t}$, we have:*

$$\widetilde{\mathbf{F}}_t(z_{\leq t}) = \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s}) \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t) \Big| \mathbf{F}_t, \mathbf{B}_t\Big].$$

Proof. As discussed above, $\mathbf{X}_{\leq s}$ and $\mathbf{X}_{[s+1:t]}$ are independent conditioned on $\mathbf{F}_s$, $\mathbf{F}_t$, $\mathbf{B}_t$. Therefore

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{X}}\big[(-1)^{z \cdot \mathbf{X}} \big| \mathbf{F}_t\big] &= \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\mathop{\mathbb{E}}_{\mathbf{X}}\big[(-1)^{z \cdot \mathbf{X}} \big| \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t\big] \Big| \mathbf{F}_t, \mathbf{B}_t\Big] \\
&= \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\mathop{\mathbb{E}}_{\mathbf{X}_{\leq s}, \mathbf{X}_{[s+1:t]}}\big[(-1)^{z_{\leq s} \cdot \mathbf{X}_{\leq s}} (-1)^{z_{[s+1:t]} \cdot \mathbf{X}_{[s+1:t]}} \big| \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t\big] \Big| \mathbf{F}_t, \mathbf{B}_t\Big] \\
&= \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\mathop{\mathbb{E}}_{\mathbf{X}_{\leq s}}\big[(-1)^{z_{\leq s} \cdot \mathbf{X}_{\leq s}} \big| \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t\big] \mathop{\mathbb{E}}_{\mathbf{X}_{[s+1:t]}}\big[(-1)^{z_{[s+1:t]} \cdot \mathbf{X}_{[s+1:t]}} \big| \mathbf{F}_s, \mathbf{F}_t, \mathbf{B}_t\big] \Big| \mathbf{F}_t, \mathbf{B}_t\Big] \\
&= \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s}) \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t) \Big| \mathbf{F}_t, \mathbf{B}_t\Big].
\end{aligned}$$

□

In the next section we combine Lemma 3.5 with bounds on the probability of extensions and merges from Section 4.5 to obtain Lemma 3.4, our key lemma on the expected evolution of Fourier coefficients as a function of t.

4.7 Evolution of Fourier coefficients (Lemma 3.4)

Define, for every $t \in [n]$ and $\beta \in \mathbb{Z}_+^{\{\}}$:

$$\mathbf{H}_\beta^t = \sum_{z \in \{0,1\}^t, z \sim_t \beta} \widetilde{\mathbf{F}}_t(z)^2 \tag{18}$$

The main result of this section is Lemma 3.4, restated here for convenience of the reader:

Lemma 3.4. *For every $t \in T$ one has for $\beta \in \mathbb{Z}_+^{\{\}}$ that contain at least one component of size more than 1 and have $|\beta|_* \le \ell - 2$,*

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\beta^t\right] \le \sum_{s=1}^{t-1} \sum_{\alpha \in \beta - 1} q(|\beta - \alpha|) \cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}[\mathbf{H}_\alpha^s \cdot p(\alpha, \beta, \mathbf{B}_s)]$$

and for β with all components of size 1

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\beta^t\right] \le q(|\beta|),$$

where

$$q(k) = \begin{cases} \left(\frac{8c}{k}\right)^k & \text{if } k \le c \\ 2^c & \text{otherwise.} \end{cases}$$

Before proving Lemma 3.4, we derive several useful bounds on Fourier coefficients of Boolean functions, which we later apply to $\mathbf{F}_t$. To bound sums of Fourier coefficients of $\mathbf{F}_t$, we need a corollary of the KKL Lemma of [KKL88], an extended version of that of [GKK$^+$07].

Lemma 4.28 ([KKL88])**.** *Let f be a function $f : \{0,1\}^n \to \{-1,0,1\}$. Let $A = \{x | f(x) \neq 0\}$. Then for every $\delta \in [0,1]$ we have*

$$\sum_{s \in \{0,1\}^n} \delta^{|s|} \widehat{f}(s)^2 \le \left(\frac{|A|}{2^n}\right)^{\frac{2}{1+\delta}}.$$

We will use

Lemma 4.29 (Lemma F.3 of [KK19])**.** *Let f be a function $f : \{0,1\}^n \to \{0,1\}$. Let $A = \{x | f(x) \neq 0\}$, and let $|A| \ge 2^{n-c}$. Then for every $y \in \{0,1\}^n$ and every $q \le c$ one has*

$$\sum_{\substack{x \in \{0,1\}^n \\ |x \oplus y| = q}} \widetilde{f}^2(x) \le \left(\frac{4c}{q}\right)^q.$$

The following lemma is an immediate consequence of Lemma 4.29:

Lemma 4.30. *Let f be a function $f : \{0,1\}^n \to \{0,1\}$. Let $A = \{x | f(x) \neq 0\}$, and let $|A| \ge 2^{n-c}$. Then for every $k \in \{1, \ldots, c\}$, we have*

$$\sum_{v \in \{0,1\}^{n-1}, |v| = k} \widetilde{f}(1 \cdot v)^2 \le \left(\frac{4c}{k}\right)^k.$$

Lemma 4.31. *Let f be a function $f : \{0,1\}^n \to \{0,1\}$. Let $A = \{x | f(x) \neq 0\}$, and let $|A| \geq 2^{n-c}$. Then for every $k \in \{1, \ldots, c\}$, any partition P of $[n]$, and any set S consisting of size-k unions of elements of P, we have*

$$\sum_{v \in S} \widetilde{f}(v)^2 \leq \left(\frac{4c}{k}\right)^k.$$

Proof. For each $s \in P$, write j_s for the first index in s. For each $z \in \{0,1\}^n$, let z' be defined by:

$$z'_j = \begin{cases} z \cdot s \bmod 2 & \text{if } j = j_s \text{ for some } s \in P \\ z_j & \text{otherwise.} \end{cases}$$

Then, define:

$$A' = \{z' : z \in A\}$$

The transformation $z \to z'$ is bijective, as it is given by a triangular matrix with ones along the diagonal. Therefore, $|A'| = |A|$, and so $|A'|/2^n = 2^{-c}$. Now, let f' be the indicator of A'. By the definition of $z \to z'$, for all $V \subseteq P$,

$$\begin{aligned} \widetilde{f}\left(\bigcup V\right) &= \frac{1}{|A|} \sum_{z \in A} \prod_{s \in V} (-1)^{z \cdot s} \\ &= \frac{1}{|A|} \sum_{z \in A} \prod_{s \in V} (-1)^{z'_{j_s}} \\ &= \frac{1}{|A'|} \sum_{z \in A'} \prod_{s \in V} (-1)^{z \cdot \{j_s\}} \\ &= \widetilde{f'}\left(\bigcup_{s \in V} \{j_s\}\right) \end{aligned}$$

and so by applying Lemma 4.28 with $\delta = k/c$,

$$\begin{aligned} \sum_{v \in S} \widetilde{f}(v)^2 &\leq \sum_{v \in \{0,1\}^n : |v| = k} \widetilde{f'}(v)^2 \\ &= \frac{2^{2n}}{|A'|^2} \sum_{v \in \{0,1\}^n : |v| = k} \widehat{f}(v)^2 \\ &\leq \delta^{-k} \frac{2^{2n}}{|A'|^2} \sum_{v \in \{0,1\}^n} \delta^{|v|} \widehat{f}(v)^2 \\ &\leq \delta^{-k} \frac{2^{2n}}{|A'|^2} \left(\frac{|A'|}{2^n}\right)^{\frac{2}{1+\delta}} \\ &= \delta^{-k} 2^{\frac{2c\delta}{1+\delta}} \\ &\leq \left(\frac{4c}{k}\right)^k. \end{aligned}$$

□

We will use this with Parseval's identity to obtain a bound on coefficients of r. Here $\|.\|_2$ is the functional ℓ_2-norm given by $\|f\|_2^2 = 2^{-n} \sum_{x \in \{0,1\}^n} |f(x)|^2$.

Lemma 4.32 (Parseval)**.** *For every function $f : \{0,1\}^n \to \mathbb{R}$,*

$$\sum_{v \in \{0,1\}^n} \widehat{f}(v)^2 = \|f\|_2^2 .$$

Note that if, as in the previous lemma statements, f is the indicator function of a set A such that $|A| = 2^{n-c}$, this implies that

$$\sum_{v \in \{0,1\}^n} \widetilde{f}(v)^2 = 2^c$$

as $\widetilde{f} = 2^{n-c}\widehat{f}$.

Lemma 4.33. *For any $s < t \in T$, and any $k \in [t-s-1]$,*

$$\sum_{\substack{z \in \{0,1\}^{t-s-1} \\ |z| = k}} \mathbb{E}_{\mathbf{X}_{[s+1:t]}} \left[\widetilde{\mathbf{r}}(1 \cdot z; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{X}_{\leq s}, \mathbf{B}_t \right] \leq q(k)$$

where $q(k) = \begin{cases} \left(\frac{8c}{k}\right)^k & \text{if } k \leq c \\ 2^c & \text{otherwise.} \end{cases}$

Proof. Condition on $\mathbf{B}_t = B_t$, $\mathbf{X}_{\leq s} = y$, for any B_t, y in the support of $\mathbf{B}_t, \mathbf{X}_{\leq s}$. Let F_s be the resulting value of $\mathbf{F}_s$. Conditioned on these, each $x \in \{0,1\}^{t-s}$ results in a fixed value of $\mathbf{F}_t$ when $\mathbf{X}_{[s+1:t]} = x$. As $\mathbf{F}_t$ is supported on at most 2^c elements, this gives a size at most 2^c partition of $\{0,1\}^{t-s}$. Call this partition $\mathcal{A}$, and for each F_t in the support of $\mathbf{F}_t$ conditioned on $\mathbf{B}_t = B_t$, $\mathbf{X}_{\leq s} = y$, call the (unique) corresponding element of the partition $A(F_t)$.

For each $A \in \mathcal{A}$, write c_A for $n - \log|A|$, and p_A for $\Pr[\mathbf{F}_t = F_t | \mathbf{X}_{\leq s} = y, \mathbf{B}_t = B_t]$, where $A(F_t) = A$. As $\mathbf{F}_t = F_t$ iff $\mathbf{X}_{[s+1:t]} \in A$, $p_A = |A| 2^{-n} = 2^{-c_A}$. Note that by Lemma 4.30 and Parseval's equality, for each F_t in the support of $\mathbf{F}_t$ conditioned on $\mathbf{B}_t = B_t$, $\mathbf{X}_{\leq s} = y$,

$$\sum_{\substack{z \in \{0,1\}^{t-s-1} \\ |z| = k}} \widetilde{\mathbf{r}}(1 \cdot z; F_s, F_t)^2 \leq q_{A(F_t)}(k)$$

where for each $A \in \mathcal{A}$,

$$q_A(k) = \begin{cases} \min\left(2^{c_A}, \left(\frac{4c_A}{k}\right)^k\right) & \text{if } k \leq c_A \\ 2^{c_A} & \text{otherwise} \end{cases}$$

(note the different constants from q in the first line). We will consider two cases, based on the value of $k = |z|$.

Case 1: $c < 2k$ For each $A \in \mathcal{A}$, $q_A(k) \leq 2^{c_A}$ and so $p_A q_A(k) \leq 1$. Therefore,

$$\begin{aligned} \sum_{\substack{z \in \{0,1\}^{t-s-1} \\ |z| = k}} \mathbb{E}_{\mathbf{X}_{[s+1:t]}} \left[\widetilde{\mathbf{r}}(1 \cdot z; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{X}_{\leq s} = y, \mathbf{B}_t = B_t \right] &\leq \sum_{A \in \mathcal{A}} p_A q_A(k) \\ &\leq |\mathcal{A}| \\ &\leq 2^c \\ &\leq q(k) \end{aligned}$$

as if $k \geq c$, $q(k) = 2^c$, and if $k \in (c/2, c]$, $q(k) \geq 8^{c/2} \geq 2^c$.

Case 2: $c \geq 2k$ Let $\mathcal{A} = \mathcal{A}^+ \cup \mathcal{A}^-$, with $\mathcal{A}^+$ containing all A in $\mathcal{A}$ such that $c_A > c$ and $\mathcal{A}^-$ containing everything else. Then $q_A(k) \leq \left(\frac{4c}{k}\right)^k$ for each A in $\mathcal{A}^-$ and so

$$\sum_{a \in A^-} p_A q_A(k) \leq \left(\frac{4c}{k}\right)^k.$$

At the same time for each $A \in A^+$, $c_A > c \geq 2k$, so

$$\begin{aligned} p_A q_A &= \left(\frac{4c_A}{k}\right)^k 2^{-c_A} \\ &\leq \left(\frac{4c}{k}\right)^k 2^{-c} \end{aligned}$$

as for $x \geq 2k$, $\frac{d}{dx}(x^k 2^{-x}) = (k-x)x^{k-1}2^{-x} < 0$.

Therefore,

$$\begin{aligned} \sum_{\substack{z \in \{0,1\}^{t-s-1} \\ |z| = k}} \mathbb{E}_{\mathbf{X}_{[s+1:t]}} \left[\widetilde{\mathbf{r}}(1 \cdot z; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{X}_{\leq s} = y, \mathbf{B}_t = B_t\right] &\leq \sum_{A \in \mathcal{A}^+} p_A q_A(k) + \sum_{A \in \mathcal{A}^-} p_A q_A(k) \\ &\leq \left(\frac{4c}{k}\right)^k + |\mathcal{A}| \left(\frac{4c}{k}\right)^k 2^{-c} \\ &\leq 2\left(\frac{4c}{k}\right)^k \\ &\leq q(k) \end{aligned}$$

and so the result follows. □

Lemma 4.34. *For any $s \leq t \in T$, and any $k \in [t-s]$,*

$$\sum_{\substack{z \in \{0,1\}^{t-s} \\ z \text{ a union of } k \text{ paths}}} \mathbb{E}_{\mathbf{X}_{[s+1:t]}} \left[\widetilde{\mathbf{r}}(z; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{X}_{\leq s}, \mathbf{B}_t\right] \leq q(k)$$

where $q(k) = \begin{cases} \left(\frac{8c}{k}\right)^k & \text{if } k \leq c \\ 2^c & \text{otherwise.} \end{cases}$

Proof. The proof follows the proof of Lemma 4.33 except for its reliance on Lemma 4.31 as opposed to Lemma 4.30. Condition on $\mathbf{B}_t = B_t$, $\mathbf{X}_{\leq s} = y$, for any B_t, y in the support of $\mathbf{B}_t, \mathbf{X}_{\leq s}$. Let F_s be the resulting value of $\mathbf{F}_s$. Conditioned on these, each $x \in \{0,1\}^{t-s}$ results in a fixed value of $\mathbf{F}_t$ when $\mathbf{X}_{[s+1:t]} = x$. As $\mathbf{F}_t$ is supported on at most 2^c elements, this gives a size at most 2^c partition of $\{0,1\}^{t-s}$. Call this partition $\mathcal{A}$, and for each F_t in the support of $\mathbf{F}_t$ conditioned on $\mathbf{B}_t = B_t$, $\mathbf{X}_{\leq s} = y$, call the (unique) corresponding element of the partition $A(F_t)$.

For each $A \in \mathcal{A}$, write c_A for $n - \log|A|$, and p_A for $\Pr[\mathbf{F}_t = F_t | \mathbf{X}_{\leq s} = y, \mathbf{B}_t = B_t]$, where $A(F_t) = A$. As $\mathbf{F}_t = F_t$ iff $\mathbf{X}_{[s+1:t]} \in A$, $p_A = |A|2^{-n} = 2^{-c_A}$. Note that by Lemma 4.31 and Parseval's equality, for each F_t in the support of $\mathbf{F}_t$ conditioned on $\mathbf{B}_t = B_t$, $\mathbf{X}_{\leq s} = y$,

$$\sum_{\substack{z \in \{0,1\}^{t-s} \\ z \text{ a union of } k \text{ paths}}} \widetilde{\mathbf{r}}(z; F_s, F_t)^2 \leq q_{A(F_t)}(k)$$

where for each $A \in \mathcal{A}$,

$$q_A(k) = \begin{cases} \min\left(2^{c_A}, \left(\frac{4c_A}{k}\right)^k\right) & \text{if } k \le c_A \\ 2^{c_A} & \text{otherwise} \end{cases}$$

(note the different constants from q in the first line). We will consider two cases, based on the value of $k = |z|$.

Case 1: $c < 2k$ For each $A \in \mathcal{A}$, $q_A(k) \le 2^{c_A}$ and so $p_A q_A(k) \le 1$. Therefore,

$$\begin{aligned} \sum_{\substack{z \in \{0,1\}^{t-s} \\ z \text{ a union of } k \text{ paths}}} \mathop{\mathbb{E}}_{\mathbf{X}_{[s+1:t]}} \left[\widetilde{\mathbf{r}}(z; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{X}_{\le s} = y, \mathbf{B}_t = B_t\right] &\le \sum_{A \in \mathcal{A}} p_A q_A(k) \\ &\le |\mathcal{A}| \\ &\le 2^c \\ &\le q(k) \end{aligned}$$

as if $k \ge c$, $q(k) = 2^c$, and if $k \in (c/2, c]$, $q(k) \ge 8^{c/2} \ge 2^c$.

Case 2: $c \ge 2k$ Let $\mathcal{A} = \mathcal{A}^+ \cup \mathcal{A}^-$, with $\mathcal{A}^+$ containing all A in $\mathcal{A}$ such that $c_A > c$ and $\mathcal{A}^-$ containing everything else. Then $q_A(k) \le \left(\frac{4c}{k}\right)^k$ for each A in $\mathcal{A}^-$ and so

$$\sum_{a \in A^-} p_A q_A(k) \le \left(\frac{4c}{k}\right)^k.$$

At the same time for each $A \in A^+$, $c_A > c \ge 2k$, so

$$\begin{aligned} p_A q_A &= \left(\frac{4c_A}{k}\right)^k 2^{-c_A} \\ &\le \left(\frac{4c}{k}\right)^k 2^{-c} \end{aligned}$$

as for $x \ge 2k$, $\frac{d}{dx}(x^k 2^{-x}) = (k - x)x^{k-1}2^{-x} < 0$.

Therefore,

$$\begin{aligned} \sum_{\substack{z \in \{0,1\}^{t-s} \\ z \text{ a union of } k \text{ paths}}} \mathop{\mathbb{E}}_{\mathbf{X}_{[s+1:t]}} \left[\widetilde{\mathbf{r}}(z; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{X}_{\le s} = y, \mathbf{B}_t = B_t\right] &\le \sum_{A \in \mathcal{A}^+} p_A q_A(k) + \sum_{A \in \mathcal{A}^-} p_A q_A(k) \\ &\le \left(\frac{4c}{k}\right)^k + |\mathcal{A}| \left(\frac{4c}{k}\right)^k 2^{-c} \\ &\le 2\left(\frac{4c}{k}\right)^k \\ &\le q(k) \end{aligned}$$

and so the result follows. □

We now give a proof of Lemma 3.4.

Proof of Lemma 3.4: By Lemma 3.5, for every $t \in T$, every $s < t$, and every $z \in \{0,1\}^t$, we have

$$\widetilde{\mathbf{F}}_t(z) = \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s}) \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)\Big|\mathbf{F}_t, \mathbf{B}_t\Big].$$

As exactly one extension event $\mathbf{Grow}(z,s,t)$ will hold, we have

$$\widetilde{\mathbf{F}}_t(z) = \sum_{s=1}^{t-1} \mathbf{Grow}(z,s,t) \cdot \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s}) \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)\Big|\mathbf{F}_t, \mathbf{B}_t\Big],$$

where we condition on $\mathbf{B}_t$ on both sides, so $\mathbf{Grow}(z,s,t)$ is well-defined. Since the above sum contains only one nonzero term, we have by Jensen's inequality

$$\widetilde{\mathbf{F}}_t(z)^2 \leq \sum_{s=1}^{t-1} \mathbf{Grow}(z,s,t) \cdot \mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s})^2 \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{F}_t, \mathbf{B}_t\Big]. \tag{19}$$

Summing (19) over all z such that $z \sim_t \beta$, taking expectations over $\mathbf{X}$ and noting that for every $s \in T$ such that $\mathbf{Grow}(z,s,t) = 1$ one has $z_{\leq s} \sim_s \alpha$ for some $\alpha \in \beta - 1$ (recall Definition 4.11)), we get

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{X}}\big[\mathbf{H}^t_\beta | \mathbf{B}_t\big] &= \mathop{\mathbb{E}}_{\mathbf{X}}\Bigg[\sum_{z:z\sim_t\beta} \widetilde{\mathbf{F}}_t(z)^2 \Bigg| \mathbf{B}_t\Bigg] \\
&= \sum_{s=1}^{t-1} \sum_{z:z\sim_t\beta} \mathbf{Grow}(z,s,t) \cdot \mathop{\mathbb{E}}_{\mathbf{F}_t}\Bigg[\mathop{\mathbb{E}}_{\mathbf{F}_s}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s})^2 \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)^2 \Big| \mathbf{F}_t, \mathbf{B}_t\Big] \Bigg| \mathbf{B}_t\Bigg] \\
&= \sum_{s=1}^{t-1} \sum_{z:z\sim_t\beta} \mathbf{Grow}(z,s,t) \cdot \mathop{\mathbb{E}}_{\mathbf{F}_s,\mathbf{F}_t}\Big[\widetilde{\mathbf{F}}_s(z_{\leq s})^2 \cdot \widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)^2 \Big| \mathbf{B}_t\Big] \\
&= \sum_{s=1}^{t-1} \sum_{z:z\sim_t\beta} \mathbf{Grow}(z,s,t) \cdot \mathop{\mathbb{E}}_{\mathbf{F}_s}\Bigg[\widetilde{\mathbf{F}}_s(z_{\leq s})^2 \cdot \mathop{\mathbb{E}}_{\mathbf{F}_t}\big[\widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)^2 \big| \mathbf{F}_s, \mathbf{B}_t\big] \Bigg| \mathbf{B}_t\Bigg]
\end{aligned} \tag{20}$$

For every $\beta \in \mathbb{Z}_+^{\{\}}$ and $\alpha \in \beta - 1$ we let $\beta^{-\alpha}$ be given by setting $\beta[1] = \alpha[1]$. Splitting each $z \sim_t \beta$ into the co-ordinates before $\mathbf{Grow}(z,s,t)$ holds, namely $z_{\leq s}$, and those after, namely $z_{[s+1:t]}$ (note that the latter all corresponding to isolated edges in $\mathbf{O}_t$), we get

$$\begin{aligned}
&\sum_{z:z\sim_t\beta} \mathbf{Grow}(z,s,t) \cdot \mathop{\mathbb{E}}_{\mathbf{F}_s}\Bigg[\widetilde{\mathbf{F}}_s(z_{\leq s})^2 \cdot \mathop{\mathbb{E}}_{\mathbf{F}_t}\big[\widetilde{\mathbf{r}}(z_{[s+1:t]}; \mathbf{F}_s, \mathbf{F}_t)^2 \big| \mathbf{F}_s, \mathbf{B}_t\big] \Bigg| \mathbf{B}_t\Bigg] \\
&\leq \sum_{a=0}^{|\beta|} \sum_{\substack{\alpha\in\beta-1,\\ |\beta-\alpha|=a}} \sum_{\substack{r\in\{0,1\}^s:\\ r\sim_s\alpha \\ r\cdot 1\sim_{s+1}\beta^{-\alpha}}} \mathop{\mathbb{E}}_{\mathbf{F}_s}\Bigg[\widetilde{\mathbf{F}}_s(r)^2 \sum_{\substack{p\in\{0,1\}^{t-s-1},|p|=a,\\ r\cdot 1\cdot p\sim_t\beta}} \mathop{\mathbb{E}}_{\mathbf{X}}\big[\widetilde{\mathbf{r}}(1\cdot p; \mathbf{F}_s, \mathbf{F}_t)^2 \big| \mathbf{F}_s, \mathbf{B}_t\big] \Bigg| \mathbf{B}_t\Bigg]
\end{aligned} \tag{21}$$

Using Lemma 4.33 we upper bound the sum in the expectation in (21) by

$$\sum_{\substack{p\in\{0,1\}^{t-s-1},|p|=a,\\ r\cdot 1\cdot p\sim_t\beta}} \mathop{\mathbb{E}}_{\mathbf{X}}\big[\widetilde{\mathbf{r}}(1\cdot p; \mathbf{F}_s, \mathbf{F}_t)^2 | \mathbf{F}_s, \mathbf{B}_t\big] \leq \sum_{\substack{p\in\{0,1\}^{[s+2:t]}\\ |p|=a}} \mathop{\mathbb{E}}_{\mathbf{X}}\big[\widetilde{\mathbf{r}}(1\cdot p; \mathbf{F}_s, \mathbf{F}_t)^2 \big| \mathbf{F}_s, \mathbf{B}_t\big] \leq q(a). \tag{22}$$

Indeed, note that the bound of Lemma 4.33 holds conditioned on *any* value of $X_{\leq s}$, and thus certainly holds conditioned on $\mathbf{F}_s$. Substituting the above into (20), we get

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{X}}\left[\mathbf{H}_\beta^t\middle|\mathbf{B}_t\right] &\leq \sum_{s=1}^{t-1}\sum_{a=0}^{|\beta|} q(a) \sum_{\substack{\alpha\in\beta-1,\\ |\beta-\alpha|=a}} \sum_{\substack{r\in\{0,1\}^s:\\ r\sim_s\alpha\\ r\cdot 1\sim_{s+1}\beta^{-\alpha}}} \mathop{\mathbb{E}}_{\mathbf{F}_s}\left[\widetilde{\mathbf{F}}_s(r)^2\middle|\mathbf{B}_t\right] \\ &= \sum_{s=1}^{t-1}\sum_{a=0}^{|\beta|} q(a) \sum_{\substack{\alpha\in\beta-1,\\ |\beta-\alpha|=a}} \sum_{\substack{r\in\{0,1\}^s:\\ r\sim_s\alpha\\ r\cdot 1\sim_{s+1}\beta^{-\alpha}}} \mathop{\mathbb{E}}_{\mathbf{X}}\left[\widetilde{\mathbf{F}}_s(r)^2\middle|\mathbf{B}_s\right]\end{aligned} \tag{23}$$

as $\mathbf{F}_s$ is independent of $\mathbf{B}_{s+1:t}$. Note that the condition $r\sim_s\alpha$ and $r\cdot 1\sim_{s+1}\beta^{-\alpha}$ above makes sense since we condition on $\mathbf{B}_t$ on both sides. Taking expectations with respect to $\mathbf{B}_t$ of both sides of (23), we get

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\beta^t\right] &\leq \sum_{s=1}^{t-1}\sum_{a=0}^{|\beta|} q(a) \sum_{\substack{\alpha\in\beta-1,\\ |\beta-\alpha|=a}} \mathop{\mathbb{E}}_{\mathbf{B}_s}\left[\mathop{\mathbb{E}}_{\mathbf{B}_{s+1:t}}\left[\sum_{\substack{r\in\{0,1\}^s:\\ r\sim_s\alpha\\ r\cdot 1\sim_{s+1}\beta^{-\alpha}}} \mathop{\mathbb{E}}_{\mathbf{X}}\left[\widetilde{\mathbf{F}}_s(z_{\leq s})^2\middle|\mathbf{B}_s\right]\middle|\mathbf{B}_s\right]\right] \\ &\leq \sum_{s=1}^{t-1}\sum_{a=0}^{|\beta|} q(a) \sum_{\substack{\alpha\in\beta-1,\\ |\beta-\alpha|=a}} \mathop{\mathbb{E}}_{\mathbf{B}_s,\mathbf{F}_s}\left[\sum_{r\in\{0,1\}^s:\ r\sim_s\alpha} \Pr_{\mathbf{b}_{s+1}}\left[r\cdot 1\sim_{s+1}\beta^{-\alpha}|\mathbf{B}_s\right]\widetilde{\mathbf{F}}_s(z_{\leq s})^2\middle|\mathbf{B}_s\right] \\ &= \sum_{s=1}^{t-1}\sum_{\alpha\in\beta-1} q(|\beta-\alpha|) \sum_{z:z\sim_s\alpha} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\widetilde{\mathbf{F}}_s(z_{\leq s})^2\cdot p_s(\alpha,\beta,\mathbf{B}_s)\right] \\ &= \sum_{s=1}^{t-1}\sum_{\alpha\in\beta-1} q(|\beta-\alpha|)\cdot \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}_\alpha^s\cdot p_s(\alpha,\beta,\mathbf{B}_s)\right]\end{aligned}$$

as required.

For the base case where β is all size 1 components, and as $\mathbf{F}_0$ is always the trivial function on $\emptyset$, we apply Lemma 3.5 to get

$$\begin{aligned}\mathbf{H}_\beta^t &\leq \sum_{z:z\sim_t\beta} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\widetilde{\mathbf{r}}(z,\mathbf{F}_0,\mathbf{F}_t)^2\right] \\ &\leq q(|\beta|)\end{aligned}$$

by Lemma 4.33. □

4.8 Main lemma (Lemma 3.1)

The main result of this section is Lemma 3.1, restated here for convenience of the reader:

Lemma 3.1. *For all $\varepsilon>0$, there is a $D>0$ depending only on ε such that, if $c<\min(\ell^{\ell/D},n^{1-\varepsilon})$ and $D<\ell<D^{-1}\log n$,*

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\mathbf{H}_{\{\ell\}}^n\right]\leq\varepsilon.$$

Before proving the lemma, we introduce a useful definition of potential function p below, prove a useful property (Claim 4.36), as well as establish two technical lemmas. The first lemma, namely Lemma 4.37 below, bounds the expected evolution of $\mathbf{H}^t_\beta$ until almost the end of the stream. Lemma 4.38 provides a useful combinatorial characterization of the board towards the end of the stream.

Definition 4.35. *For every $\beta \in \mathbb{Z}_+^{\{\}}$ define $\nu(\beta) = \sum_{i\in\beta} \lceil \frac{i}{2} \rceil$.*

We will need the following properties of $\nu(\beta)$:

Claim 4.36. *For every $\beta \in \mathbb{Z}_+^{\{\}}$ and every $\alpha \in \beta - 1$ one has $\nu(\alpha) \le \nu(\beta)$.*

Proof. We first consider **extensions**, i.e. suppose that β can be obtained from α by incrementing one of the elements in α by 1, followed by possibly adding an arbitrary number of 1's to α. It suffices to note that every added 1 contributes 0 or 1 to $\nu(\beta) - \nu(\alpha)$, and the increment similarly contributes either 1 or 0.

We now consider **merges**, i.e. suppose that β is obtained from α by replacing two elements $a, b \in \alpha$ with $a + b + 1$, followed by possibly adding an arbitrary number of 1's to α. As before, every added 1 contributes 0 or 1 to $\nu(\beta) - \nu(\alpha)$. We now verify that $\lceil \frac{a+b+1}{2} \rceil \ge \lceil \frac{a}{2} \rceil + \lceil \frac{b}{2} \rceil$ for all non-negative integers a, b. If one of a and b is even (suppose it is a), we get

$$\left\lceil \frac{a+b+1}{2} \right\rceil = \left\lceil \frac{a}{2} \right\rceil + \left\lceil \frac{b+1}{2} \right\rceil \ge \left\lceil \frac{a}{2} \right\rceil + \left\lceil \frac{b}{2} \right\rceil .$$

If both are odd, then

$$\left\lceil \frac{a+b+1}{2} \right\rceil = \left\lceil \frac{a+b+2}{2} \right\rceil = \left\lceil \frac{a+1}{2} \right\rceil + \left\lceil \frac{b+1}{2} \right\rceil \ge \left\lceil \frac{a}{2} \right\rceil + \left\lceil \frac{b}{2} \right\rceil ,$$

as required. □

Lemma 4.37. *For every $\beta \in \mathbb{Z}_+^{\{\}}$ such that $|\beta|_* < \ell$, $t \in [n - 2^\ell \sqrt{n \cdot c} - Qn^{5/6} 2^{2\ell} \log n]$ one has*

$$\mathbb{E}\big[\mathbf{H}^t_\beta\big] \le \left(\prod_{j\in\beta} \frac{1}{j!} \right) \cdot Q^{|\beta|_*} \cdot \left(\frac{t}{n} \right)^{|\beta|_* - \nu(\beta)} \cdot c^{|\beta|} + Qn^{|\beta|_* - \ell}$$

for some absolute constant $Q > 1$.

Proof. We will proceed by induction on $|\beta|_*$. Recall that as per Definition 4.12 we denote the weight of β by

$$|\beta|_* := \sum_{i\in\beta} i$$

and the size of β (i.e., the number of elements in β) by $|\beta|$. Applying Lemma 3.4. For our base case, note that if $|\beta|_* = 1$ then $\beta = \{1\}$, and so,

$$\mathbf{H}^t_\beta \le q(|\beta|) \le (8c)^{|\beta|}$$

for all t. Now suppose $|\beta|_* > 1$ and the result holds for all α with $|\alpha|_* < |\beta|_*$. If every element of β is 1, we again have

$$\mathbf{H}^t_\beta \le q(|\beta|) \le (8c)^{|\beta|}$$

for all t. Otherwise, we have

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}^t_\beta\right] \le \sum_{s=1}^{t} \sum_{\alpha\in\beta-1} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\left[\mathbf{H}^s_\alpha \cdot q(|\beta-\alpha|)\cdot p(\alpha,\beta,\mathbf{B}_s)\right] \tag{24}$$

with every term of the outer sum being zero if β cannot be made from α by either an extension or a merge (with possible addition of ones in either case). Let A^e represent the set of α that can become β through extension (and adding ones) and A^m represent the set of α that can become β through merges (and adding ones). Let $\mathcal{E}_s$ denote the event from Lemma 4.18 such that

$$p_s(\alpha,\beta,B_t) \le \begin{cases} \frac{O(\alpha[a])}{n} & \text{if } \alpha\to\beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a]\cdot\alpha[b])}{(n-s)^2} & \text{if } \alpha\to\beta \text{ is a merge of paths of size } a \text{ and } b. \end{cases}$$

for each $B_s\in\mathcal{E}_s$. Substituting this bound in (24) and using the fact that $\mathbf{H}^s_\alpha$ is always non-negative, we get

$$\begin{aligned}\mathbb{E}\left[\mathbf{H}^t_\beta\right] \le{}& \sum_{\alpha\in A^e}\sum_{s=0}^{t} q(|\beta-\alpha|)\,\mathbb{E}[\mathbf{H}^s_\alpha]p_s(\alpha,\beta) + \sum_{\alpha\in A^m}\sum_{s=0}^{t} q(|\beta-\alpha|)\,\mathbb{E}[\mathbf{H}^s_\alpha]p_s(\alpha,\beta)\\ &+ \sum_{\alpha\in\beta-1}\sum_{s=0}^{t} q(|\beta-\alpha|)\,\mathbb{E}\left[\mathbf{H}^s_\alpha\middle|\overline{\mathcal{E}_s}\right]\Pr\left[\overline{\mathcal{E}_s}\right]\end{aligned} \tag{25}$$

where

$$p_s(\alpha,\beta) = \begin{cases} \frac{O(\alpha[a])}{n} & \text{if } \alpha\to\beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a]\cdot\alpha[b])}{(n-s)^2} & \text{if } \alpha\to\beta \text{ is a merge of paths of size } a \text{ and } b. \end{cases} \tag{26}$$

We will proceed to bound each of these three terms—the contribution from extensions, merges, and $\overline{\mathcal{E}_t}$—in turn.

Bounding the contribution of extensions. Let $o=\beta[1]$. The α that can be made into β by an extension are given by removing up to o ones from β and then choosing one non-one element of β to decrement. For each $x\in\{0,\dots,o\}$ and $y\in[n]\setminus\{1\}$ such that $\beta[j]>0$, let β' be β with x ones removed and one y replaced with $y-1$.

By our inductive hypothesis,

$$\begin{aligned}\mathbb{E}\left[\mathbf{H}^s_{\beta'}\right] &\le \left(\prod_{j\in\beta'}\frac{1}{j!}\right)\cdot Q^{|\beta'|_*}\cdot\left(\frac{s}{n}\right)^{|\beta'|_*-\nu(\beta')}\cdot c^{|\beta'|} + Qn^{|\beta'|_*-\ell}\\ &\le \frac{y}{Q^{x+1}c^x}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1}\cdot c^{|\beta|} + n^{-x-1}Qn^{|\beta|_*-\ell},\end{aligned}$$

where we used the fact that $|\beta'|_* = |\beta|_* - x - 1$ and the fact that $\nu(\beta)\ge\nu(\beta')$ by Claim 4.36. Using (26), we can therefore bound the contribution of extensions that take β' to β to the sum by

$$\sum_{s=1}^{t-1} q(x)\,\mathbb{E}\left[\mathbf{H}^s_{\beta'}\right]p_s(\beta',\beta) \le \frac{O(q(x)y\alpha[y])}{Qc^x}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\frac{1}{n}\sum_{s=1}^{t-1}\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1} \tag{27}$$

$$+ O(n^{-x-1})q(x)\alpha[y]Qn^{|\beta|_*-\ell}\frac{t}{n} \tag{28}$$

$$\tag{29}$$

Since

$$\frac{1}{n}\sum_{s=1}^{t-1}\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1} \le \frac{1}{n}\int_2^t \left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1} ds \le \frac{1}{|\beta|_*-\nu(\beta)}\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)},$$

we upper bound the lhs in (27) by

$$\sum_{s=1}^{t-1} q(x)\,\mathbb{E}\big[\mathbf{H}^s_{\beta'}\big]p_s(\beta',\beta) \le \frac{O(q(x)y\alpha[y])}{Qc^x(|\beta|_*-\nu(\beta))}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)}\cdot c^{|\beta|} + O(n^{-x-1})q(x)Q\ell n^{|\beta|_*-\ell}.$$

Summing the above over $x\in\{0,1,\ldots,o\}$ and $y>1$ such that $\beta[y]>0$, we get

$$\sum_{\alpha\in A^e}\sum_{s=1}^{t}\mathbb{E}[\mathbf{H}^s_\alpha]p_t(\alpha,\beta) \le O(Q^{-1})\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)}\cdot c^{|\beta|}\left(\sum_{x=0}^{o}\frac{q(x)}{c^x}\right)\sum_{\substack{y\in\beta:\\ y>1}}\frac{y}{|\beta|_*-\nu(\beta)} \tag{30}$$

$$+ O(Q)n^{|\beta|_*-\ell}\frac{|\beta|\ell}{n}\sum_{x=0}^{o}q(x)n^{-x}. \tag{31}$$

Recall that β is a multiset of integers, and the above sum over $y\in\beta$ goes over all elements of β, taking multiplicities into account.

We now bound the multiplicative terms in (30). For the first multiplicative terms we have

$$\begin{aligned}\sum_{x=0}^{o}q(x)c^{-x} &\le \sum_{x=0}^{c}\left(\frac{8x}{x}\right)^k c^{-x} + \sum_{x=c}^{\infty}2^c c^{-x}\\ &\le \sum_{x=0}^{\infty}\left(\frac{O(1)}{x}\right)^x + O(1)\\ &= O(1).\end{aligned} \tag{32}$$

For the second term we have

$$\begin{aligned}\sum_{\substack{y\in\beta:\\ y>1}}\frac{y}{|\beta|_*-\nu(\beta)} &= \frac{\sum_{\substack{y\in\beta:\ y\\ y>1}}}{\sum_{y\in\beta}\left\lfloor\frac{y}{2}\right\rfloor}\\ &\le \frac{\sum_{\substack{y\in\beta:\ y\\ y>1}}}{\sum_{\substack{y\in\beta:\\ y>1}}\left\lfloor\frac{y}{2}\right\rfloor}\\ &\le 3.\end{aligned}$$

Finally, for the last additive term in (31) we have

$$\begin{aligned}\sum_{x=0}^{o}q(x)n^{-x} &\le \sum_{x=0}^{o}q(x)c^{-x}\\ &= O(1)\end{aligned} \tag{33}$$

by (32) and the fact that $c \leq n$ whenever the interval of permitted t's is non-empty, provided $Q \geq 1$. Therefore,

$$\begin{aligned} O(Q)n^{|\beta|_*-\ell}\frac{|\beta|\ell}{n}\sum_{x=0}^{o} q(x)n^{-x} &\leq O(Q)n^{|\beta|_*-\ell}\frac{2|\beta|\ell}{n} \\ &\leq \frac{1}{3}Qn^{|\beta|_*-\ell} \end{aligned}$$

as $|\beta|\ell < n/D$ for any constant $D > 0$ whenever the interval of permitted t's is non-empty, provided Q is chosen to be sufficiently large. We therefore have

$$\sum_{\alpha\in A^e}\sum_{s=1}^{t}\mathbb{E}[\mathbf{H}^s_\alpha]p_t(\alpha,\beta) \leq \frac{1}{2}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)} + \frac{Q}{3}n^{|\beta|_*-\ell}$$

provided Q is chosen to be sufficiently large.

Bounding the contribution of merges. Let $o = \beta[1]$. The α that can be made into β by a merge are given by removing up to o ones from β and then choosing a $y > 2$ in β to replace with a, b, where $a + b + 1 = y$.

For each $x \in \{0, \ldots, o\}$, $y > 2$ such that $\beta[y] > 0$, and a, b such that $a+b+1 = y$, let $\beta_x^{y\to a,b}$ be β with x ones removed and one y replaced with a, b. Note that $|\beta_x^{y\to a,b}|_* = |\beta|_* - 1 - x \leq |\beta|_* - 1$, $|\beta_x^{y\to a,b}| = |\beta| - x + 1$ and that $\nu(\beta) \geq \nu(\beta')$ by Claim 4.36. By our inductive hypothesis we thus have

$$\begin{aligned} \mathbb{E}\left[\mathbf{H}^s_{\beta_x^{y\to a,b}}\right] &\leq c^{1-x}\frac{y!}{a!b!}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*-1}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1}\cdot c^{|\beta|} + n^{-1-x}Qn^{|\beta|_*-\ell} \\ &= c^{1-x}(b+1)\binom{y}{a}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*-1}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1}\cdot c^{|\beta|} + n^{-1-x}Qn^{|\beta|_*-\ell} \end{aligned}$$

and so we can bound the contribution of a, b to $a + b + 1$ merges by

$$\begin{aligned} \sum_{s=1}^{t-1} q(x)\,\mathbb{E}\left[\mathbf{H}^s_{\beta_x^{y\to a,b}}\right]p_s(\beta_x^{y\to a,b},\beta) &\leq O(c^{1-x})q(x)b\binom{y}{a}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*-1}\cdot c^{|\beta|}\cdot\frac{\alpha[a]\cdot\alpha[b]}{(n-t)^2} \\ &\quad\cdot\sum_{s=1}^{t-1}\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1} + O(tn^{-1-x})q(x)Qn^{|\beta|_*-\ell}\frac{\alpha[a]\cdot\alpha[b]}{(n-t)^2} \\ &\leq \frac{O(c\cdot n\cdot\ell^2)}{Q(n-t)^2}q(x)c^{-x}b\binom{y}{a}\left(\prod_{j\in\beta}\frac{1}{j!}\right)Q^{|\beta|_*}c^{|\beta|}\left(\frac{t}{n}\right)^{|\beta|_*-\nu(\beta)} \\ &\quad + O(n^{-x})Qq(x)n^{|\beta|_*-\ell}\frac{\ell^2}{(n-t)^2}, \end{aligned}$$

where we used the bound

$$\begin{aligned}\sum_{s=1}^{t-1}\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1} &\le \int_2^t \left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)-1} ds \\ &= \frac{n}{|\beta|_*-\nu(\beta)}\left(\frac{t}{n}\right)^{|\beta|_*-\nu(\beta)} \\ &\le n\left(\frac{t}{n}\right)^{|\beta|_*-\nu(\beta)},\end{aligned}$$

since $|\beta|_* - \nu(\beta) = \sum_{i\in\beta}(i - \lceil i/2\rceil) \ge 1$, as β contains at least one component of size more than 1 (the others are taken care of by the base case).

Now since $b \le \ell$ and $\sum_{a=1}^{y}\binom{y}{a} \le 2^y \le 2^\ell$ and $|\beta| < \ell$, we can sum over $y \in \beta$ and $x = 0, \dots, o$ to get

$$\begin{aligned}\sum_{\alpha\in A^m}\sum_{s=1}^{t-1} q(|\beta-\alpha|)\,\mathbb{E}[\mathbf{H}_\alpha^s]p_s(\alpha,\beta) &\le \frac{c\cdot n\cdot \ell^4 2^\ell}{Q(n-t)^2}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\left(\frac{t}{n}\right)^{|\beta|_*-\nu(\beta)}\left(\sum_{x=0}^{o} q(x)c^{-x}\right) \\ &\quad + \frac{\ell^4 2^\ell}{(n-t)^2} Q n^{|\beta|_*-\ell}\left(\sum_{x=0}^{o} q(x)n^{-x}\right) \\ &\le \frac{c\cdot n\cdot \ell^4 2^\ell}{Q(n-t)^2}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\left(\frac{t}{n}\right)^{|\beta|_*-\nu(\beta)} \\ &\quad + \frac{Q}{3}n^{|\beta|_*-\ell}\end{aligned}$$

by (32), (33), and the fact that, by the bound on t in the lemma statement, $\frac{\ell^4 2^\ell}{(n-t)^2}$ can be bounded above by any constant if Q is chosen to be large enough. Finally, as $n - t \ge 2^\ell\sqrt{c\cdot n}$,

$$\sum_{\alpha\in A^m}\sum_{s=1}^{t-1} q(|\beta-\alpha|)\,\mathbb{E}[\mathbf{H}_\alpha^s]p_s(\alpha,\beta) \le \frac{1}{2}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\left(\frac{s}{n}\right)^{|\beta|_*-\nu(\beta)} + \frac{Q}{3}n^{|\beta|_*-\ell}$$

provided Q is chosen to be large enough.

Bounding the contribution of the low probability event By Lemma 4.18, $\Pr\big[\overline{\mathcal{E}_s}\big] \le 1/n^{\ell+1}$ for each s. Then we have, using Lemma 4.34,

$$\begin{aligned}
\sum_{\alpha\in\beta-1}\sum_{s=1}^{t-1} q(|\beta-\alpha|)\,\mathbb{E}\big[\mathbf{H}_\alpha^s\big|\overline{\mathcal{E}_s}\big]\Pr\big[\overline{\mathcal{E}_s}\big] &\le \frac{1}{n^{\ell+1}}\sum_{s=1}^{t-1}\sum_{k=1}^{|\beta|} c^{|\beta|-k} \sum_{\substack{z\in\{0,1\}^s\\ z \text{ a collection of } k \text{ paths}}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\Big[\widetilde{\mathbf{F}}_s^2(z)\Big|\overline{\mathcal{E}_s}\Big] \\
&= \frac{1}{n^{\ell+1}}\sum_{s=1}^{t-1}\sum_{k=1}^{|\beta|} c^{|\beta|-k} \sum_{\substack{z\in\{0,1\}^s\\ z \text{ a collection of } k \text{ paths}}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_s}\big[\widetilde{\mathbf{r}}(z,\mathbf{F}_0,\mathbf{F}_s)\big|\overline{\mathcal{E}_s}\big] \\
&\le \frac{1}{n^{\ell+1}}\sum_{s=1}^{t-1}\sum_{k=1}^{|\beta|} c^{|\beta|-k} q(k) \\
&\le \frac{1}{n^{\ell}}\left(\sum_{k=1}^{c} c^{|\beta|-k}(c/k)^k + \sum_{k=c}^{\infty} c^{|\beta|-k}2^c\right) \\
&\le \frac{1}{n^{\ell}}\left(\sum_{k=1}^{\infty} k^{-k} + \sum_{k=c}^{\infty} c^{c-k}\right) \\
&= \frac{O\big(c^{|\beta|}\big)}{n^{\ell}} \\
&\le O\Big(n^{|\beta|_*-\ell}\Big) \\
&\le \frac{Q}{3}n^{|\beta|_*-\ell}
\end{aligned}$$

provided Q is chosen to be large enough (which in particular implies $c \le n$). Here we assumed c is at least 2—if it is 1 the result follows from $\sum_{s=1}^{t-1}\sum_{k=1}^{|\beta|} q(|\beta|-k)q(k) \le 4\ell^2$.

With these three bounds in hand, we return to equation (25).

$$\begin{aligned}
\mathbb{E}\big[\mathbf{H}_\beta^t\big] &\le \sum_{\alpha\in A^e}\sum_{s=0}^{t} q(|\beta-\alpha|)\,\mathbb{E}[\mathbf{H}_\alpha^s]p_s(\alpha,\beta) + \sum_{\alpha\in A^m}\sum_{s=0}^{t} q(|\beta-\alpha|)\,\mathbb{E}[\mathbf{H}_\alpha^s]p_s(\alpha,\beta) \\
&\quad + \sum_{\alpha\in\beta-1}\sum_{s=0}^{t} q(|\beta-\alpha|)\,\mathbb{E}\big[\mathbf{H}_\alpha^s\big|\overline{\mathcal{E}_s}\big]\Pr\big[\overline{\mathcal{E}_s}\big] \\
&\le \frac{1}{2}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\Big(\frac{s}{n}\Big)^{|\beta|_*-\nu(\beta)} + \frac{Q}{3}n^{|\beta|_*-\ell} \\
&\quad + \frac{1}{2}\left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\Big(\frac{s}{n}\Big)^{|\beta|_*-\nu(\beta)} + \frac{Q}{3}n^{|\beta|_*-\ell} \\
&\quad + \frac{Q}{3}n^{|\beta|_*-\ell} \\
&= \left(\prod_{j\in\beta}\frac{1}{j!}\right)\cdot Q^{|\beta|_*}\cdot c^{|\beta|}\cdot\Big(\frac{s}{n}\Big)^{|\beta|_*-\nu(\beta)} + Qn^{|\beta|_*-\ell}
\end{aligned}$$

□

Lemma 4.38. *For all $\varepsilon > 0$, with probability $1 - 1/n^2$ over $\mathbf{B}_n$, every cycle has at least $\ell - 3/\varepsilon$ edges present at time $n - n^{1-\varepsilon}/\ell$.*

Proof. The probability of any edge *not* being present at time t is $(n-t)/n = 1/\ell n^{\varepsilon}$. Moreover, these events are negatively associated, so for any cycle the probability that at least k edges are not present is at most $\ell^k/(\ell n^{\varepsilon})^k = n^{-\varepsilon k}$. So by setting $k = 3/\varepsilon$ and taking a union bound over all n/ℓ cycles the result follows. □

Proof of Lemma 3.1: We will assume $\varepsilon \le 1/24$ (as if not, it will suffice to use the Q that would be chosen for $\varepsilon = 1/24$). Let $\varepsilon' = 2\varepsilon$, and let D be chosen to be large enough that

$$2^{\ell}\sqrt{n \cdot c} + Qn^{5/6}2^{2\ell}\log n < n^{1-\varepsilon'}/\ell$$

where Q is the universal constant from Lemma 4.37. Our bounds on c and ℓ allow this to be done with D only depending on ε.

Let $t' = n - n^{1-\varepsilon'}/\ell$. Let $\mathcal{E}$ denote the event (over $\mathbf{B}_n$) that at time t' no cycle had more than $3/\varepsilon'$ edges missing. By Lemma 3.5, for all $z \in \{0,1\}^n$, $t \in T$,

$$\widetilde{\mathbf{F}}_n(z) = \mathop{\mathbb{E}}_{\mathbf{F}_t}\Big[\widetilde{\mathbf{F}}_t(z_{\le t}) \cdot \widetilde{\mathbf{r}}(z_{[t+1:n]}; \mathbf{F}_t, \mathbf{F}_n)\Big|\mathbf{F}_n, \mathbf{B}_n\Big]$$

and so, choosing any $B_n \in \mathcal{E}$,

$$\mathop{\mathbb{E}}_{\mathbf{X}}\Big[\widetilde{\mathbf{F}}_n(z)^2\Big|\mathbf{B}_n = B_n\Big]] \le \mathop{\mathbb{E}}_{\mathbf{X}}\Big[\widetilde{\mathbf{F}}_t(z)^2\Big|\mathbf{B}_n = B_n\Big].$$

Now, for each of the n/ℓ cycles present at time n, either at most $\ell - 3$ of its edges are present at time t' or there is a time $t < t'$ when the $(\ell-2)^{\text{th}}$ edge of the cycle arrives. Furthermore, this implies that there are fewer than $3/\varepsilon'$ different paths present in the cycle at time t'. We may therefore write

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{X}}\big[\mathbf{H}_{\{\ell\}}\big|\mathbf{B}_n = B_n\big] \le &\sum_{\substack{\alpha \in \mathbb{Z}_+^{\{\}} \\ \ell - 3/\varepsilon' \le |\alpha|_* \le \ell - 3 \\ |\alpha| \le 3/\varepsilon'}} \mathop{\mathbb{E}}_{\mathbf{X}}\Big[\mathbf{H}_{\alpha}^{t'}|\mathbf{B}_n = B_n\Big] \\ &+ \sum_{t=1}^{t'} \sum_{\substack{\beta \in \mathbb{Z}_+^{\{\}} \\ |\beta|_* = \ell - 2 \\ |\beta| \le 2}} \sum_{\alpha \in \beta - 1} \sum_{\substack{z \in \{0,1\}^t \\ z \sim_t \alpha \\ z \cdot 1 \sim_{t+1} \beta}} \mathop{\mathbb{E}}_{\mathbf{X}}\Big[\widetilde{\mathbf{F}}(z)^2|\mathbf{B}_n = B_n\Big].\end{aligned}$$

Now noting that $\mathbf{H}_{\ell} \le n/\ell$ with probability 1 and $\mathbf{H}_{\alpha} \ge 0$ for all α, we take expectation over

$\mathbf{B}_n$, getting

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\mathbf{H}_{\{\ell\}}\right] \leq & \sum_{\substack{\alpha\in\mathbb{Z}_+^{\{\}} \\ \ell-3/\varepsilon'\leq|\alpha|_*\leq\ell-3 \\ |\alpha|\leq 3/\varepsilon'}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_{t'}}\left[\mathbf{H}_\alpha^{t'}\middle|\mathcal{E}\right]\cdot\Pr[\mathcal{E}] \\ & + \sum_{t=1}^{t'}\sum_{\substack{\beta\in\mathbb{Z}_+^{\{\}} \\ |\beta|_*=\ell-2 \\ |\beta|\leq 2}}\sum_{\alpha\in\beta-1}\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\alpha^t\cdot p(\alpha,\beta,\mathbf{B}_t)\middle|\mathcal{E}\right]\Pr[\mathcal{E}] \\ & + n\Pr\left[\overline{\mathcal{E}}\right] \\ \leq & \sum_{\substack{\alpha\in\mathbb{Z}_+^{\{\}} \\ \ell-3/\varepsilon'\leq|\alpha|_*\leq\ell-3 \\ |\alpha|\leq 3/\varepsilon'}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_{t'}}\left[\mathbf{H}_\alpha^{t'}\right] + \sum_{t=1}^{t'}\sum_{\substack{\beta\in\mathbb{Z}_+^{\{\}} \\ |\beta|_*=\ell-2 \\ |\beta|\leq 2}}\sum_{\alpha\in\beta-1}\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\alpha^t\cdot p(\alpha,\beta,\mathbf{B}_t)\right] + 1/n.\end{aligned}$$

We now proceed to bound the first term in this sum. By Lemma 4.37, for all α, and for some universal constant Q,

$$\mathbb{E}\left[\mathbf{H}_\alpha^{t'}\right] \leq \left(\prod_{j\in\alpha}\frac{1}{j!}\right)\cdot Q^{|\alpha|_*}\cdot\left(\frac{t}{n}\right)^{|\alpha|_*-\nu(\alpha)}\cdot c^{|\alpha|} + Qn^{|\alpha|_*-\ell}$$

and for α with $\ell - 3/\varepsilon' \leq |\alpha|_* \leq \ell - 3$, all terms in the inside product are at most 1 and at least one is at most $\frac{1}{(\ell\cdot\varepsilon'/3-1)!}$, and so

$$\mathbb{E}\left[\mathbf{H}_\alpha^{t'}\right] \leq \frac{Q^\ell\cdot c^{3/\varepsilon'}}{(\ell\cdot\varepsilon'/3-1)!} + Qn^{-3}$$

and so as there are at most 2^ℓ distinct α with $|\alpha|_* \leq \ell$,

$$\sum_{\substack{\alpha\in\mathbb{Z}_+^{\{\}} \\ \ell-3/\varepsilon'\leq|\alpha|_*\leq\ell-3 \\ |\alpha|\leq 3/\varepsilon'}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_{t'}}\left[\mathbf{H}_\alpha^{t'}\right] \leq \frac{(2Q)^\ell\cdot c^{3/\varepsilon'}}{(\ell\cdot\varepsilon'/3-1)!} + Q2^\ell n^{-3} \leq \varepsilon/4$$

provided D is chosen to be a sufficiently large constant. Next, we bound the second term. By Lemma 4.18, there is an event $\mathcal{E}_t$ for each t such that, for any $B_t \in \mathcal{E}_t$,

$$p(\alpha,\beta,B_t) \leq \begin{cases}\frac{O(\alpha[a])}{n} & \text{if } \alpha\to\beta \text{ is an extension of a path of size } a \\ \frac{O(\alpha[a]\cdot\alpha[b])}{(n-t)^2} & \text{if } \alpha\to\beta \text{ is a merge of paths of size } a \text{ and } b.\end{cases}$$

and so for each α such that a β with $|\beta|_* = \ell - 2$ is reachable by an extension, the sum of $p(\alpha,\beta,B_t)$ over all such β is at most

$$\sum_{a\in\alpha}\frac{O(\alpha[a])}{n} = \frac{O(\ell)}{n}.$$

For α such that a β with $|\beta|_* = \ell - 2$ is reachable by a merge we use the fact that $t \leq n - n^{1-2\varepsilon} \leq n - \sqrt{n}$ and so $\frac{1}{(n-t)^2} \leq 1/n$ to obtain that the sum of $p(\alpha, \beta, B_t)$ over all such β is at most

$$\sum_{a,b \in \alpha} \frac{O(\alpha[a] \cdot \alpha[b])}{n} = \frac{O(\ell^2)}{n}$$

and so (again using the fact that $\mathbf{H}^t_\alpha$ is never negative)

$$\sum_{t=1}^{t'} \sum_{\substack{\beta \in \mathbb{Z}_+^{\{\}} \\ |\beta|_* = \ell-2 \\ |\beta| \leq 2}} \sum_{\alpha \in \beta - 1} \mathop{\mathbb{E}}_{\mathbf{X}, \mathbf{B}_t} \left[\mathbf{H}^t_\alpha \cdot p(\alpha, \beta, \mathbf{B}_t) \right] \leq \frac{O(\ell^2)}{n} \sum_{t=1}^{t'} \sum_{\substack{\alpha \in \mathbb{Z}_+^{\{\}} \\ |\alpha|_* = \ell-3 \\ |\alpha| \leq 3}} \mathop{\mathbb{E}}_{\mathbf{X}, \mathbf{B}_t} \left[\mathbf{H}^t_\alpha \right]$$
$$+ \ell \sum_{t=1}^{t'} \sum_{\substack{\alpha \in \mathbb{Z}_+^{\{\}} \\ |\alpha|_* = \ell-3 \\ |\alpha| \leq 3}} \mathop{\mathbb{E}}_{\mathbf{X}, \mathbf{B}_t} \left[\mathbf{H}^t_\alpha \middle| \overline{\mathcal{E}}_t \right] \Pr_{\mathbf{B}_t} \left[\overline{\mathcal{E}}_t \right]$$

with the second sum coming from the fact that there are at most ℓ different β that can be reached from any given α by an extension or merge. For the first of these sums, we note that for any α with $|\alpha|_* = \ell - 3$ and $|\alpha| \leq 3$, at least one path in α is length at least $\ell/3 - 1$ and so

$$\mathop{\mathbb{E}}_{\mathbf{X}, \mathbf{B}_t} \left[\mathbf{H}^t_\alpha \right] \leq \frac{D^\ell \cdot c^3}{(\ell/3 - 1)!} + Dn^{-3}$$

for all $t \leq t'$. So by summing over the $t' \leq n$ time steps and at most 2^ℓ choices of α, we get

$$\frac{O(\ell^2)}{n} \sum_{t=1}^{t'} \sum_{\substack{\alpha \in \mathbb{Z}_+^{\{\}} \\ |\alpha|_* = \ell-3 \\ |\alpha| \leq 3}} \mathop{\mathbb{E}}_{\mathbf{X}, \mathbf{B}_t} \left[\mathbf{H}^t_\alpha \right] \leq O(\ell^2) \frac{(2D)^\ell \cdot c^3}{(\ell/3 - 1)!} + \frac{O(\ell^2 D)}{n^3}$$
$$\leq \varepsilon/4$$

if D is chosen to be large enough. For the second sum, note that by Lemma 4.18, $\Pr\left[\overline{\mathcal{E}_t}\right] \leq 1/n^{\ell+1}$,

and so by applying Lemma 3.5 and Lemma 4.33,

$$\begin{aligned}
\ell \sum_{t=1}^{t'} \sum_{\substack{\alpha \in \mathbb{Z}_+^{\{\}} \\ |\alpha|_* = \ell-3 \\ |\alpha| \leq 3}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\alpha^t \middle| \overline{\mathcal{E}}_t\right] \Pr_{\mathbf{B}_t}\left[\overline{\mathcal{E}}_t\right] &\leq \frac{\ell}{n^{\ell+1}} \sum_{t=1}^{t'} \sum_{\substack{\alpha \in \mathbb{Z}_+^{\{\}} \\ |\alpha|_* = \ell-3 \\ |\alpha| \leq 3}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\mathbf{H}_\alpha^t \middle| \overline{\mathcal{E}}_t\right] \\
&\leq \frac{\ell}{n^{\ell+1}} \sum_{t=1}^{t'} \sum_{\substack{z \in \{0,1\}^t \\ z \text{ a union of 1, 2, or 3 components}}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\widetilde{\mathbf{F}}_t(z)^2 \middle| \overline{\mathcal{E}}_t\right] \\
&= \frac{\ell}{n^{\ell+1}} \sum_{t=1}^{t'} \sum_{\substack{z \in \{0,1\}^t \\ z \text{ a union of 1, 2, or 3 components}}} \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_t}\left[\widetilde{\mathbf{r}}(z; \mathbf{F}_0, \mathbf{F}_t)^2 \middle| \overline{\mathcal{E}}_t\right] \\
&\leq \frac{\ell}{n^{\ell+1}} \sum_{t=1}^{t'} (q(1) + q(2) + q(3)) \\
&\leq \frac{3\ell \cdot c^3}{n^\ell} \\
&\leq \varepsilon/4
\end{aligned}$$

provided D is chosen to be large enough. Finally, $1/n \leq \varepsilon/4$ if D is chosen to be large enough, giving us

$$\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\mathbf{H}_{\{\ell\}}\right] \leq \varepsilon$$

by summing these four bounds together. □

4.9 Proof of Theorem 4.1

Proof of Theorem 4.1: Suppose there was a protocol solving the distributional version of StreamingCycles(n, ℓ) with probability 2/3 and using $\min\left(\ell^{\ell/D}, n^{1-\varepsilon}\right)$ space, for some D to be chosen later. By the min-max theorem, there is a deterministic algorithm that, given a uniformly random instance of the communication problem, returns the identity of a cycle and its parity with probability 2/3. Let $\mathbf{F}_t$ be the random indicator function associated with the messages of this protocol, as in the discussion in the previous sections. Let $Z \in \{0,1\}^n$ be the random variable denoting the coefficient of the cycle whose parity the protocol returns if the final message is $\mathbf{F}_n$, and let $\mathbf{F} = \mathcal{F}_n$ to simplify notation. Let $\mathbf{A} = \{x \in \{0,1\}^n : \mathbf{F}(x) = 1\}$. Then

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\mathbf{H}_{\{\ell'\}}^n\right] &\geq \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\widetilde{\mathbf{F}}(Z)^2\right] \\
&= \mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\left(|\mathbf{A}|^{-1} \sum_{x \in \mathbf{A}} (-1)^{x \cdot Z}\right)^2\right].
\end{aligned}$$

Now, as $\mathbf{X}$ and $\mathbf{B}_n$ are both uniformly distributed, 2/3 of the possible pairs result in the algorithm giving a correct answer when given as input. Call these "good" pairs. Then

$$\mathop{\mathbb{E}}_{\mathbf{B}_n}\left[\Pr_{\mathbf{X}}[(X, \mathbf{B}_n) \text{ is good } | \mathbf{B}_n]\right] \geq 2/3$$

and so as, for each value of $\mathbf{B}_n$, the possible realizations A of $\mathbf{A}$ conditional on $\mathbf{B}_n$ partition the possible values of $\mathbf{X}$,

$$\mathop{\mathbb{E}}_{\mathbf{B}_n}\left[\Pr_{\mathbf{X}}[(x,\mathbf{B}_n) \text{ is good for at least } 7/12 \text{ of the } x \in \mathbf{A} \mid \mathbf{B}_n]\right] \geq 1/12.$$

If at least $7/12$ of the $x \in \mathbf{A}$ are good, in particular they all give the same value of $(-1)^{x\cdot Z}$, so

$$\left||\mathbf{A}|^{-1}\sum_{x\in\mathbf{A}}(-1)^{x\cdot Z}\right| \geq 1/6$$

and therefore

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{X},\mathbf{B}_n}\left[\mathbf{H}^n_{\{\ell'\}}\right] &\geq \mathop{\mathbb{E}}_{\mathbf{B}_n}\left[\mathop{\mathbb{E}}_{\mathbf{X}}\left[\left(|\mathbf{A}|^{-1}\sum_{x\in\mathbf{A}}(-1)^{x\cdot Z}\right)^2\middle|\mathbf{B}_n\right]\right]\\ &\geq \mathop{\mathbb{E}}_{\mathbf{B}_n}\left[1/12\cdot(1/6)^2\right]\\ &= 1/432\end{aligned}$$

which contradicts Lemma 3.1 if D is chosen to be large enough. □

4.10 Proof of Theorem 4.3

Proof of Theorem 4.3: Suppose we had an algorithm solving $(1,\ell)$ component estimation in the $(2,0)$-batch random order streaming model with probability at least $2/3$ and using $\min\left(\ell^{\Omega(\ell)}, n^{1-\varepsilon}\right)$ space. Let $\zeta = \text{poly}(n^{-1})$ denote the timestamp precision assumed by the algorithm (as per the discussion in Definition 5.3, it is assumed that time stamps are presented at this resolution, so they can be expressed in $O(\log n)$ bits). We design a protocol for StreamingCycles(n,ℓ) using only $O(\log n)$ extra bits of space.

Let V be the vertex set associated with the StreamingCycles(n,ℓ') problem, where $\ell' = \lfloor \ell/2 \rfloor$. We will use $V \times \{0,1\}$ as the vertex set of our component estimation input. When player i receives the i^{th} edge uv with bit label x, they give the component estimation algorithm two edges (in random order, say), $(u,0)(u,x)$ and $(u,1)(u,\overline{x})$ with timestamp $\mathbf{t}_i$. The timestamps $\mathbf{t}_i$ are given by drawing n uniform random variables with precision ζ and presenting them in ascending order. See Appendix A for how this can be done in $O(\log n)$ space in the stream, by storing only the most recent timestamp generated. They then send the state of the algorithm, along with the most recent timestamp generated, to player $i+1$. The final player reads off the vertex returned by the algorithm and the size of the component it is reported to contain, returning 0 if the size is 2ℓ and 1 if it is ℓ.

The stream ingested by the component collection algorithm will be a $(2,0)$-batch random order stream, as the edges given to us were in random order, and we are generating 2 edges for each one, and the batches are ordered by randomly drawn timestamps (we let the timestamps of the two edges be equal to the timestamp of the corresponding batch). Moreover, the graph it generates will, for each cycle in the communication problem, have one cycle of length $2\ell'$ (if the parity of the cycle in the communication problem is even) or two cycles of length ℓ' (otherwise).

So this graph has no component of size more than $2\ell' \leq \ell$, so with probability $2/3$ the component collection algorithm will correctly return a vertex in $V \times \{0,1\}$ and the corresponding component. The answer returned by the final player will therefore be a correct solution to the StreamingCyclesinstance. □

4.11 Proof of Theorem 4.7

Proof of Theorem 4.7: Suppose we had an algorithm solving $(k, s, 1/10, 1/10)$ random walk generation for some k and s in the $(2, 0)$-batch random order streaming model. Let $\zeta = \text{poly}(n^{-1})$ denote the timestamp precision assumed by the algorithm.

Let V be the vertex set associated with the StreamingCycles(n, ℓ') problem, where we choose $\ell' = \lfloor\sqrt{k/C}\rfloor$ for a large constant $C > 1$ for the first lower bound and $\ell' = \lfloor k/2\rfloor$ for the second lower bound. We will use $V \times \{0, 1\}$ as the vertex set of our component estimation input. When we receive the i^{th} edge uv with bit label x, we will give the component estimation algorithm two edges (in random order, say), $(u, 0)(u, x)$ and $(u, 1)(u, \overline{x})$ with timestamp $\mathbf{t}_i$. The timestamps $\mathbf{t}_i$ are given by drawing n uniform random variables with precision ζ and presenting them in ascending order. See Appendix A for how this can be done in $\mathrm{O}(\log n)$ space in the stream.

The stream ingested by the random walk generation algorithm will be a $(2, 0)$-batch random order stream, as the edges given to us were in random order, and we are generating 2 edges for each one, and the batches are ordered by randomly drawn timestamps (we let the timestamps of the two edges be equal to the timestamp of the corresponding batch). Moreover, the graph it generates will, for each cycle in the communication problem, have one cycle of length $2\ell'$ (if the parity of the cycle in the communication problem is even) or two cycles of length ℓ' (otherwise).

For the first lower bound, let $\ell' = \lfloor\sqrt{k/C}\rfloor$ for a sufficiently large absolute constant C, so that $k = C(\ell')^2$. Generate a 1/10-approximate sample of the walk of length k, with error at most $1/10$ in total variation distance. The walk loops around the cycle that it starts in with probability at least $2/3$ as long as the constant C is sufficiently large (indeed, this happens with probability at least $9/10$ for a true sample of the random walk of length k, and therefore at least with probability $(1 - 1/10)9/10 - 1/10 \geq 2/3$ for a 1/10-approximate sample with TVD error bounded by $1/10$). Let (v, b) denote the starting vertex. If the cycle is of length ℓ', output v and parity $= 0$. If the cycle is of length $2\ell'$, output v and parity $= 1$.

For the second lower bound, let $\ell' = \lfloor k/2\rfloor$ and run $C4^k$ 1/10-approximate random walks of length k started at uniformly random vertices, with error at most $1/10$ in total variation distance (for the joint distribution), for a sufficiently large constant $C > 0$. With probability at least $2/3$ at least one of the walks will loop around the cycle that it started in (it suffices for the walk to take a step in the same direction for k consecutive steps, which happens with probability $(1 - 1/10)2^{-k} \geq 4^{-k}$, so $C4^k$ independent repetitions suffice for one of the walks to cover the cycle with probability $9/10$; accounting for the at most $1/10$ error in total variation distance gives the result). Let (v, b) denote the starting vertex. If the cycle is of length ℓ, output v and parity $= 0$. If the cycle is of length $2\ell'$, output v and parity $= 1$.

Now, this is the hard instance of Theorem 4.3, with ℓ' larger than a sufficiently large absolute constant since k is. Thus, solving it requires $\min\{(\ell')^{\Omega(\ell')}, n^{0.99}\}$ space, setting the ε parameter to 0.01. Expressing this lower bound in terms of k, we now get that $(k, 1, 1/10)$-random walk generation requires at least $\min\{(\ell')^{\Omega(\ell')}, n^{0.99}\} = \min\{k^{\Omega(\sqrt{k})}, n^{0.99}\}$ space, and $(k, C4^k, 1/10)$-random walk generation requires at least $\min\{(\ell')^{\Omega(\ell')}, n^{0.99}\} = \min\{k^{\Omega(k)}, n^{0.99}\}$ space, as required. $\square$

5 Component Collection and Counting

5.1 Algorithmic Techniques

Many random-order streaming algorithms work, at a high level, in the following way:

- Sample some connected structure from the stream in an order-dependent way (for instance, "growing" a component by randomly choosing a vertex and then keeping every edge either incident to the vertex or to an already-sampled edge).
- Weight the sampled structures by the inverse of the prior probability of sampling them.

Such techniques make use of the fact that, in a fully random-order stream, the probability of any given set of edges arriving in any given order can be determined exactly and without any additional information about other edges in the graph. Now consider a stream divided into *known* batches of size b. Such techniques can be applied here by increasing the number of edges we sample by a factor of b:

- Whenever we would keep an edge, instead keep the entire *batch* containing that edge.
- When weighting a structure, adjust the prior probability of sampling it accordingly.

This is possible because we know the batches—when we see an edge we know which batch it was in, and the probability of a given set of batches arriving in any given order can still be determined exactly.

But what can we do when those batches are unknown? The key observation we apply is that, if the structures we are sampling are not too large and there are not too many of them, we can *guess* the batches and only err on "irrelevant" edges:

- Maintain a buffer of all edges with timestamps less than w before the present edge.
- Whenever we would keep an edge, instead keep every edge within w of it in either direction.
- When weighting structures, assume that any pair of edges that we kept and that had timestamps separated by at most $2w$ were in the same batch.

It is clear that, at least, when we keep an edge we will keep every other edge in the same batch. Furthermore, as long as our structures are not too large and there are not too many of them, any pair of edges in the same structure will, with probability w, either be in the same batch, or at least $2w$ away from each other. So as long as the total number of these edges is not much larger than $1/\sqrt{w}$, our batch guesses will probably be correct, and so we may proceed as if we were in the known-batch setting.

Component Collection and Counting. We apply the "batch guessing" technique described above to the problem of counting and collecting bounded-size components in a batch random order graph stream.

Similarly to the component counting strategy of [PS18], we approach this problem by first uniformly sampling a set of vertices, and then for each such vertex "growing" a connected subgraph

$\mathbf{D}_v$ containing v by keeping every edge with a path to v in our already-sampled edges, as long as $V(\mathbf{D}_v)$ never exceeds a given limit k. We then construct random variables $\mathbf{X}_v$ with the following properties:

1. $\mathbb{E}[\mathbf{X}_v \cdot \mathbb{1}(\mathbf{D}_v = K_v)] = 1$ whenever $V(K_v) \leq k$.
2. $\mathbb{E}[\mathbf{X}_v \cdot \mathbb{1}(\mathbf{D}_v \neq K_v)]$ is small.

Here $\mathbb{1}$ denotes the indicator function and K_v denotes the actual component containing G. This will allow us to approximately count the number of size $\leq k$ components, and therefore obtain a εn additive approximation to the component count if $k \geq 1/\varepsilon$. Furthermore, if a large enough fraction of the vertices of G are in size $\leq k$ components, it will let us sample a size k component (by sampling vertices v from G and then choosing a sampled subgraph $\mathbf{D}_v$ with probability proportional to $\mathbf{X}_v$). The total number of samples required will go as the inverse of the variance of $\mathbf{X}_v$.

The approach of [PS18] was based on defining a canonical spanning tree for each component, and then keeping components iff this spanning tree was collected first *and* it was entirely collected in the first λ fraction of the stream.

The weighting of this component in their estimator ($\mathbf{X}_v$ in our formulation) is then given by $(k-1)!/\lambda^{k-1}$ when the component is kept. They then make use of the fact that any subgraph H that is *not* the entirety of K_v will have a non-empty boundary, and therefore cannot be collected if any of the boundary edges arrive in the last $1-\lambda$ fraction of the stream (as they are incident to the spanning tree of H but are not in H). This means that if λ is small enough, $\mathbb{E}[\mathbf{X}_v \cdot \mathbb{1}(\mathbf{D}_v \neq K_v)]$ is small too. However, they need λ to be at most $k^{-\Theta(k^2)}$ (which is based on counting the number of possible "canonical" spanning trees with a given boundary size that can be rooted at b), which in turn means that the probability of successfully collecting a k-vertex component is at most $k^{-\Theta(k^3)}$, and so their algorithm needs $(1/\varepsilon)^{\mathrm{O}(1/\varepsilon^3)} \log n$ bits of space.

We instead define our weighting based on explicitly calculating the prior probability of collecting a component, and by a different combinatorial analysis of the number of subgraphs rooted at v with a given boundary size, are able to have $\lambda = 1/\operatorname{poly}(k)$, for a $(1/\varepsilon)^{\mathrm{O}(1/\varepsilon)} \operatorname{polylog} n$ space cost.

This also has the virtue of translating easily to the known-batches model: when growing a component, we keep an entire batch whenever we would keep an edge in it, and then calculate the prior collection probability of a component based on our knowledge of how it was collected.it was collected. We extend this to the *hidden* batch model through the "batch guessing" technique discussed earlier in this section.

Our main results in this section are Theorem 5.1 and Theorem 5.2 below.

Theorem 5.1 (Counting Components)**.** *For all $\varepsilon, \delta \in (0,1)$, there is a (b,w)-hidden batch random order streaming algorithm that achieves an εn additive approximation to $c(G)$ with $1-\delta$ probability, using*

$$(1/\varepsilon\delta)^{\mathrm{O}(1/\varepsilon)}(b+wm)\operatorname{polylog}(n)$$

bits of space.

Theorem 5.2 (Component Collection)**.** *For all $\delta \in (0,1)$, there is a (b,w)-hidden batch random order streaming algorithm such that, if at least a β fraction of the vertices of G are in components of size at most ℓ, returns a vertex in G and the component containing it with probability $1-\delta$ over its internal randomness and the order of the stream, using*

$$(\ell/\beta\delta)^{\mathrm{O}(\ell)}(b+wm)\operatorname{polylog} n$$

bits of space.

We start by stating a formal definition of the (b, w)-hidden batch stream model:

Definition 5.3 (Hidden-batch random order stream model; formal definition)**.** *In the (b, w)-hidden-batch random order stream model the edge set of the input graph $G = (V, E)$ is presented as follows:*

1. *An adversary partitions E into batches $\mathcal{B} = \{B_1, \ldots, B_q\}$ of size at most b, so that*

$$E = \bigcup_{B \in \mathcal{B}} B, \text{ and } |B| \leq b \text{ for all } B \in \mathcal{B}.$$

2. *Each batch $B \in \mathcal{B}$ is assigned an uniformly distributed starting time $\mathbf{t}_B \sim \mathcal{U}([0, 1])$.*
3. *For each $B \in \mathcal{B}$, the adversary assigns each edge $e \in B$ a timestamp $\mathbf{t}_e \in [\mathbf{t}_B, \mathbf{t}_B + w]$.*
4. *The items and timestamps $\{(e, \mathbf{t}_e) : e \in E\}$ are presented to the algorithm in non-decreasing order of $\mathbf{t}_e$, with the adversary breaking ties.*

While the batch timestamps are continuous random variables, we assume the adversary presents the edge timestamps with precision $\text{poly}(1/n)$, so each timestamp can be stored in $O(\log n)$ space. We do, however, require that the adversary present the edges in the order given by the edge timestamps, so in particular if $w = 0$ the adversary does not have the option of re-arranging batches that happen to fall within $\text{poly}(n^{-1})$ of each other.

When $b = 1$ and $w = 0$ this therefore collapses to standard random order streaming, as an algorithm presented with an ordinary random order stream can generated a sequence of appropriate timestamps "on the fly" (see Appendix A for details).

5.2 Notation

We will use $\sigma = (e, \mathbf{t}_e)_{e \in E}$ to denote a (b, w)-hidden batch stream, received in order of the timestamps $(\mathbf{t}_e)_{e \in E}$.

Throughout we will use K_v to refer to the component of G containing v, and $\mathcal{K}_v$ to refer to $\{B \cap K_v : B \in \mathcal{B}, B \cap K_v \neq \emptyset\}$, the partitioning of K_v into batches. When v is unambiguous we will sometimes drop the subscript.

We will use $\mathbb{I}(p)$ to denote the variable that is 1 if the predicate p holds and 0 otherwise.

5.3 Component Collection

5.3.1 The Real and the Idealized Algorithm

In this section we describe an algorithm CollectComponent(v, k) for collecting a subset $\mathbf{D}$ of K_v, the component containing v in G, along with a guess $\mathcal{D}$ of how $\mathbf{D}$ is partitioned into batches in $\mathcal{B}$. This algorithm will be a primitive in our component counting and collection algorithms.

To aid with the analysis of this algorithm, we will define a second "idealized" algorithm CollectComponentIdeal(v, k). This algorithm will be allowed to know how the stream is partitioned into batches B and have direct access to the batch timestamps $\mathbf{t}_B$. We will show that typically

both algorithms will have almost the same output, allowing us to analyze CollectComponent by way of CollectComponentIdeal.

CollectComponent will work as follows:

- Grow a subgraph $\mathbf{D}$ from a vertex v, keeping any edge that connects to v through a path in the edges already added to $\mathbf{D}$.
- Whenever a new edge e is added to the component as described above, ensure that all edges from the batch containing e are added to $\mathbf{D}$ as well, by adding every edge with a timestamp up to w before or after $\mathbf{t}_e$. In order to facilitate this, we keep a buffer W of all edges with timestamps up to w before the edge currently being processed.
- If the component of v in $\mathbf{D}$ ever has more than k vertices, return $\perp$.
- Otherwise, return the component of v in $\mathbf{D}$ (now discarding edges that do not connect to v), the timestamp of the last edge to add a new *vertex* to it, and a guess $\mathcal{D}$ at how it is partitioned into batches (based on assuming that any two edges that arrived within w of each other were in the same batch).

We now describe the algorithm formally – see Algorithm 1 below. The parameter s is used to track which edges should be added to $\mathbf{D}$ on the grounds of having timestamps up to w *after* an edge in the component containing v.

Algorithm 1 Collecting a component of size at most k in a hidden batch stream.

procedure CollectComponent(v, k)
  $W \leftarrow \emptyset$ ▷ Buffering edges from up to w ago.
  $\mathbf{D} \leftarrow (V, \emptyset)$ ▷ The subgraph we are building up.
  ▷ $\mathbf{D}$ has at most $d\binom{k}{2}$ edges so can be stored as a sparse graph.
  $\mathbf{T} \leftarrow 0$ ▷ The last time at which a new vertex was added to the component of v in $\mathbf{D}$.
  $s \leftarrow 0$ ▷ The last time an edge was added to the component of v in $\mathbf{D}$.
  for $(e, \mathbf{t}_e)$ from σ **do**
    Remove all edges from W with time stamps before $\mathbf{t}_e - w$.
    Add $(e, \mathbf{t}_e)$ to W.
    if an endpoint of e is connected to v in $\mathbf{D}$ **then**
      ▷ Add e to subgraph $\mathbf{D}$ together with all edges up to w before and after.
      $\mathbf{D} \leftarrow \mathbf{D} \cup \{f : (f, \mathbf{t}_f) \in W\}$
      ▷ Record timestamps for edges added to $\mathbf{D}$.
      $s \leftarrow \mathbf{t}_e$
    else if $\mathbf{t}_e \leq s + w$ **then**
      ▷ Add e since it might be in the same batch as f based on timestamp
      $\mathbf{D} \leftarrow \mathbf{D} \cup \{e\}$
    end if
    if the component of $\mathbf{D}$ containing v has more than k vertices **then**
      return $(\perp, \perp, \perp)$
    end if
  end for
  $\mathbf{D} \leftarrow$ the component containing v in $\mathbf{D}$.
  $\mathcal{D} \leftarrow$ the finest partition of $\mathbf{D}$ such that $\forall e, f \in \mathbf{D}, |\mathbf{t}_e - \mathbf{t}_f| \leq w \Rightarrow \exists P \in \mathcal{D}, e, f \in P$.
  return $(\mathbf{D}, \mathcal{D}, \mathbf{T})$ ▷ A subcomponent of K_v, a guess at how it is partitioned, and the last time a *vertex* was added to it.
end procedure

CollectComponentIdeal will be almost identical to CollectComponent, except now we will know the batches and so we will not need to guess which edges are in which batch. Let σ' denote the stream of batches and time stamps $(B, \mathbf{t}_B)$, ordered by $\mathbf{t}_B$.

Algorithm 2 Collecting a component of size at most k in a stream with known batches.

procedure CollectComponentIdeal(v, k)
  $\mathbf{S} \leftarrow (V, \emptyset)$ ▷ The subgraph we are building up.
  $\mathcal{S} \leftarrow \emptyset$ ▷ The set of batches intersecting $\mathbf{S}$.
  $\mathbf{U} \leftarrow 0$ ▷ The last time at which a new vertex was added to the component of v in $\mathbf{S}$.
  for $(B, \mathbf{t}_B)$ from σ' **do**
    if $\exists e \in B$ with at least one endpoint connected to v in $\mathbf{S}$ **then**
      if $\exists e \in B$ with exactly one endpoint connected to v in $\mathbf{S}$ **then**
        $\mathbf{U} \leftarrow \mathbf{t}_B$
      end if
      $\mathbf{S} \leftarrow \mathbf{S} \cup B$
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{B\}$
      if the component of $\mathbf{S}$ containing v has more than k vertices **then**
        return $(\perp, \perp, \perp)$
      end if
    end if
  end for
  $\mathbf{S} \leftarrow$ the component containing v in $\mathbf{S}$.
  for $B \in \mathcal{S}$ **do**
    $B \leftarrow B \cap \mathbf{S}$
  end for ▷ To match CollectComponent, we take all the batches we added to $\mathbf{S}$ and intersect them with the final value of $\mathbf{S}$ (that is, the component containing v in $\mathbf{S}$).
  return $(\mathbf{S}, \mathcal{S}, \mathbf{U})$ ▷ A subcomponent of K_v, its partitioning into batches, and the last time a *vertex* was added to it.
end procedure

Note that $\mathbf{S}$ uniquely determines $\mathcal{S}$, as $\mathcal{S}$ is the set of batches containing an edge from $\mathbf{S}$, with all edges not in $\mathbf{S}$ removed.

5.3.2 Correspondence to Ideal Algorithm

In this section, we prove that CollectComponent is a close approximation of CollectComponentIdeal, which we will then analyze in the subsequent section.

For these lemmas, we will need to consider the "boundary" batches of $\mathbf{S}$.

Definition 5.4. *The boundary batch set $\mathcal{F}$ of $\mathbf{S}$ consists of every batch $B \in \mathcal{B}$ such that at least one of the following holds:*

1. *$B \cap \mathbf{S} \neq \emptyset$ (i.e. $B \in \mathcal{S}$).*
2. *$\exists e \in B$ such that e is incident to either v or some edge in $\mathbf{S}$.*

Note that this is determined uniquely by $\mathbf{S}$.

First we show that the component collected by CollectComponent is always at least the component collected by CollectComponentIdeal.

Lemma 5.5. *Let $\mathbf{D}$, $\mathbf{S}$ be the subgraphs returned by* CollectComponent, CollectComponentIdeal, *respectively. Then*

$$\mathbf{D} \supseteq \mathbf{S}.$$

Proof. For $t \in [0,1]$, let $\mathbf{D}_t$, $\mathbf{S}_t$ be the states of $\mathbf{D}$ and $\mathbf{S}$, respectively, after every edge (for $\mathbf{D}$) or batch (for $\mathbf{S}$) with a timestamp no greater than t has been processed. It will therefore suffice to prove that $\mathbf{D}_{1+w} \supseteq \mathbf{S}_1$. (As ultimately $\mathbf{D}$, $\mathbf{S}$ will be the components of v in $\mathbf{D}_{1+w}$, $\mathbf{S}_1$, respectively, and taking the component containing v will preserve the superset relation.)

Fix any assignment of timestamps $(\mathbf{t}_B)_{B\in\mathcal{B}}$, $(\mathbf{t}_e)_{e\in E}$. We will prove the following by (strong) induction on the order of the time stamps $(\mathbf{t}_B)_{B\in\mathcal{B}}$: for all $B \in \mathcal{B}$, $\mathbf{D}_{\mathbf{t}_B+w} \supseteq \mathbf{S}_{\mathbf{t}_B}$. As no more edges are added to $\mathbf{S}$ after $\max\{\mathbf{t}_B : B \in \mathcal{B}\}$, this will suffice.

For any $B \in \mathcal{B}$, suppose that this holds for all B' with $\mathbf{t}_{B'} < \mathbf{t}_B$. Then we have $\mathbf{D}_{t+w} \supseteq \mathbf{S}_t$ for all $t < \mathbf{t}_B$, as if B', B'' are any pair of badges with no batches arriving between them, $\mathbf{S}_t$ is unchanged in the interval $[\mathbf{t}_{B'}, \mathbf{t}_{B''})$, while $\mathbf{D}_t$ is non-decreasing.

Now, if the batch B was not added to $\mathbf{S}$ at time $\mathbf{t}_B$, the result holds immediately. So suppose B was added. Then B contains at least one edge e that is incident to some edge f in a batch B' with $\mathbf{t}_{B'} < \mathbf{t}_B$. By our inductive hypothesis, $f \in \mathbf{D}_{\mathbf{t}_{B'}+w}$.

Suppose f was added to $\mathbf{D}$ at some time after $\mathbf{t}_B$. Then as $t_{B'} < t_B$, this time was was in $[t_B, t_B + w]$, and so every edge in $[t_B, t_B + w]$ will be added to $\mathbf{D}$ by the end of that window. So $B \subseteq \mathbf{D}_{t_B+w}$ and therefore $\mathbf{D}_{t_B+w} \supseteq \mathbf{S}_{t_B}$.

Now suppose instead f was added to $\mathbf{D}$ before $\mathbf{t}_B$. Then in particular $f \in \mathbf{D}_s$ for all $s < \mathbf{t}_e$. Therefore, when e arrives, it is added to $\mathbf{D}$ along with every edge that arrives within w of $\mathbf{t}_e$, including all of B. So $B \subseteq \mathbf{D}_{t_B+w}$ and therefore $\mathbf{D}_{t_B+w} \supseteq \mathbf{S}_{t_B}$. □

Next, we show that if the boundary batches of $\mathbf{S}$ are sufficiently well-separated in time, CollectComponent returns the same subgraph as CollectComponentIdeal, with the right partitioning and almost the same final time.

Lemma 5.6. *Let* $(\mathbf{D}, \mathcal{D}, \mathbf{T})$ *and* $(\mathbf{S}, \mathcal{S}, \mathbf{U})$ *be returned by* CollectComponent, CollectComponentIdeal, *respectively, and let* $\mathcal{F}$ *be as defined in Definition 5.4. If no pair* $B, B' \in \mathcal{F}$ *has* $|\mathbf{t}_B - \mathbf{t}_{B'}| \leq 2w$, *then* $\mathbf{D} = \mathbf{S}, \mathcal{D} = \mathcal{S}$, *and* $|\mathbf{T} - \mathbf{U}| \leq w$.

Proof. First we will prove that, under these conditions, $\mathbf{D} = \mathbf{S}$. By Lemma 5.5, it will suffice to prove that $\mathbf{D} \subseteq \mathbf{S}$. As in the previous proof, define $\mathbf{D}_t$, $\mathbf{S}_t$ to be the states of $\mathbf{D}$ and $\mathbf{S}$, respectively, after every edge (for $\mathbf{D}$) or batch (for $\mathbf{S}$) with a timestamp no greater than t has been processed. Let $\mathbf{F} = \bigcup \mathcal{F}$. It will suffice to prove that $\mathbf{D}_{1+w} \cap \mathbf{F} \subseteq \mathbf{S}_{1+w} \cap \mathbf{F}$, as the component of $\mathbf{S}_{1+w}$ containing v is contained in $\mathbf{F}$, and if the component of $\mathbf{D}_{1+w}$ containing v included any edge not in $\mathbf{S}_{1+w} \cap \mathbf{F}$, it would also include at least one edge in $\mathbf{F} \setminus \mathbf{S}_{1+w}$, as $\mathbf{F}$ contains the entire boundary of the component of $\mathbf{S}_{1+w}$ containing v (recalling that $\mathbf{D}$, $\mathbf{S}$ are the components of $\mathbf{D}_{1+w}$, $\mathbf{S}_1$ containing v, respectively, and $\mathbf{S}_1 = \mathbf{S}_{1+w}$ trivially).

Fix any assignment of timestamps $(\mathbf{t}_B)_{B\in\mathcal{B}}$, $(\mathbf{t}_e)_{e\in E}$ such that the lemma criterion holds. We will prove the following by (strong) induction on the order of the time stamps $(t_e)_{e\in E}$: for all $e \in G$, $\mathbf{D}_{\mathbf{t}_e} \cap \mathbf{F} \subseteq \mathbf{S}_{t_e} \cap \mathbf{F}$. As no more edges arrive in $(\max_{e\in E} \mathbf{t}_e, w]$, this will give us $\mathbf{D}_{1+w} \subseteq \mathbf{S}_{1+w}$.

For any $e \in G$, suppose that this holds for all f with $\mathbf{t}_f \leq \mathbf{t}_e$. Then we have $\mathbf{D}_t \cap \mathbf{F} \subseteq \mathbf{S}_t \cap \mathbf{F}$ for all $t < \mathbf{t}_e$, as edges are only added to $\mathbf{D}$ at times corresponding to the timestamp of some edge.

Now, if no edges in $\mathbf{F}$ were added to $\mathbf{D}$ at the time $\mathbf{t}_e$, the result holds immediately. So suppose $f \in \mathbf{F}$ was added. Then, one of the following holds:

1. e is connected to v through some path in $\mathbf{D}_s$ for some $s < \mathbf{t}_e$, and $\mathbf{t}_f \in [\mathbf{t}_e - w, \mathbf{t}_e]$.

2. $f = e$, and there is some g such that $\mathbf{t}_g \in [\mathbf{t}_e - w, \mathbf{t}_e]$, and g is connected to v through some path in $\mathbf{D}_s$ for some $s < \mathbf{t}_g$

In the first case, the path connecting e to v in $\mathbf{D}_s$ must be contained in $\mathbf{F}$. To see this, note that every edge incident to v is in $\mathbf{F}$ along with every edge incident to the component containing v in $\mathbf{S}_1$. So if the paths was not containing in $\mathbf{F}$, consider the first edge of the path not in $\mathbf{F}$. This cannot be the first edge of the path, as that edge is incident to v. So consider the edge immediately preceding it. This edge is in $\mathbf{F}$ but not in $\mathbf{S}_1$, as if it were in $\mathbf{S}_1$ every edge incident to it would be in $\mathbf{F}$. So it is not in $\mathbf{S}_s$ and thus by our inductive hypothesis it is not in $\mathbf{D}_s$, contradiction.

As the path is contained in $\mathbf{F}$, it is contained in $\mathbf{S}_s$ by our inductive hypothesis. In that case, e is connected to v through some path in $\mathbf{S}_s$, and so $e \in \mathbf{F}$. Therefore, as the batches in $\mathcal{F}$ have timestamps separated by $2w$, and the batch B containing e has $\mathbf{t}_B \in [\mathbf{t}_e - w, \mathbf{t}_e]$, there is no other batch intersecting $\mathbf{S}_s$ with a time stamp after $\mathbf{t}_e - 2w$, and so e is also connected to v through a path in $\mathbf{S}_{\mathbf{t}_e - 2w} \subseteq \mathbf{t}_{B-w}$. Therefore, B was added to $\mathbf{S}$ at the time $\mathbf{t}_B \in [\mathbf{t}_e - w, \mathbf{t}_e]$. As $f \in \mathbf{F}$, it is in B, as otherwise the batch B' containing it would have to have $\mathbf{t}_{B'} < \mathbf{t}_e - 2w$, which is inconsistent with $\mathbf{t}_f$. So $f \in \mathbf{D}_{\mathbf{t}_B} \subseteq \mathbf{D}_{\mathbf{t}_e}$, completing the proof for this case.

Now consider the second case. By the same argument as for f in the first case, g and a path connecting it to v are in $\mathbf{F}$, and so the path is in $\mathbf{S}_s$. Furthermore, no batch intersecting $\mathbf{S}_s$ arrives in $[\mathbf{t}_g - 2w, \mathbf{t}_g]$, as the batch B containing g is in $\mathcal{F}$ and $\mathbf{t}_B \in [\mathbf{t}_g - w, \mathbf{t}_g]$. Therefore, there is a path connecting g to v in $\mathbf{S}_{\mathbf{t}_B - w}$ and so $B \subseteq \mathbf{S}_{\mathbf{t}_B}$. As $f \in \mathbf{F}$, it is in B, as otherwise the batch B' containing it would have to have $\mathbf{t}_{B'} < \mathbf{t}_e - 2w$, which is inconsistent with $\mathbf{t}_f$. So $f \in \mathbf{D}_{\mathbf{t}_B} \subseteq \mathbf{D}_{\mathbf{t}_e}$, completing the proof.

Now we will prove that the partitioning $\mathcal{D} = \mathcal{S}$. This follows from the fact that $\mathbf{D} = \mathbf{S}$ and the separation of the batch timestamps—if $e, f \in B \in \mathcal{S}$, then $|\mathbf{t}_e - \mathbf{t}_f| \leq w$ and so they are in the same partition in $\mathcal{D}$. Conversely, if e, f are in the same partition in $\mathcal{D}$, $|\mathbf{t}_e - \mathbf{t}_f|$, then they are in the same batch B, as if they were in different batches, both batches would be in $\mathcal{F}$ and would have timestamps within $2w$ of each other. So e, f are in the same partition in $\mathcal{S}$.

Finally, we prove that $|\mathbf{T} - \mathbf{U}| \leq w$. Note that these are the final time a vertex is added to the component containing v in $\mathbf{D}$ or $\mathbf{S}$, respectively. This will therefore follow directly from our proof that $\mathbf{D}_s \subseteq \mathbf{S}_s$, and the Lemma 5.5 proof that $\mathbf{D}_{s+w} \supseteq \mathbf{S}_s$. □

We now show that the criterion of Lemma 5.6 holds, and therefore CollectComponent "almost" matches CollectComponentIdeal, with high probability whenever $\mathcal{F}$ is not too large.

Lemma 5.7. *Let* $(\mathbf{D}, \mathcal{D}, \mathbf{T})$ *and* $(\mathbf{S}, \mathcal{S}, \mathbf{U})$ *be returned by* CollectComponent, CollectComponentIdeal, respectively, and let $\mathcal{F}$ be as defined in Definition 5.4. Then n

$$\Pr[(\mathbf{D} = \mathbf{S}) \wedge (\mathcal{D} = \mathcal{S}) \wedge (|\mathbf{T} - \mathbf{U}| < w) | \mathbf{S}] \geq 1 - 2w\,|\mathcal{F}|^2\,.$$

Proof. Condition on the order in which the batches in $\mathcal{F}$ arrive. First note that *any* value of $(\mathbf{t}_B)_{B \in \mathcal{B}}$ such that these particular batches arrive in the given order is sufficient to fix the value of $\mathcal{F}$ and $\mathbf{S}$.

This means that, conditioned on $\mathbf{S}$, $\mathcal{F}$, and this order, the unlabelled set of timestamps $\{\mathbf{t}_B : B \in \mathcal{F}\}$ is distributed as $|\mathcal{F}|$ independent and uniform samples from $[0, 1]$. Therefore, the probability that any pair of them are within $2w$ is at most $\binom{|\mathcal{F}|}{2} 4w \leq 2w\,|\mathcal{F}|^2$, and so the result follows by Lemma 5.6. □

Our component counting and collection algorithms, Algorithms 3 and 4, will use CollectComponent by setting some small threshold λ and throwing away the result whenever $\mathbf{T} > \lambda$. Now, when $\mathcal{F}$ is large, $\mathbf{T}$ and $\mathbf{U}$ will concentrate near 1. We use this, along with Lemma 5.7, to show that, if $\lambda < 1/2$ and w is small enough in terms of k, either the batches and partitions will match between CollectComponent and CollectComponentIdeal, or they will both be thrown away.

Lemma 5.8. *Let* $(\mathbf{D}, \mathcal{D}, \mathbf{T})$ *and* $(\mathbf{S}, \mathcal{S}, \mathbf{U})$ *be returned by* CollectComponent, CollectComponentIdeal, *respectively. For any* $\lambda \leq 1/2$, *with probability* $1 - \mathrm{O}\big(k^4 w \log^2 1/w\big)$, *either both* $\mathbf{T}$ *and* $\mathbf{U}$ *are greater than* λ, *or* $\mathbf{T}$ *and* $\mathbf{U}$ *are both smaller than* λ *and* $(\mathbf{D}, \mathcal{D}) = (\mathbf{S}, \mathcal{S})$.

Proof. Condition on $\mathbf{S}$ and therefore $\mathcal{F}$. Furthermore, condition on the order in which the batches of $\mathcal{F}$ arrive. First, suppose $|\mathcal{F}| \leq 5k^2 + 10 \log 1/w$. Then by Lemma 5.7, with probability $1 - \mathrm{O}\big(wk^4 \log^2 1/w\big)$,

$$\mathbf{D} = \mathbf{S}, \mathcal{D} = \mathcal{S}, |\mathbf{T} - \mathbf{U}| < w$$

and so the result will hold provided $|\mathbf{U} - \lambda| > w$. $\mathbf{U}$ is always $\mathbf{b}_B$ for some $B \in \mathcal{F}$, so again using the fact that the unlabelled set $\{\mathbf{b}_B : B \in \mathcal{F}\}$ is distributed as $|\mathcal{F}|$ independent and uniform samples, this happens with probability at least $1 - \mathrm{O}(w\,|\mathcal{F}|)$. So for any realization S of $\mathbf{S}$ such that $\mathbf{S} = S$ implies

$$|\mathcal{F}| \leq 5k^2 + 10 \log 1/w$$

we have

$$\Pr[\mathbf{D} = \mathbf{S}, \mathcal{D} = \mathcal{S}, (\mathbf{T}, \mathbf{U} < \lambda \vee \mathbf{T}, \mathbf{U} > \lambda) | \mathbf{S} = S] \geq 1 - \widetilde{\mathrm{O}}\big(wk^4\big).$$

Now suppose $|\mathcal{F}| > 5k^2 + 10 \log 1/w$. In particular, as $\mathbf{D}$ and $\mathbf{S}$ have edges incident to at most k vertices, and $\mathbf{D} \supseteq \mathbf{S}$, at least $9k^2/2 + 10 \log 1/w$ of the batches in $\mathcal{F}$ have an edge with exactly one endpoint in $\mathbf{S}$ and $\mathbf{D}$. Call this set $\mathcal{F}'$.

Each batch in $\mathcal{F}'$ must arrive before $\mathbf{T}$ and $\mathbf{U}$, as otherwise it would've been included in $\mathbf{S}$ or $\mathbf{D}$ (since after these times they each have reached their final vertex set), and so $\mathbf{T}, \mathbf{U} \geq \max_{B \in \mathcal{F}'} \mathbf{t}_B$.

Using again the fact that the unlabelled set $\{\mathbf{b}_B : B \in \mathcal{F}\}$ is distributed as $|\mathcal{F}|$ independent and uniform samples, this means that $\mathbf{T}, \mathbf{U} \leq 1/2$ only if at least $9/10$ of $\mathcal{F}$ has timestamps $\leq 1/2$, which happens with probability at most

$$\begin{aligned}
\binom{|\mathcal{F}|}{9\,|\mathcal{F}|\,/10} 2^{-9|\mathcal{F}|/10} &= \binom{|\mathcal{F}|}{|\mathcal{F}|\,/10} 2^{-9|\mathcal{F}|/10} \\
&\leq \left(\frac{e\,|\mathcal{F}|}{|\mathcal{F}|\,/10}\right)^{|\mathcal{F}|/10} 2^{-9|\mathcal{F}|/10} \\
&= \left(\frac{10e}{2^9}\right)^{|\mathcal{F}|} \\
&\leq 2^{-\log 1/w} \\
&= w
\end{aligned}$$

and so for any realization S of $\mathbf{S}$ such that $\mathbf{S} = S$ implies

$$|\mathcal{F}| > 5k^2 + 10 \log 1/w$$

we have

$$\Pr[\mathbf{T}, \mathbf{U} > 1/2 \geq \lambda | \mathbf{S} = S] \geq 1 - w.$$

As $\mathbf{S}$ uniquely determines $\mathcal{F}$, the lemma therefore holds when conditioning on any realization of $\mathbf{S}$. □

5.3.3 Space complexity of CollectComponent

Lemma 5.9. *Algorithm 1 can be implemented in* $\mathrm{O}\big(k^2(b+mw)\log^2 n\big)$ *bits of space in expectation.*

Proof. At all times in the execution of CollectComponent, W contains, at most, edges with timestamps up to w before that of the last edge processed. Other than W, the algorithm has to keep, up to $\binom{k}{2}$ times, all edges with timestamps within w of some specified edge.

Therefore, the space usage of the algorithm is at most

$$\mathrm{O}\big(k^2 \cdot M^* \cdot \log n\big)$$

bits, where M^* is the largest number of edges with timestamps within any width-$2w$ window in the stream. To bound the expectation of M^*, we start by noting that, as each batch B has at most b edges, all with timestamps in $[\mathbf{t}_B, \mathbf{t}_B + w]$,

$$\begin{aligned} M^* &\leq \max_{B\in\mathcal{B}} \left(|B| + \sum_{\substack{B'\in\mathcal{B}:\\ |\mathbf{t}_{B'}-\mathbf{t}_B|\leq 3w}} \big|B'\big| \right) \\ &\leq b + \max_{B\in\mathcal{B}} \sum_{\substack{B'\in\mathcal{B}:\\ |\mathbf{t}_{B'}-\mathbf{t}_B|\leq 3w}} \big|B'\big| \end{aligned}$$

Now, fix some $B \in \mathcal{B}$. For all $B' \in \mathcal{B} \setminus \{B\}$, let $\mathbf{A}_{B'}$ be the random variable that is $|B'|$ if $|\mathbf{t}_{B'} - \mathbf{t}_B| \leq 3w$ and 0 otherwise. Then the variables $\mathbf{A}_{B'}$ are independent, are size at most b, and the sum of their expectations is at most wm while

$$\begin{aligned} \sum_{B'\in\mathcal{B}\setminus\{B\}} \mathbb{E}\big[\mathbf{A}_{B'}^2\big] &= \sum_{B'\in\mathcal{B}\setminus\{B\}} 3w\,\big|B'\big|^2 \\ &\leq 3wmb. \end{aligned}$$

So by the Bernstein inequalities, for all t, $\sum_{B'\in\mathcal{B}\setminus\{B\}} \mathbf{A}_{B'} \leq m + t$ with probability at least

$$e^{-\Omega\left(\frac{t^2}{wmb+bt}\right)}$$

and so in particular, it is at most $\mathrm{O}((wm+b)\log n)$ with probability at least $1 - m^{-2}$, and so by a union bound $|M^*| = \mathrm{O}((wm+b)\log n)$ with probability at least $1 - m^{-1}$.

We therefore have

$$\begin{aligned} \mathbb{E}[M^*] &\leq \mathrm{O}((wm+b)\log n) + m \cdot m^{-1} \\ &= \mathrm{O}((wm+b)\log n) \end{aligned}$$

and so the lemma follows. □

5.4 Properties of Idealized Component Collection

In this section we will establish some properties of CollectComponentIdeal that will be useful for both *counting* and *collecting* components.

Let $\lambda > 0$ be some parameter to be defined later.

Definition 5.10. *For any $v \in V$, $H \subseteq G$, $\lambda > 0$, the batch probability $p_v^\lambda(H)$ is the probability that both of the following happen:*

- *The batches intersecting H arrive in* batch order*—any order such that, for every batch B intersecting H, there is a path from v to an edge in B consisting entirely of edges in batches intersecting H with timestamps before $\mathbf{t}_B$.*
- *H is covered by time λ—for every $w \in V(H)$, there is a path from v to w consisting entirely of edges in batches intersecting H with timestamps before λ.*

We will use this to define a family of random variables $\mathbf{X}_v$. Let

$$(\mathbf{S}_v, \mathcal{S}_v, \mathbf{U}_v) = \textsc{CollectComponentIdeal}(v, k).$$

Then we define

$$\mathbf{X}_v = \begin{cases} 0 & \text{if } \mathbf{S}_v = \perp \\ 0 & \text{if } \mathbf{U} > \lambda \\ 1/p_v^\lambda(\mathbf{S}_v) & \text{otherwise.} \end{cases}$$

Note that this can be determined entirely from the output of CollectComponentIdeal, without knowing anything else about the stream.

We then define

$$\mathbf{X}_v = \mathbf{Y}_v + \mathbf{Z}_v$$

where $\mathbf{Y}_v$ is $\mathbf{X}_v$ when $\mathbf{S}_v$ is the component of G containing v, and zero otherwise. Note that while $\mathbf{Y}_v$, $\mathbf{Z}_v$ are determined by the stream, they cannot be identified from the output of CollectComponentIdeal alone.

We want to prove that the variables $\mathbf{Y}_v$, corresponding to $\mathbf{S}_v$ being "correct", have "nice" properties—good expectation, bounded variance, and approximate independence. Meanwhile, we want to prove that the "error" variables $\mathbf{Z}_v$ are small in expectation.

5.4.1 Correct Component Contribution

Lemma 5.11.

$$\mathbb{E}[\mathbf{Y}_v] = \begin{cases} 1 & \textit{if } v \textit{ is in a component with} \leq k \textit{ vertices.} \\ 0 & \textit{otherwise.} \end{cases}$$

Proof. If v is *not* in a component with $\leq k$ vertices it is 0 by definition. Otherwise, let H be the component containing it and $\mathcal{H} = \{B : B \in \mathcal{B}, B \cap H \neq \emptyset\}$. Then it will be $1/p_v^\lambda(H)$ if $\mathbf{S}_v = H$, $\mathbf{U} \leq \lambda$, and 0 otherwise. $\mathbf{S}_v = H$ iff the batches in $\mathcal{H}$ arrive in batch order (as defined in Definition 5.10), while $\mathbf{U} \leq \lambda$ iff H is covered by time λ. The probability that both of these happen is exactly $p_v^\lambda(H)$. □

To bound the variance of $\mathbf{Y}_v$, we will need some lower bounds on $p_v^\lambda(H)$ when H is the component containing v.

Lemma 5.12. *For any $v \in V$, let H be the component of G containing v. Then*

$$p_v^\lambda(H) \geq (\lambda/k^2)^k.$$

Proof. Consider a depth-first search tree for H. Suppose that the batches intersecting the edges in this tree arrive in the order corresponding to the depth-first search, before every other batch intersecting H, and before time λ. Then:

- The batches intersecting H arrive in batch order.
- Every vertex in H is covered by a tree of edges in these batches, each of which has timestamp before λ.

Recall that batch order is defined in Definition 5.10.

So $p_v^\lambda(H)$ is at least the probability that this occurs. Now, as there are no more than $\binom{k}{2} \leq k^2$ distinct batches intersecting H, and at most $k-1 < k$ batches intersecting the tree, the probability that the batches intersecting the tree arrive in the depth-first search order and before every other batch intersecting H is at least $(1/k^2)^k$.

Now note that unconditionally, the probability that a given set of fewer than k batches would all arrive before time λ is at least λ^k, and conditioning on them being the first k of the batches intersecting H to arrive only increases this probability. So the probability that both events hold is at least $(\lambda/k^2)^k$, completing the proof. □

Lemma 5.13.

$$\mathrm{Var}(\mathbf{Y}_v) \leq (k/\lambda)^{\mathrm{O}(k)}$$

Proof. If v is in a component with more than k vertices, $\mathbf{Y}_v$ is always 0. Otherwise, by Lemma 5.12, we have

$$\begin{aligned}\mathrm{Var}(\mathbf{Y}_v) &\leq \mathbb{E}\left[\mathbf{Y}_v^2\right] \\ &\leq 1/(\lambda/k^2)^{2k}\end{aligned}$$

completing the proof. □

Lemma 5.14. *For any vertex $v \in V$, if v is in a component with more than k vertices, let $\mathcal{K}_v = \emptyset$, otherwise let it be the set of batches that contain at least one edge in the component of G containing v. For any $U \subseteq V$, if the sets $(\mathcal{K}_v)_{v\in U}$ are disjoint, the variables $(\mathbf{Y}_v)_{v\in U}$ are independent.*

Proof. Suppose v is in a component with more than k vertices. Then $\mathbf{Y}_v = 0$ always and therefore it is independent of $\mathbf{Y}_u$ for all $u \in V$.

Otherwise, for a batch to affect the output of CollectComponentIdeal(v, k), there must be a path from it to v in G. Therefore, $\mathbf{Y}_v$ depends only on the timestamps of batches intersecting its component, that is $\mathcal{K}_v$. So the variables $(\mathbf{Y}_v)_{v\in U}$ are independent provided the sets $(\mathcal{K}_v)_{v\in U}$ are disjoint. □

Lemma 5.15. *Let S be a set of r vertices sampled uniformly (with replacement) from V. Then with probability at least $1 - r^2bk^3/n$ over the choice of S, the variables $(\mathbf{Y}_v)_{v\in U}$ are independent conditioned on S.*

Proof. For each $v \in U$, let K_v be the component containing v. By Lemma 5.14, it will suffice to show that the sets $\{\mathcal{K}_v : v \in U, V(H_v) \leq k\}$ are disjoint with this probability.

For each vertex v in a component with at most k vertices, there are at most k^2 different batches intersecting this component, and therefore at most k^2b edges in these batches, and therefore at most k^2b components such that if w is in that component, $\mathcal{K}_v \cap \mathcal{K}_w \neq \emptyset$.

So if we fix a v in a component with at most k vertices and then select a w from V, there are at most k^3b choices of w such that w is in a component with at most k vertices such that $\mathcal{K}_v \cap \mathcal{K}_w \neq \emptyset$. So for a randomly selected pair v, w this happens with probability at most $\frac{k^3b}{n}$.

The proof then follows by taking a union bound over the $\leq r^2$ pairs of vertices in U. □

5.4.2 Bounding the Contribution of Bad Components

We want to prove that the expectation of the variables $\mathbf{Z}_v$ is very small. For this we need the fact that, for any vertex v, if there are too many different "wrong" components that we might find connecting v, as can be the case when v is in a large component, each of these components also has a large boundary, and so is unlikely to be found.

Lemma 5.16. *For any $\mathcal{H} \subset \mathcal{B}$, $v \in V$, let the* v-boundary *of $\mathcal{H}$ be the set of batches in $\mathcal{B} \setminus \mathcal{H}$ that contain at least one edge that is connected to v by a path of edges in $\bigcup \mathcal{H}$.*

For any $v \in V$, $h, a \in \mathbb{N}$, the number of size-h subsets $\mathcal{H}$ of $\mathcal{B}$ such that

1. *every batch in $\mathcal{H}$ contains an edge e such that there is a path from v to an endpoint of e in $\bigcup \mathcal{H}$*
2. *the v-boundary of $\mathcal{H}$ contains exactly a batches*

is at most

$$\binom{h+a}{a}.$$

Proof. In this proof we will make use of edge *contraction*—to contract by an edge uv, uv is removed from the graph and the vertices u, v are identified with each other, and so any edge incident to either is now incident to the merged vertex. This may result in the graph becoming a multigraph (if u, v are incident to the same edge) and having self-loops (if the graph is already a multigraph, and one of multiple edges between u and v is contracted). We will therefore prove the lemma for multigraphs with self-loops, and it will follow for simple graphs as a special case.

Note that when contracting multiple edges, it does not matter in which order we contract them.

We proceed by induction on $h + a$. If $h + a = 0$, the result follows automatically. So suppose $h + a > 0$ and the result holds for all smaller values of $h + a$. Then there is at least one batch B containing an edge incident to v. We will use the inductive hypothesis to bound the number of choices of $\mathcal{H}$ that contain B, and the number that do not.

First, consider any choice of $\mathcal{H}$ that does not contain B. Consider the graph $G' = (V, E \setminus B)$ with batching $\mathcal{B}' = \mathcal{B} \setminus \{B\}$. Then, each such choice of $\mathcal{H}$ is a subset of $\mathcal{B}'$ whose v-boundary with

respect to $G', \mathcal{B}'$ contains $a-1$ batches. Moreover, each batch in $\mathcal{H}$ still contains an edge e with a path from v to an endpoint of e in $\bigcup \mathcal{H}$. So by applying our inductive hypothesis to $G', \mathcal{B}'$, there are at most $\binom{h+a-1}{a-1}$ such choices of $\mathcal{H}$.

Secondly, consider any choice of $\mathcal{H}$ that *does* contain B. Consider the graph G^* obtained by contracting every edge in B, with batching $\mathcal{B}^*$ given by removing B and contracting each of its edges for the other batches. Then there is a one-to-one correspondence between such choices of $\mathcal{H}$ and subsets $\mathcal{H}^*$ of $\mathcal{B}^*$, again given by removing B and contracting each of its edges for the other batches. Each such subset will have a v-boundary containing exactly a batches, but will only contain $h-1$ batches. Furthermore, as contracting edges preserves connectedness, every batch in $\mathcal{H}^*$ will contain an edge e such that there is a path from v to an endpoint of e in $\bigcup \mathcal{H}^*$. So, by applying our inductive hypothesis to $G^*, \mathcal{B}^*$, there are at most $\binom{h+a-1}{a}$ such choices of $\mathcal{H}^*$, and therefore of $\mathcal{H}$.

The lemma then follows from the fact that

$$\binom{h+a-1}{a-1} + \binom{h+a-1}{a} = \binom{h+a}{a}.$$

□

Now we are ready to bound the expectation of the "bad" contributions $\mathbf{Z}_v$, corresponding to $\mathbf{X}_v$ when the subgraph $\mathbf{S}$ returned by CollectComponentIdeal is *not* equal to the actual component K_v.

Lemma 5.17. *As long as* $\lambda \leq 1/2ek^2$,

$$\mathbb{E}[\mathbf{Z}_v] = O(\lambda k^4).$$

Proof. $\mathbf{Z}_v$ is non-zero precisely when CollectComponentIdeal(v, k) returns $(\mathbf{S}_v, \mathcal{S}, \mathbf{U})$ such that $\mathbf{U} \leq \lambda$ and $\mathbf{S}_v \neq C_v$, where C_v is the component containing v in G. For any possible value of $\mathbf{S}_v$, this requires that

1. the set of batches $\mathcal{H}$ that intersect $\mathbf{S}_v$ arrive in batch order (as defined in Definition 5.10)
2. $\mathbf{S}_v$ is covered by time λ
3. every batch B in the v-boundary of $\mathcal{H}$ has $\mathbf{t}_B < \lambda$

with the latter being necessary because as $\mathbf{S}_v$ is covered by time λ, any batch with timestamp at least λ and connected to v by edges in $\bigcup \mathcal{H}$ has an edge incident to $\mathbf{S}_v$, and will therefore be in $\mathcal{H}$ and therefore not in its v-boundary.

So the probability that it happens is at most $p_v^\lambda(\mathbf{S}_v)\lambda^a$ if the v-boundary of $\mathcal{H}$ contains a batches. So for each possible $\mathbf{S}_v$ with a batch set with boundary a, the expected contribution to $\mathbf{Z}_v$ from it is at most λ^a.

As $\mathcal{H}$ determines $\mathbf{S}_v$ exactly (since $\mathbf{S}_v$ is the component containing v in $\bigcup \mathcal{H}$, we can therefore use Lemma 5.16 to bound the expected contribution to $\mathbf{Z}_v$ from collecting size-h sets of batches $\mathcal{H}$ with size-a boundaries by $\binom{h+a}{a}\lambda^a$.

As $\mathbf{S}_v \neq C_v$ requires $\mathcal{H}$ to have at least one batch in its v-boundary, we can therefore bound $\mathbb{E}[\mathbf{Z}_v]$ by

$$\begin{aligned}\sum_{h=0}^{\binom{k}{2}}\sum_{a=1}^{\infty}\binom{h+a}{a}\lambda^a &\leq \sum_{h=0}^{\binom{k}{2}}\sum_{a=1}^{\infty}(\lambda e(h/a+1))^a \\ &\leq k^2\sum_{a=1}^{\infty}(\lambda e(k^2+1))^a \\ &= \mathrm{O}\big(\lambda k^4\big)\end{aligned}$$

provided $\lambda \leq 1/2ek^2$. □

5.5 Component Counting

For any $v \in V$, we will use c_v to denote the size of the component containing v. In particular this means that $c(G)$, the number of components in G is $\sum_{v\in V}\frac{1}{c_v}$.

Let $\lambda \in (0, 1/2)$ and $r \in \mathbb{N}$. We now present an algorithm for component counting based on r copies of CollectComponent. Informally, the algorithm works as follows:

- Sample r vertices $(\mathbf{v}_i)_{i=1}^r$.
- For each vertex v, run CollectComponent(v, k), rejecting the answer if the timestamp $\mathbf{T}$ returned is greater than λ.
- Approximate the component count $c(G)$ by $\frac{n}{r}\sum_{i=1}^r \frac{\mathbf{X}_{\mathbf{v}_i}}{\mathbf{c}_{\mathbf{v}_i}}$, where $\mathbf{c}_{\mathbf{c}_i}$ is the number of vertices in the component that CollectComponent sampled at $\mathbf{v}_i$. We don't have direct access to the variables $\mathbf{X}_v$ but we can do this (with good enough probability) because CollectComponent normally almost-matches CollectComponentIdeal and therefore $\mathbf{X}_v$ is usually $1/p_v^\lambda(\mathbf{D})$.

This will usually give us a good approximation to $\frac{n}{r}\sum_{i=1}^r \frac{1}{c_{\mathbf{v}_i}}$, because $\mathbf{X}_v = \mathbf{Y}_v + \mathbf{Z}_v$ is dominated by $\mathbf{Y}_v$, corresponding to the case when the returned component $\mathbf{D}$ is actually K_v (and so $\mathbf{c}_v = c_v$), provided λ is small enough (so that $\mathbf{Z}_v$ is small in expectation) and r is large enough (so that $\sum_{i=1}^r \frac{1}{c_v}\mathbf{Y}_{\mathbf{v}_i}$ concentrates around its expectation over $(\mathbf{X}_{\mathbf{v}_i})_{i=1}^r$).

Then, if r is big enough, $\frac{n}{r}\sum_{i=1}^r \frac{1}{c_{\mathbf{v}_i}}$ will usually approximate $\sum_{v\in V}\frac{1}{c_v} = c(G)$ well enough, so we are done.

We now formally describe the algorithm.

Algorithm 3 Counting the number of components in a graph.

procedure CountComponents(k, λ, r)
  $\mathbf{C} \leftarrow 0$
  for $i \in [r]$ **do**
    $\mathbf{v}_i \leftarrow \mathcal{U}(V)$
    $(\mathbf{D}_i, \mathcal{D}_i, \mathbf{T}_i) \leftarrow$ CollectComponent($\mathbf{v}_i, k$)
    if $\mathbf{T}_i \leq \lambda$ **then**
      $\mathbf{c}_{\mathbf{v}_i} \leftarrow |V(\mathbf{D}_i)|$
      $\mathbf{C} \leftarrow \mathbf{C} + \frac{n}{r\mathbf{c}_{\mathbf{v}_i}} \cdot \frac{1}{p^{\lambda}_{\mathbf{v}_i}(\mathbf{D}_i)}$ ▷ Usually $\frac{n\mathbf{X}_{\mathbf{v}_i}}{r\mathbf{c}_{\mathbf{v}_i}}$.
    end if
  end for
end procedure

Recall that $\frac{1}{p^{\lambda}_{\mathbf{v}_i}(\mathbf{D}_i)} = \mathbf{X}_{\mathbf{v}_i}$ whenever $(\mathbf{D}_i, \mathcal{D}_i)$ match the subgraph and batching given by $(\mathbf{S}, \mathcal{S}, \mathbf{U}) =$ CollectComponentIdeal($\mathbf{v}_i, k$) and the time stamps $\mathbf{T}_i$, $\mathbf{U}$ are either both smaller than or both larger than λ. We start by showing that when this is the case, the algorithm approximates $c(G)$ with good probability.

We start by showing that, if our inner loop simply added $\frac{n}{r} \cdot \frac{1}{c_v}$ for each v (or zero if $c_v > k$), we would get a good approximation to $c(G)$ with good probability.

Lemma 5.18. *With probability at least $1 - k^2/r$ over $(\mathbf{v}_i)_{i=1}^r$,*

$$\left| c(G) - \frac{n}{r} \sum_{i=1}^{r} \frac{1}{c_{\mathbf{v}_i}} \mathbb{I}(c_{\mathbf{v}_i} \leq k) \right| < \frac{2}{k} n.$$

Proof. As the $\mathbf{v}_i$ are sampled independently, the variables

$$\left(\frac{n}{c_{\mathbf{v}_i}} \mathbb{I}(c_{\mathbf{v}_i} \leq k) \right)_{i=1}^{r}$$

are independent. Each has expectation at least $c(G) - \frac{1}{k} n$, as

$$\mathbb{E}\left[\frac{n}{\mathbf{c}_{\mathbf{v}_i}} \right] = c(G)$$

and any time $c_{\mathbf{v}_i} > k$, $\frac{n}{\mathbf{c}_{\mathbf{v}_i}} \leq \frac{1}{k} n$.

Moreover, each has variance at most n^2, and so by Chebyshev's inequality their average will be within $\frac{1}{k} n$ of their expectation with probability $1 - k^2/r$. □

Next, we use the fact that the "good" part of $\mathbf{X}_{\mathbf{v}_i}$, $\mathbf{Y}_{\mathbf{v}_i}$, is zero whenever $\mathbf{D}_i$ is not the right guess for the component containing $\mathbf{v}_i$ to show that $\frac{1}{\mathbf{c}_v} \mathbf{Y}_{\mathbf{v}_i}$ is a good enough substitute for $\frac{1}{c_v} \mathbb{I}(c_v \leq k)$.

Lemma 5.19. *With probability at least $1 - r^2 b k^3/n - (k/\lambda)^{O(k)}/\sqrt{r}$ over $(\mathbf{v}_i)_{i=1}^r$ and the order of the stream,*

$$\left| \frac{n}{r} \sum_{i=1}^{r} \frac{1}{\mathbf{c}_v} \mathbf{Y}_{\mathbf{v}_i} - \frac{n}{r} \sum_{i=1}^{r} \frac{1}{c_v} \mathbb{I}(c_v \leq k) \right| \leq \frac{1}{k} n.$$

Proof. For any v, $\mathbf{Y}_v$ is zero whenever $c_v > k$ and otherwise by Lemmas 5.11, 5.13, it has expectation 1 and variance $(k/\lambda)^{\mathrm{O}(k)}$. Furthermore, whenever $\mathbf{Y}_v$ is non-zero, $\mathbf{c}_v = c_v$.

So

$$\frac{n}{r}\sum_{i=1}^{r}\mathbb{E}\left[\frac{1}{\mathbf{c}_v}\mathbf{Y}_{\mathbf{v}_i}\middle|(\mathbf{v}_i)_{i=1}^r\right] = \frac{n}{r}\sum_{i=1}^{r}\frac{1}{c_v}\mathbb{I}(c_v \le k)$$

and by Lemma 5.15, with probability at least $1 - r^2bk^3/n$ over $(\mathbf{v}_i)_{i=1}^r$, the variables $\left(\frac{1}{\mathbf{c}_v}\mathbf{Y}_{\mathbf{v}_i}\right)_{i=1}^r$ are independent with variances at most $(k/\lambda)^{\mathrm{O}(k)}$. So by taking a union bound with Chebyshev's inequality, the lemma follows. □

This leaves an error term $\left|\frac{n}{r}\sum_{i=1}^{r}\frac{1}{\mathbf{c}_v}\mathbf{Z}_{\mathbf{v}_i}\right|$ to deal with. As the expectation of $\mathbf{Z}_{\mathbf{v}_i}$ is small when λ is small enough, we can show that this is usually small through applying Markov's inequality.

Lemma 5.20. *Fix any value of $(\mathbf{v}_i)_{i=1}^r$. As long as $\lambda \le 1/2ek^2$, with probability at least $1 - O(\lambda k^5)$ over the order of the stream,*

$$\left|\frac{n}{r}\sum_{i=1}^{r}\frac{1}{\mathbf{c}_v}\mathbf{Z}_{\mathbf{v}_i}\right| \le \frac{1}{k}n.$$

Proof. By Lemma 5.17, whenever $\lambda \le 1/2ek^2$, $\mathbb{E}[\mathbf{Z}_v] = \mathrm{O}\big(\lambda k^4\big)$ for all $v \in V$, so this follows by a direct application of Markov's inequality. □

This tells us that $\frac{n}{r}\sum_{i=1}^{r}\frac{1}{\mathbf{c}_v}\mathbf{X}_{\mathbf{v}_i}$ is a good approximation to $c(G)$ with good probability, and so we can use Lemma 5.8 to lower bound our success probability.

Lemma 5.21. *As long as $\lambda \le 1/2ek^2$, with probability at least*

$$1 - \mathrm{O}\Big(rk^4w\log^2 1/w + r^2bk^3/n + (k/\lambda)^{O(k)}/\sqrt{r} + \lambda k^5\Big)$$

over $(\mathbf{v}_i)_{i=1}^r$ and the order of the stream,

$$|\mathbf{C} - c(G)| \le \frac{4}{k}n.$$

Proof. By Lemma 5.8 and a union bound, with probability $1 - \mathrm{O}\big(k^4w\log^2 1/w\big)$, taking the output of an instance of CollectComponent and proceeding iff the timestamp output is at most λ will give the same result as doing so with an instance of CollectComponentIdeal. Therefore, by taking a union bound over the r iterations of the inner loop, with probability $1 - \mathrm{O}\big(rk^4w\log^2 1/w\big)$,

$$\mathbf{C} = \frac{n}{r}\sum_{i=1}^{r}\frac{1}{\mathbf{c}_v}\mathbf{X}_{\mathbf{v}_i}.$$

So recalling that $\mathbf{X} = \mathbf{Y} + \mathbf{Z}$, and taking a union bound over Lemmas 5.18, 5.19, and 5.20, the lemma follows. □

We now prove Theorem 5.1, restated here for convenience of the reader. The theorem follows by carefully choosing our algorithm parameters in terms of w, b, and m.

Theorem 5.1 (Counting Components)**.** *For all $\varepsilon, \delta \in (0,1)$, there is a (b,w)-hidden batch random order streaming algorithm that achieves an εn additive approximation to $c(G)$ with $1-\delta$ probability, using*

$$(1/\varepsilon\delta)^{\mathrm{O}(1/\varepsilon)}(b+wm)\operatorname{polylog}(n)$$

bits of space.

Proof. Assume that $\delta, \varepsilon \leq 1/2$ (if they are in $(1/2,1)$, the result will follow from the $1/2$ case). We start by setting $k = 4/\varepsilon$, and $\lambda = \Theta\big(\delta/k^5\big)$ such that the $\mathrm{O}\big(\lambda k^5\big)$ term in Lemma 5.21 is at most $\delta/4$ and $\lambda \leq 1/2ek^2$. Then, we set $r = (1/\varepsilon\delta)^{\Theta(1/\varepsilon)}$ such that the $\mathrm{O}\big((k/\lambda)^{O(k)}/\sqrt{r}\big)$ term is at most $\delta/4$.

Now, consider the $\mathrm{O}\big(r^2bk^3/n\big)$ term. If it is greater than $\delta/4$, we have

$$n\log n \leq (1/\varepsilon\delta)^{\mathrm{O}(1/\varepsilon)} b\log n$$

and so the theorem follows immediately by using a union-find to exactly calculate the components of G.

If the $\mathrm{O}\big(rk^4w\log^2 1/w\big)$ term is greater than $\delta/4$, we start by noting that

$$w\log^2 1/w = \mathrm{O}\big((1/m+w)\log^2 1/n\big)$$

as $w\log^2 1/w = \mathrm{O}\big(m^{-1}\log m\big)$ when $w \leq 1/m$), so we have

$$n\log n \geq (1/\varepsilon\delta)^{\mathrm{O}(1/\varepsilon)}(1+wm)\log^3 n$$

and so the theorem again follows from using a union-find.

If neither of these hold, Lemma 5.21 tells us that running CountComponents(k,λ,r) will give a εn additive approximation to $c(G)$ with probability $1-\delta$. As the space needed is that required to run r copies of CollectComponent, by Lemma 5.9 we achieve the desired space. □

5.6 Component Collection

Let $\lambda \in (0,1/2)$ and $r \in \mathbb{N}$. We now present an algorithm for collecting components based on r copies of CollectComponent. Informally, the algorithm works as follows:

- Sample r vertices $(\mathbf{v}_i)_{i=1}^r$.
- For each vertex v, run CollectComponent(v,k), rejecting the answer if the timestamp $\mathbf{T}$ returned is greater than λ.
- Sample one of the components $\mathbf{D}$ returned by these with probability weighted by approximately $\mathbf{X}_v$ (using the fact that CollectComponent normally almost-matches CollectComponentIdeal and therefore $\mathbf{X}_v$ is usually $1/p_v^\lambda(\mathbf{D})$.

This will usually give us an actual component, because $\mathbf{X}_v = \mathbf{Y}_v + \mathbf{Z}_v$ is dominated by $\mathbf{Y}_v$, corresponding to the case when the returned component $\mathbf{D}$ is actually K_v, provided λ is small enough (so that $\mathbf{Z}_v$ is small in expectation) and r is large enough (so that $\sum_{i=1}^r \mathbf{Y}_{\mathbf{v}_i}$ concentrates around its expectation).

We now formally describe the algorithm.

Algorithm 4 Collecting a component in a graph.

procedure FINDCOMPONENT(k, λ, r)
  $p \leftarrow 0$
  for $i \in [r]$ **do**
    $\mathbf{v}_i \leftarrow \mathcal{U}(V)$
    $(\mathbf{D}_i, \mathcal{D}_i, \mathbf{T}_i) \leftarrow$ COLLECTCOMPONENT($\mathbf{v}_i, k$)
    if $\mathbf{D}_i \neq \perp \wedge \mathbf{T}_i < \lambda$ **then**
      $p_i \leftarrow 1/p^{\lambda}_{\mathbf{v}_i}(\mathbf{D}_i))$ ▷ Usually $\mathbf{X}_{\mathbf{v}_i}$.
      $p \leftarrow p + p_i$
    else
      $p_i \leftarrow 0$
    end if
  end for
  if $p = 0$ **then**
    return $\perp$
  end if
  $(\mathbf{v}^*, \mathbf{D}^*) \leftarrow (\mathbf{v}_i, \mathbf{D}_i)$ with probability p_i/p for each i.
  return $(\mathbf{v}^*, \mathbf{D}^*)$
end procedure

Recall that $\frac{1}{p^{\lambda}_{\mathbf{v}_i}(\mathbf{D}_i)} = \mathbf{X}_{\mathbf{v}_i}$ whenever $(\mathbf{D}_i, \mathcal{D}_i)$ match the subgraph and batching given by $(\mathbf{S}, \mathcal{S}, \mathbf{U}) =$ COLLECTCOMPONENTIDEAL($\mathbf{v}_i, k$) and the time stamps $\mathbf{T}_i$, $\mathbf{U}$ are either both smaller than or both larger than λ. We start by showing that when this is the case, the algorithm returns an actual component of G with good probability.

When it holds (and assuming at least one run does not return $\perp$), the probability of returning a real component will be proportional to $\sum_{i=1}^{r} \mathbf{Y}_{\mathbf{v}_i}$, as these are the $\mathbf{X}_{\mathbf{v}_i}$ such that $\mathbf{D}_i$ is the component containing $\mathbf{v}_i$. Meanwhile the probability of returning a bad component will be proportional to $\sum_{i=1}^{r} \mathbf{Z}_{\mathbf{v}_i}$. So we need to prove that $\sum_{i=1}^{r} \mathbf{Y}_{\mathbf{v}_i}$ is non-zero and large relative to $\sum_{i=1}^{r} \mathbf{Z}_{\mathbf{v}_i}$.

First, we need a good enough fraction of the vertices sampled to be in size $\leq k$ components, as otherwise the $\mathbf{Y}_{\mathbf{v}_i}$ will be identically zero.

Lemma 5.22. *Suppose a β fraction of vertices of G are in components of size at most k. Then with probability $1 - e^{-r\lambda/2\beta}$ over $(\mathbf{v}_i)_{i=1}^{r}$, at least a $\beta - \sqrt{\lambda}$ fraction of the vertices $(\mathbf{v}_i)_{i=1}^{r}$ are in components of size at most k.*

Proof. The vertices are sampled independently, so this follows directly by the Chernoff bounds. □

Given this, we show that $\sum_{i=1}^{r} \mathbf{Y}_{\mathbf{v}_i}$ is reasonably large.

Lemma 5.23. *Suppose a β fraction of vertices of G are in components of size at most k. Then with probability at least $1 - e^{-r\lambda/2\beta} - r^2bk^3/n - (k/\lambda)^{O(k)}/\sqrt{r}$ over $(\mathbf{v}_i)_{i=1}^{r}$ and the order of the stream,*

$$\sum_{i=1}^{r} \mathbf{Y}_{\mathbf{v}_i} \geq r(\beta - 2\sqrt{\lambda}).$$

Proof. For any v, $\mathbf{Y}_v$ is zero whenever it is in a component of size greater than k and otherwise by Lemmas 5.11, 5.13, it has expectation 1 and variance $(k/\lambda)^{\mathrm{O}(k)}$.

So

$$\sum_{i=1}^{r} \mathbb{E}[\mathbf{Y}_{\mathbf{v}_i} | (\mathbf{v}_i)_{i=1}^r] = \sum_{i=1}^{r} \mathbb{I}(c_v \leq k)$$

and by Lemma 5.15, with probability at least $1 - r^2bk^3/n$ over $(\mathbf{v}_i)_{i=1}^r$, the variables $(\mathbf{Y}_{\mathbf{v}_i})_{i=1}^r$ are independent with variances at most $(k/\lambda)^{\mathrm{O}(k)}$. So by taking a union bound with Chebyshev's inequality and the result of Lemma 5.22, the lemma follows. □

We show that $\sum_{i=1}^r \mathbf{Z}_{\mathbf{v}_i}$ is small (when λ is small enough) by invoking the bound on the expectation of individual $\mathbf{Z}_v$.

Lemma 5.24. *Fix any value of $(\mathbf{v}_i)_{i=1}^r$. As long as $\lambda \leq 1/2ek^2$, with probability at least $1 - O(\sqrt{\lambda}k^4)$ over the order of the stream,*

$$\sum_{i=1}^{r} \mathbf{Z}_{\mathbf{v}_i} \leq r\sqrt{\lambda}.$$

Proof. By Lemma 5.17, whenever $\lambda \leq 1/2ek^2$, $\mathbb{E}[\mathbf{Z}_v] = \mathrm{O}\big(\lambda k^4\big)$ for all $v \in V$, so this follows by a direct application of Markov's inequality. □

So now we have that, with good enough probability, $\sum_{i=1}^r \mathbf{Y}_{\mathbf{v}_i}$ is large relative to $\sum_{i=1}^r \mathbf{Z}_{\mathbf{v}_i}$ and so we use the fact that the output of CollectComponent usually almost matches the output of CollectComponentIdeal to lower bound the probability with which our algorithm outputs a valid component.

Lemma 5.25. *As long as $\lambda \leq 1/2ek^2$, with probability at least*

$$1 - \mathrm{O}\Big(e^{-r\lambda/2\beta} + rk^4 w \log^2 1/w + r^2bk^3/n + (k/\lambda)^{O(k)}/\sqrt{r} + \sqrt{\lambda}k^4 + \sqrt{\lambda}/\beta\Big)$$

over $(\mathbf{v}_i)_{i=1}^r$ and the order of the stream, $\mathbf{D}^*$ is the component of G containing $\mathbf{v}^*$.

Proof. By Lemma 5.8 and a union bound, with probability $1 - \mathrm{O}\big(k^4 w \log^2 1/w\big)$, taking the output of an instance of CollectComponent and proceeding iff the timestamp output is at most λ will give the same result as doing so with an instance of CollectComponentIdeal. Therefore, by taking a union bound over the r iterations of the inner loop, with probability $1 - \mathrm{O}\big(rk^4 w \log^2 1/w\big)$,

$$(p_i)_{i=1}^r = (\mathbf{X}_{\mathbf{v}_i})_{i=1}^r.$$

So recalling that $\mathbf{X} = \mathbf{Y} + \mathbf{Z}$, and taking a union bound over Lemmas 5.23 and 5.24, we have that with probability at least

$$1 - \mathrm{O}\Big(e^{-r\lambda/2\beta} + k^4 w \log^2 1/w + r^2bk^3/n + (k/\lambda)^{O(k)}/\sqrt{r} + \sqrt{\lambda}k^4\Big)$$

over $(\mathbf{v}_i)_{i=1}^r$ and the order of the stream,

$$\begin{aligned}\frac{\sum_{i=1}^r \mathbf{Y}_{\mathbf{v}_i}}{\sum_{i=1}^r \mathbf{Z}_{\mathbf{v}_i}} &\geq \frac{r(\beta - 2\sqrt{\lambda})}{r(\beta - \sqrt{\lambda})} \\ &= 1 - \mathrm{O}\Big(\sqrt{\lambda}/\beta\Big)\end{aligned}$$

So if this holds, the algorithm will output a correct component with probability $1 - \mathrm{O}\left(\sqrt{\lambda}/\beta\right)$, as $\mathbf{Y}_i > 0$ iff $\mathbf{D}_i$ is the component of G containing $\mathbf{v}_i$. The lemma therefore follows from taking one final union bound. □

Finally Theorem 5.2 follows by carefully choosing the algorithm parameters in terms of w, b, and m.

Theorem 5.2 (Component Collection)**.** *For all $\delta \in (0,1)$, there is a (b,w)-hidden batch random order streaming algorithm such that, if at least a β fraction of the vertices of G are in components of size at most ℓ, returns a vertex in G and the component containing it with probability $1-\delta$ over its internal randomness and the order of the stream, using*

$$(\ell/\beta\delta)^{\mathrm{O}(\ell)}(b + wm)\operatorname{polylog} n$$

bits of space.

Proof. Assume that $\delta \leq 1/2$ (if it is in $(1/2, 1)$, the result will follow from the $1/2$ case). We start by setting $k = \ell$, and $\lambda = \Theta\big(\delta^2/k^8 + \delta^2/\beta^2\big)$ such that the $\mathrm{O}\left(\sqrt{\lambda}/\beta\right)$ and $\mathrm{O}\left(\sqrt{\lambda k^4}\right)$ terms in Lemma 5.25 sum to at most $\delta/3$ and $\lambda \leq 1/2ek^2$. Then, we set $r = (\ell/\beta\delta)^{\Theta(1/\varepsilon)}$ such that the $\mathrm{O}\big((k/\lambda)^{O(k)}/\sqrt{r}\big)$ and $e^{-r\lambda/2\beta}$ terms are at most $\delta/3$.

Now, consider the $\mathrm{O}\big(r^2bk^3/n\big)$ term. If it is greater than $\delta/4$, we have

$$\ell n \log n \leq (\ell/\beta\delta)^{\mathrm{O}(1/\varepsilon)} b \log n$$

and so the theorem follows immediately by keeping the first ℓ edges incident to each vertex we see.

If the $\mathrm{O}\big(rk^4 w \log^2 1/w\big)$ term is greater than $\delta/4$, we start by noting that

$$w \log^2 1/w = \mathrm{O}\big((1/m + w)\log^2 1/n\big)$$

as $w \log^2 1/w = \mathrm{O}\big(m^{-1}\log m\big)$ when $w \leq 1/m$, so we have

$$\ell n \log n \geq (\ell/\beta\delta)^{\mathrm{O}(1/\varepsilon)}(1 + wm)\log^3 n$$

and so the theorem again follows.

If neither of these hold, Lemma 5.25 tells us that running FindComponent(k, λ, r) will give a vertex in G and the component containing it with probability $1 - \delta$. As the space needed is that required to run r copies of CollectComponent, by Lemma 5.9 we achieve the desired space. □

Acknowledgements

Ashish Chiplunkar was partially supported by the Pankaj Gupta New Faculty Fellowship. John Kallaugher and Eric Price were supported by NSF Award CCF-1751040 (CAREER). Michael Kapralov was supported by ERC Starting Grant 759471.

John was also supported by Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525. Also supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Accelerated Research in Quantum Computing program.

References

[AKL17] Sepehr Assadi, Sanjeev Khanna, and Yang Li. On estimating maximum matching size in graph streams. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1723–1742. SIAM, 2017.

[AKSY20] Sepehr Assadi, Gillat Kol, Raghuvansh R. Saxena, and Huacheng Yu. Multi-pass graph streaming lower bounds for cycle counting, MAX-CUT, matching size, and other problems. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 354–364. IEEE, 2020.

[AMS96] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 20–29. ACM, 1996.

[AN21] Sepehr Assadi and Vishvajeet N. Graph streaming lower bounds for parameter estimation and property testing via a streaming XOR lemma. *STOC*, 2021.

[BGM+19] Marc Bury, Elena Grigorescu, Andrew McGregor, Morteza Monemizadeh, Chris Schwiegelshohn, Sofya Vorotnikova, and Samson Zhou. Structural results on matching estimation with applications to streaming. *Algorithmica*, 81(1):367–392, 2019.

[BS15] Marc Bury and Chris Schwiegelshohn. Sublinear estimation of weighted matchings in dynamic data streams. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 263–274. Springer, 2015.

[CCE+16] Rajesh Chitnis, Graham Cormode, Hossein Esfandiari, MohammadTaghi Hajiaghayi, Andrew McGregor, Morteza Monemizadeh, and Sofya Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1326–1344. SIAM, 2016.

[CCM08] Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Robust lower bounds for communication and stream computation. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 641–650. ACM, 2008.

[CFPS20] Artur Czumaj, Hendrik Fichtenberger, Pan Peng, and Christian Sohler. Testable properties in general graphs and random order streaming. In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 16:1–16:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[CGV20] Chi-Ning Chou, Sasha Golovnev, and Santhoshini Velusamy. Optimal streaming approximations for all boolean max-2csps and max-ksat. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 330–341, 2020.

[CJMM17] Graham Cormode, Hossein Jowhari, Morteza Monemizadeh, and S. Muthukrishnan. The sparse awakens: Streaming algorithms for matching size estimation in sparse graphs. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPIcs*, pages 29:1–29:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[CJP08] Amit Chakrabarti, T. S. Jayram, and Mihai Patrascu. Tight lower bounds for selection in randomly ordered streams. In Shang-Hua Teng, editor, *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 720–729. SIAM, 2008.

[Dav03] H. A. (Herbert Aron) David. *Order statistics H.A. David, H.N. Nagaraja.* John Wiley, Hoboken, N.J, 3rd ed. edition, 2003.

[EHL+15] Hossein Esfandiari, Mohammad Taghi Hajiaghayi, Vahid Liaghat, Morteza Monemizadeh, and Krzysztof Onak. Streaming algorithms for estimating the matching size in planar graphs and beyond. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1217–1233. SIAM, 2015.

[EHM16] Hossein Esfandiari, MohammadTaghi Hajiaghayi, and Morteza Monemizadeh. Finding large matchings in semi-streaming. In Carlotta Domeniconi, Francesco Gullo, Francesco Bonchi, Josep Domingo-Ferrer, Ricardo A. Baeza-Yates, Zhi-Hua Zhou, and Xindong Wu, editors, *IEEE International Conference on Data Mining Workshops, ICDM Workshops 2016, December 12-15, 2016, Barcelona, Spain.*, pages 608–614. IEEE Computer Society, 2016.

[GKK+07] Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 516–525. ACM, 2007.

[GT19] Venkatesan Guruswami and Runzhou Tao. Streaming hardness of unique games. In Dimitris Achlioptas and László A. Végh, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019, September 20-22, 2019, Massachusetts Institute of Technology, Cambridge, MA, USA*, volume 145 of *LIPIcs*, pages 5:1–5:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[GVV17] Venkatesan Guruswami, Ameya Velingker, and Santhoshini Velusamy. Streaming complexity of approximating max 2CSP and max acyclic subgraph. In Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 8:1–8:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[KK19] Michael Kapralov and Dmitry Krachun. An optimal space lower bound for approximating MAX-CUT. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 277–288. ACM, 2019.

[KKL88] J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, SFCS '88, pages 68–80, USA, 1988. IEEE Computer Society.

[KKP18] 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.

[KKP22] John Kallaugher, Michael Kapralov, and Eric Price. Simulating random walks in random streams. In *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022 (to appear)*, 2022.

[KKS14] Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Approximating matching size from random streams. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 734–751. SIAM, 2014.

[KKS15] Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming lower bounds for approximating MAX-CUT. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1263–1282. SIAM, 2015.

[KKSV17] Michael Kapralov, Sanjeev Khanna, Madhu Sudan, and Ameya Velingker. $(1 + \Omega(1))$-approximation to MAX-CUT requires linear space. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1703–1722. SIAM, 2017.

[KMNT20] Michael Kapralov, Slobodan Mitrovic, Ashkan Norouzi-Fard, and Jakab Tardos. Space efficient approximation to maximum matching size from uniform edge samples. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1753–1772. SIAM, 2020.

[MMPS17] Morteza Monemizadeh, S. Muthukrishnan, Pan Peng, and Christian Sohler. Testable bounded degree graph properties are random order streamable. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 131:1–131:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[MV16] Andrew McGregor and Sofya Vorotnikova. Planar matching in streams revisited. In Klaus Jansen, Claire Mathieu, José D. P. Rolim, and Chris Umans, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016, September 7-9, 2016, Paris, France*, volume 60 of *LIPIcs*, pages 17:1–17:12. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[MV18] Andrew McGregor and Sofya Vorotnikova. A simple, space-efficient, streaming algorithm for matchings in low arboricity graphs. In Raimund Seidel, editor, *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*,

volume 61 of *OASICS*, pages 14:1–14:4. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[PS18] Pan Peng and Christian Sohler. Estimating graph parameters from random order streams. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 2449–2466, USA, 2018. Society for Industrial and Applied Mathematics.

[VY11] Elad Verbin and Wei Yu. The streaming complexity of cycle counting, sorting by reversals, and other problems. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 11–25. SIAM, 2011.

A Generating Timestamps in the Stream

In this section we show how an algorithm can generate n timestamps in a streaming manner, corresponding to drawing n uniform random variables from $(0, 1)$ and then presenting each in order with $\text{poly}(1/n)$ precision, using $O(\log n)$ bits of space.

Let $(\mathbf{X}_i)_{i=1}^n$ denote n variables drawn independently from $\mathcal{U}(0, 1)$ and then ordered so that $\mathbf{X}_i \leq \mathbf{X}_{i+1}$ for all $i \in [n-1]$. By standard results on the order statistics (see e.g. page 17 of [Dav03]), the distribution of $(\mathbf{X}_i)_{i=j+1}^n$ depends only on $\mathbf{X}_j$, and in particular they are distributed as drawing $(n - j)$ samples from $(\mathbf{X}_j, 1)$.

So then, to generate $(\mathbf{X}_i)_{i=1}^n$ with $\text{poly}(1/n)$ precision in the stream it will suffice to, at each step $i+1$, use $\mathbf{X}_i$ to generate $\mathbf{X}_{i+1}$ (as sampling from the minimum of k random variables to $\text{poly}(1/n)$ precision can be done in $O(\log n)$ space). We will only need to store one previous variable at a time, to $\text{poly}(1/n)$ precision, and so this algorithm will require only $O(\log n)$ space.