CS 671: Graph Streaming Algorithms and Lower Bounds **Rutgers: Fall 2020**

Lecture 6

October 13th, 2020

Instructor: Sepehr Assadi *Scribe: Chen Wang*

1 Maximum Cardinality Matching

In this lecture, we study the maximum (cardinality) matching problem in the *dynamic graph streams*. In this problem, we have a graph $G = (V, E)$ and the goal is to find a matching with largest number of edges, where a matching is any collection of vertex-disjoint edges.

We have visited maximum matchings for insertion-only graph streams in Lecture 2 (and Lecture 4 for weighted matching). In insertion-only streams, there is a straightforward semi-streaming algorithm that achieves a 2-approximation. In this lecture, we will see that in contrast, approximating matchings in the dynamic streams is quite hard. In particular, with the semi-streaming space complexity of $\tilde{O}(n)$, we cannot get better than a $n^{1/3-o(1)}$-approximation!

To begin with, let us recap the definition of dynamic graph streams:

Definition 1 ([1])**.** A dynamic graph stream $\sigma = \langle \sigma_1, \sigma_2, \cdots, \sigma_T \rangle$ defines a multi-graph $G = (V, E)$ on $|V| = n$ vertices. Each σ_t for $t \in [T]$ is a tuple of (i_t, j_t, Δ_t), where $(u_{i_t}, u_{j_t}) \in E$ and $\Delta_t \in \{-1, +1\}$. The multiplicity of an edge (u_i, u_j) (denote as f_{u_i,u_j}) is defined as:

$$f_{u_i,u_j} = \sum_{\sigma_t : i_t = i; j_t = j} \Delta_t.$$

In dynamic graph streams, we only consider non-negative edge multiplicity.

Intuitively, comparing to the insertion graph streams, dynamic graph streams allow both 'insertions' and 'deletions' of edges. We further note that the dynamic graph streams belong to the family of *turnstile streams*, and this type of turnstile streams with strict non-negative values is also called *strict turnstile streams*.

In this lecture, we will focus on two results to characterize the 'hardness' of the maximum matching problem. More specifically, we show the following results in [4]:

- Any single-pass streaming algorithm that finds an α-approximation of the maximum matching in dynamic graph streams has to use $\Omega(\frac{n^{2-o(1)}}{\alpha^3})$ bits of space.
- Assuming $\alpha = o(n^{\frac{2}{3}})$, there exists a single-pass streaming algorithm that can find an α-approximation for maximum matching in dynamic graph streams and uses $O(\frac{n^2}{\alpha^3})$ bits of space.

The above results show that the $\Theta(\frac{n^2}{\alpha^3})$ space is almost necessary and sufficient. The 'final mile' of the $n^{-o(1)}$ factor was removed by a (very) recent work of Dark and Konrad [5].

2 Preliminaries

We start by introducing the following concepts and techniques that are needed for the rest of this lecture:

- Turnstile Streams and Linear Sketches. There are some *inherent* connections between the two which allows us to unlock new lower bound proof techniques.
- k-player simultaneous communication model. The model is a somewhat different model of computation from the one-way communication model we studied so far and it can lead to stronger lower bounds.
- Ruzsa-Szemerédi Graphs. This type of graph is very useful in proving matching lower bounds, as we have used in Lecture 2.
- ℓ_0-samplers. This is a very powerful tool for dynamic/turnstile streams, which will be the 'backbone' of the algorithms we developed, similar to Lecture 5.

We dedicate the rest of this section to introducing the above concepts and techniques.

Turnstile Streams and Linear Sketches

A turnstile stream is a stream with 'increments' or 'decrements' updates (same way as the dynamic graph stream). Suppose the number of distinct elements in a turnstile stream is m and $f \in \mathbb{R}^m$ denotes the frequency of elements in the stream. A linear sketch of the input stream is characterized as $A \cdot f$, where $A \in \mathbb{N}^{d \times m}$ is a matrix.

Hitherto, most algorithms for turnstile streams rely on the linear sketching technique. This is not a coincidence: Li, Nguyen, and Woodruff [9] proved that all turnstile stream algorithms might as well be linear sketches—this characterization was extended to strict turnstile streams (and dynamic graph streams) by Ai, Hu, Li, and Woodruff [2], if we slightly extend the power of linear sketches: we say an algorithm is maintaining a *strong linear sketch* if it maintains $A \cdot f \mod q$ for some vector $q \in \mathbb{R}^d$.

We give the statement as the follows:

Proposition 2 ([9, 2])**.** *Any single-pass streaming algorithm to approximate a function f in the strict turnstile streaming model can be implemented by a strong linear sketch with* poly-logarithm *space overhead.*

The results in Proposition 2 gives us a novel approach to prove space lower bounds for dynamic graph streams. Since the existence of an algorithm on dynamic graph streams implies a linear sketch algorithm, the *contraposition* must also be true: if we can prove a space lower bound for linear sketch, we can get a space lower bound for any single-pass algorithm on *dynamic* graph streams.

Remark. We shall note the characterization in Proposition 2 places very strong restrictions on the turnstile stream algorithm before it can turned the algorithm into a linear sketch: the algorithm must be able to solve *very* long streams with $2^{2^{O(n)}}$ insertions and deletions, and the number of edges between two vertices in the multi-graph can go to some exponentially large number $2^{O(n)}$ (a recent work of Kallaugher and Price [8] shows that some of these restrictions are in fact necessary). Nevertheless, this gives us the technique to prove *worst-case* lower bounds for dynamic graph streams with linear sketches.

k-player simultaneous communication model

The standard model to prove lower bounds for single-pass streaming algorithm is the one-way communication complexity. However, when it comes to dynamic graph streams and in particular linear sketches, we can exploit a more "friendly" model for proving our lower bounds: the (number-in-hand) *k-player simultaneous communication model*.

Suppose we have a graph $G = (V, E)$ and k players $P^{(1)}, \cdots, P^{(k)}$. We partition the edges of G between these players in an adversarial way. Each player $P^{(i)}$ has a set of edges $x_i = \{0,1\}^{\binom{n}{2}}$, and the overall set of edges is denoted as $x = \sum_{i=1}^{k} x_i$. The players have *shared randomness* (i.e. the public coin), and they can,

simultaneously with each other, send a single message to a coordinator/referee who then computes some function $f(x)$, say, find a spanning forest of the graph defined by G. We are interested in the worst-case communication cost of *each player*, namely, the length of the longest message communicated by any player.

The following statement shows a relation between the k-player simultaneous communication model and the linear sketch lower bounds:

Proposition 3. *Suppose there is a linear sketch algorithm that computes a function f with probability of success at least $\frac{2}{3}$ and space of s bits; then for any $k \geq 1$, there exists a k-party (public-coin) simultaneous protocol to compute f with communication cost of $O(s)$ bits by each player and the same probability of success.*

Proof. Let Φ denote the distribution from which the linear sketch A is sampled. The players use public coin to jointly sample a matrix A from Φ. Then, each player $P^{(i)}$ computes $A \cdot x_i$ and sends it to the coordinator. The coordinator will be able to compute $A \cdot x = A \cdot \sum_{i=1}^{k} x_i = \sum_{i=1}^{k} A \cdot x_i$ using the linearity of sketches, and get the answer to $f(x)$ from $A \cdot x$. This way, the communication cost of each player is proportional to the size of the sketch which is $O(s)$ and the correctness probability remains the same. □

Combining Propositions 2 and 3 gives us a new communication complexity approach for proving lower bounds *specific to* dynamic graph stream algorithms: bounding the per-player communication cost in the simultaneous k-party communication model. We will see this method in action later in this lecture.

Ruzsa-Szemerédi Graphs

As we saw in Lecture 2 also, Ruzsa-Szemerédi Graphs (RS Graphs) played an important role for proving streaming lower bounds in the insertion-only streams. This continues to be the case for dynamic graph streams as well[1]. Intuitively, RS graphs can be described as 'graphs with many disjoint induced matchings of large size.' More formally, an (r, t)-RS graph is a graph with t pairwise disjoint *induced* matchings, each of size of r.

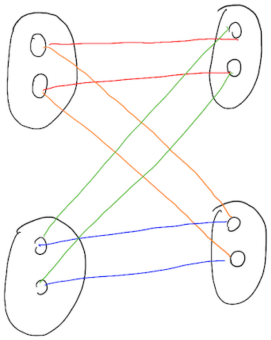

Figure 1: An example for a bipartite $(2, 4)$-RS graph.

Remark. Notice that the requirement for the matchings to be *induced* makes the construction of the RS graph far from the complete graph. Since induced matching cannot have any edge other than the matching edges, a complete graph will only admit matchings of size one. A (trivial) way to construct a bipartite RS graph can be shown as Figure 1. However, in general, one can construct much denser RS

[1] It should be noted however that unlike insertion-only streams where using RS graphs is *crucial* (at least in the one-way communication model; see [6]), for dynamic streams, they *simplify* the lower bound but they are not necessary – see the recent lower bound of [5] that bypasses using these graphs as well as simultaneous communication model and characterization of dynamic stream algorithms as linear sketches.

graphs compared to the naive approach of Figure 1.

In this lecture, we will use the following type of RS graph proved by Alon, Moitra, and Sudakov. [3]:

Proposition 4 ([3]). *For sufficiently large N, there exists an (r,t)-RS graph on N vertices with $r = N^{1-o(1)}$ and $r \cdot t = \binom{N}{2} - o(N^2)$.*

ℓ_0 samplers

Finally, recall ℓ_0-samplers from Lecture 5 that allow us to sample an element uniformly at random from the support of a vector in a dynamic stream. We again use the following construction of ℓ_0-samplers:

Proposition 5 ([7]). *There exists a linear sketch-based randomized algorithm that can perform ℓ_0 sampling on $x \in \mathbb{Z}^n$ with $\text{poly}(n)$-bounded entries using $O(\log^2(n)\log(1/\delta))$ space, and success probability $\geq (1-\delta)$.*

3 $\Omega(\frac{n^{2-o(1)}}{\alpha^3})$ space is Necessary for α-Approximation

We are going to prove a communication lower bound for matching in the simultaneous communication model in this section. As argued earlier, this would then provide a lower bound for dynamic streaming algorithms.

To prove lower bounds for randomized algorithms, we apply Yao's minimax principle to prove lower bounds for deterministic protocols over a fixed input distribution. We start by introducing our hard distribution of inputs first.

3.1 The High Level Construction of a Hard Instance

Since we are going to prove a lower bound with the k-layer simultaneous communication model, a natural idea to design hard instances for maximum matching is to leverage the obliviousness of the players from each other's input. The following is a good way to exploit this property: Suppose the input of each player $P^{(i)}$ is an (r,t)-RS graph G_i. Among the t matchings the player has, assign *one* of them as the *private matching* – the vertices of this matching are dedicated to this player and do not appear in the input of any other player. For the rest of the matchings, let different players *share* the vertices, that is, construct a multi-graph with up to k edges between each pair of matched vertices in the shared matching. In such a construction, considering that locally every player is oblivious to the identity of its private matching, it cannot communicate "a lot" about its private matching unless it communicates a lot about "every matching" in its input. An illustration of this construction can be shown as Figure 2.

To formalize the above strategy, we will assign *labels* to vertices in the construction. Given a permutation π, for each vertex v_j in the public matching, we will assign it $\pi(j)$. That is, shared permutation across different players. In contrast, for each private matching player $P^{(i)}$, we will make the label of vertex v_j as $\pi(N + (i-2)\cdot r + j)$. These seemingly complicated indices (as we will encounter soon) are merely the realization of the 'private-public separation.'

3.2 Warm-up: When Players Only Communicate Edges

Let us first consider the case when the players are only allowed to send edges. Under this setup, the argument becomes simple: fix any player $P^{(i)}$ (we can assume the coordinator already knows the input of other players) and suppose $P^{(i)}$ sends $o(\frac{r\cdot t}{\alpha})$ edges, where α is the desired approximation ratio; then, the expected number of edges in the private matching to be sent to the coordinator is going to be $o(r/\alpha)$. We will show that in the construction, this is not enough to achieve an α approximation.

We now formalize the above intuition with the hard instance and the proof:

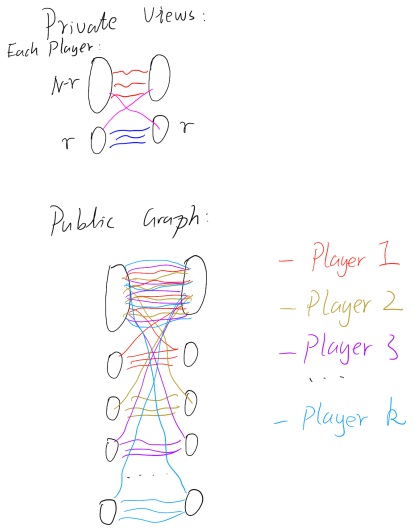

Figure 2: An illustration of the hard instance. On the top is the multi-graph with matching edges from k players. All other matchings are the 'private' matchings of each player. Better view with colors.

The hard distribution for edge-only communication protocols.

- **Parameters:**
$$N = \frac{n}{k} \quad , \quad r = N^{1-o(1)} \quad , \quad t = \frac{\binom{N}{2} - o(N^2)}{r} \quad , \quad k = 5\alpha \cdot \frac{N}{r}.$$
- For each player $P^{(i)}$ for $i \in [k]$ independently:
  - Pick a set of N vertices as V_i and construct a (r,t)-RS graph over V_i.
  - Pick one of the induced matchings as $M_{j^*}^{(i)}$ uniformly at random and let V_i^* be the vertices of $M_{j^*}^{(i)}$.
- Pick a random permutation π for $[n]$ and use it to map vertices of these RS graphs to the final graph G as follows:
  - For every $v_j \in V_i \setminus V_i^*$, label v_j as $\pi(j)$.
  - For every $v_j \in V_i^*$, label v_j as $\pi(N + (i-2) \cdot r + j)$.

We first observe that by carefully picking these parameters, we have $k = \alpha \cdot N^{o(1)}$, the number of overall edges is $O(k \cdot N^2) = \frac{n^{2-o(1)}}{\alpha}$. We have the following two simple yet key observations.

Observation 6. *Any graph G sampled from this distribution has a matching of size at least $r \cdot k$.*

(Proof: take the edges of $M_{j^*}^{(i)}$ for $i \in [k]$ which are $r \cdot k$ edges and form a matching as by construction, each of them is supported on a disjoint sets of vertices.)

Observation 7. *For any single player $P^{(i)}$, conditioned (only) on the graph given to this player (including the labels of its vertices), the choice of j^* is still uniform over $[t]$; in other words, every player is individually oblivious to the identity of the special matching in its input.*

(Proof: by the choice of permutation π, conditioned on vertices of V_i being mapped to a set L of labels, the choice of mapping of V_i^* to L is the same as all other sets of induced matching vertices marginally.)

We now show that with edge-only communication, at least one player needs to communicate $\Omega(\frac{r \cdot t}{\alpha})$ edges for the coordinator to be able to output an α-approximation. Fix any player $P^{(i)}$, and let $X_{j^*}^{(i)}$ be the random variable for the number of edges $P^{(i)}$ sent to the coordinator in its special matching $M_{j^*}^{(i)}$. We prove that,

Claim 8. *Suppose $P^{(i)}$ sends at most $\frac{r \cdot t}{10\alpha}$ edges to the coordinator. Then, $\mathbb{E}[X_{j^*}^{(i)}] \leq \frac{r}{10\alpha}$.*

Proof. We can show this by using the player's obliviousness of the edges in Observation 7. Given the input to player $P^{(i)}$, the set of edges it will communicate will be fixed deterministically. Let C_i denote these edges. However, by Observation 7, the choice of j^* is still uniform over $[t]$ and since the edges of induced matchings partition C_i as well, we have,

$$\mathbb{E}\left[X_{j^*}^{(i)}\right] = \sum_{j=1}^{t} \Pr\left(j = j^*\right) \cdot \mathbb{E}\left[|C_i \cap M_{j^*}^{(i)}| \mid j^* = j\right] = \frac{1}{t} \cdot \sum_{j=1}^{t} \mathbb{E}\,|C_i| \leq \frac{1}{t} \cdot \frac{r \cdot t}{10\alpha} = \frac{r}{10\alpha},$$

as desired. □

Claim 9. *Suppose every player communicates at most $\frac{r \cdot t}{10\alpha}$ edges; then, the coordinator cannot output an α-approximation to the maximum matching with probability at least $\frac{1}{2}$.*

Proof. Let $X = \sum_{i=1}^{k} X_{j^*}^{(i)}$ be the number of edges the coordinator received from all the special matchings. Assuming every player sends at most $\frac{r \cdot t}{10\alpha}$ edges and by Claim 8, we have that

$$\mathbb{E}\left[X\right] \leq \frac{r \cdot k}{10\alpha},$$

which by a Markov bound, implies that with probability $\geq 1/2$, value of X is at most $\frac{r \cdot k}{5\alpha}$. We condition on this event in the following.

In the graph G sampled from the distribution, all edges except for the special matching ones are incident on a set of N vertices. As such, the largest matching the coordinator can output has size $N + X$. Considering both quantities are at most $\frac{r \cdot k}{5\alpha}$, the output matching of coordinator in this case has size $< \frac{r \cdot k}{\alpha}$, which is smaller than an α-factor of maximum matching size in G by Observation 6. □

This warm-up now shows that the players need to communicate $\Omega(\frac{r \cdot t}{10\alpha}) = \Omega(\frac{N^{2-o(1)}}{\alpha}) = \Omega(\frac{n^{2-o(1)}}{\alpha^3})$ edges before they can get an α-approximation with constant probability.

3.3 The Lower Bound for the General Case

We now consider the general case. Firstly, notice that the instance we create for the edge-only communication version is not hard for the players that send arbitrary messages – the players need to simply send the mapping of RS graph vertices to their labels to the coordinator; considering the edges of the RS graph are known to all parties, this is enough for the coordinator to reconstruct the entire graph and thus solve the problem. Basically, in the previous construction, there was only $O(n \log n)$ bits of randomness that the players could communicate to the coordinator; to get a stronger lower bound, we need to increase the randomness in player's inputs as follows.

A standard technique to extend "edge-only lower bounds" to actual lower bounds is to use a simple randomization trick: By uniformly at random dropping some edges from the distribution, we can create many possible graphs as potential inputs to the players. This will then forces the players to communicate much larger messages as otherwise the coordinator may output an edge that does *not* belong to the input graph.

We now formalize the above argument. In the following, we give a simple modification of the hard distribution from the previous section (the only different part is marked in bold).

The hard distribution for the general case .

- **Parameters:**

$$N = \frac{n}{k} \quad , \quad r = N^{1-o(1)} \quad , \quad t = \frac{\binom{N}{2} - o(N^2)}{r} \quad , \quad k = 5\alpha \cdot \frac{N}{r}.$$

- For each player $P^{(i)}$ for $i \in [k]$ independently:
  - Pick a set of N vertices as V_i, construct a (r,t)-RS graph with the above parameters over v_I.
  - Uniformly at random mark one of the matchings as $M_\lambda^{(i)}$, and let V_i^* be the vertices in $M_\lambda^{(i)}$.
  - **For each induced matching, uniformly at random drop half of the edge.**
- Pick a random permutation π for $[n]$, such that for every player's input:
  - For every $v_j \in V_i \setminus V_i^*$, label v_j as $\pi(j)$.
  - For every $v_j \in V_i^*$, label v_j as $\pi(N + (i-2) \cdot r + j)$.

Fix any player $P^{(i)}$, let $\mathcal{G}_i$ be the family of all the possible graphs that this player may receive (before labeling of vertices). By definition, we have that,

$$|\mathcal{G}_i| = \binom{r}{\frac{r}{2}}^t.$$

For any message Π of player $P^{(i)}$, we use $\mathcal{G}_i(\Pi)$ to denote the family of graphs in $\mathcal{G}_i$ encoded to this message. We will show that over the randomness of Π, the size of $\mathcal{G}_i(\Pi)$ is typically "large".

Lemma 10. *Suppose player $P^{(i)}$ sends a message of length s bits. Then, with probability at least $1 - 2^{-s}$ (over the choice of Π),*

$$|\mathcal{G}_i(\Pi)| \geq |\mathcal{G}_i| \cdot 2^{-2s}.$$

Proof. Let $\Pi_1, \ldots, \Pi_{2^s}$ denote all the messages player $P^{(i)}$ may send. Since the distribution of the input graph in $\mathcal{G}_i$ is uniform, the probability that each message Π_j is sent is exactly

$$\Pr\left(P^{(i)} \text{ sends } \Pi_j\right) = |\mathcal{G}_i(\Pi_j)| \cdot |\mathcal{G}_i|^{-1}.$$

Let J denote the indices of messages with a "small" number of encoded graphs, namely,

$$J := \left\{ j \mid |\mathcal{G}_i(\Pi_j)| < |\mathcal{G}_i| \cdot 2^{-2s} \right\}.$$

Combining these two, we have,

$$\begin{aligned} \Pr_{\Pi}\left(|\mathcal{G}_i(\Pi)| < |\mathcal{G}_i| \cdot 2^{-2s}\right) &= \sum_{j \in J} \Pr\left(P^{(i)} \text{ sends } \Pi_j\right) = \sum_{j \in J} |\mathcal{G}_i(\Pi_j)| \cdot |\mathcal{G}_i|^{-1} \\ &< |J| \cdot (|\mathcal{G}_i| \cdot 2^{-2s}) \cdot |\mathcal{G}_i|^{-1} \leq 2^s \cdot 2^{-2s} = 2^{-s}, \end{aligned}$$

concluding the proof. □

Now consider the role of coordinator given the message Π from player $P^{(i)}$: the coordinator can only output an edge e as part of the output matching if e belongs to *all* graphs in $\mathcal{G}_i(\Pi)$; otherwise, there is a chance

that the coordinator outputs an edge that does not even belong to the input graph, a contradiction[2]. We now show that for most messages Π, the number of edges that the coordinator can output from the special matching of player $P^{(i)}$ is going to be small. Similar in spirit to the previous section, let $X_{j^*}^{(i)}$ be the random variable for the number of edges of $P^{(i)}$ in the matching $M_{j^*}^{(i)}$ that belong to $\mathcal{G}_i(\Pi)$ for the message Π.

Lemma 11. *Suppose $P^{(i)}$ sends at most $s = \frac{r\cdot t}{10\alpha}$ bits to the coordinator. Then, $\mathbb{E}[X_{j^*}^{(i)}] \leq \frac{r}{10\alpha}$.*

Proof. For any message Π of $P^{(i)}$, let $m_j(\Pi)$ denote the number of edges in the matching $M_j^{(i)}$ that it fixes, i.e., the edges that belong to all graphs in $\mathcal{G}_i(\Pi)$ and thus the coordinator is allowed to output them. Considering the choice of $j^* \in [t]$ is independent of the message Π (exactly as in Observation 7), we have that

(1)

We can upper bound the size of $\mathcal{G}_i(\Pi)$ using $m_j(\Pi)$ for $j \in [t]$ as follows:

$$|\mathcal{G}_i(\Pi)| \leq \prod_{j=1}^{t} \binom{r - m_j(\Pi)}{r/2},$$

simply because every graph in $\mathcal{G}_i(\Pi)$ has only $r/2$ edges in each of its matchings and none of the matching edges counted in $m_j(\Pi)$ can be dropped. We expand the upper bound on $\mathcal{G}_i(\Pi)$ in the following:

$$|\mathcal{G}_i(\Pi)| \leq \prod_{j=1}^{t} \binom{r - m_j(\Pi)}{r/2} \leq \prod_{j=1}^{t} 2^{-m_j(\Pi)} \cdot \binom{r}{r/2} \quad = 2^{-\sum_{j=1}^{t} m_j(\Pi)} \cdot |\mathcal{G}_i|, \quad (\text{as } \tbinom{a-b}{c} \leq (\tfrac{a-c}{a})^b \cdot \tbinom{a}{c})$$

where the last equality is by the bound on the size of $\mathcal{G}_i$.

$$\begin{aligned}
|\mathcal{G}(\Pi)| &\leq \prod_{i=1}^{t} \binom{r - x_j}{\frac{r}{2}} \\
&\leq \left(\frac{\sum_{i=1}^{t} \binom{r-x_j}{\frac{r}{2}}}{t}\right)^t && \text{(geometric mean} \leq \text{arithmetic mean)} \\
&= \left(\frac{t\binom{r-x}{\frac{r}{2}}}{t}\right)^t \\
& && \text{(set } x := x_1 = x_2 = \cdots = x_t \text{ for geometric mean = arithmetic mean)} \\
&= \binom{r-x}{\frac{r}{2}}^t \\
&\leq (2e)^{\frac{r\cdot t}{2}} \cdot \left(\frac{r-x}{r}\right)^{\frac{rt}{2}} \\
&\leq |\mathcal{G}| \cdot \left(\frac{r-x}{r}\right)^{\frac{rt}{2}} && \text{(by ?? for sufficiently large } r) \\
&= |\mathcal{G}| \cdot \left(1 - \frac{x}{r}\right)^{\frac{r}{x}\cdot\frac{tx}{2}} \\
&= |\mathcal{G}| \cdot \exp\left(-\frac{tx}{2}\right).
\end{aligned}$$

[2]Here, similar to [4], we make this common assumption that the coordinator may err by outputting a smaller-than-α-approximation matching, but not by outputting an edge that does not even belong to the graph. This assumption however can be lifted using a slightly more careful argument.

On the other hand, we have $|\mathcal{G}(\Pi)| \geq 2^{-2s}|\mathcal{G}|$ with high probability. This yields in

$$\begin{aligned}|\mathcal{G}| \cdot \exp(-\frac{tx}{2}) &\geq 2^{-2s} \cdot |\mathcal{G}| \\ &\geq \exp(-2s) \cdot |\mathcal{G}|\end{aligned}$$

Solving this will give us $x \geq \frac{4s}{t}$. Therefore, we have $\sum_{j=1}^{t} x_j = x \cdot t \leq 4s$. Furthermore, since the player is oblivious to the edges, the expectation for x_j on each matching should be the same. Therefore, we conclude $\mathbb{E}[X_\lambda^{(i)}] \leq \frac{4s}{t}$. □

To wrap up, if we send $s = \frac{r \cdot t}{16\alpha}$ bits for each player, the bound in lemma 8 holds since we have $|\mathcal{G}| = \binom{r}{\frac{r}{2}}^t \geq (\frac{1}{2r})^t \cdot 2^{s \cdot \alpha} >> 32 \cdot 2^s$. Now, combing with lemma 9, the expected number of edges the coordinator received from each players is at most $\frac{r}{4\alpha}$. By the same argument of the edge-only communication, the coordinator cannot get a $(\frac{3}{2} \cdot \alpha)$-approximation. Therefore, each player has to send $> \frac{r \cdot t}{16\alpha} = \Omega(\frac{n^{2-o(1)}}{\alpha^3})$ bits to get more than $(\frac{3}{2} \cdot \alpha)$-approximation with success probability $\frac{1}{2} + \frac{1}{4} = \frac{3}{4}$ (the union bound over the 'lucky' cases of size of X_λ and the size of message for $s \geq 10$ bits). We can wrap up and conclude:

Theorem 12. *Any* randomised *protocol for approximating the maximum matching with a factor $\frac{3}{2} \cdot \alpha$ with k players and success probability of $\frac{3}{4}$ has to communicate $\Omega(\frac{n^{2-o(1)}}{\alpha^3})$ bits from at least one player.*

4 Upper bound: $O(\frac{n^2}{\alpha^3})$ space is sufficient for $O(\alpha)$-approximation

We present the corresponding algorithms for the lower bounds in this section. For the convenience of the analysis, we assume the graphs we study here are bipartite, and they have a perfect matching. Also, we assume the the approximation rate α is constrained by $\alpha \leq n^{\frac{1}{2}}$.

Another part to note is that since all the algorithms are based on ℓ_0 samplers, the analysis of the correctness is not necessarily related to the streaming setup. If we can prove a technique of *sampling edges* works on non-streaming graphs, then with the help of the ℓ_0 sampler, we can simply simulate the sampling process during the stream and run maximum flow algorithm to determine the maximum matching by the end. Therefore, in the proof of the correctness, we will inter-exchange the notion of correctness under ℓ_0 *sampler-based algorithms* and correctness by *sampling edges under the non-streaming setup.*

4.1 Random Sampling: An $O(\alpha)$-approximation algorithm with $\tilde{O}(\frac{n^2}{\alpha})$ space

The simplest algorithm with the powerful ℓ_0 sampler will be to randomly sample edges. If we set the sampling probability to $\frac{15}{\alpha}$, the expected number of edges to be sampled in the matching is $\frac{15 \cdot n}{\alpha}$ (assuming perfect matching). And these are independent random variables with values $[0, 1]$, which means we can apply Chernoff bound to get high concentration results. We formalize the above intuition as the follows:

$O(\alpha)$ **approximation − simple** .

- Pre-processing:
  - For each vertices pair $(u_i \in L, v_j \in R)$, assign v_j to u_i with probability $\frac{15}{\alpha}$.
- Streaming updates:
  - Maintaining a ℓ_0 sampler between each assigned pair (u_i, v_j).
- Post-streaming phase:
  - Sample one edge from each maintained ℓ_0-sampler and compute a maximum matching M over the sampled edges.

Lemma 13. *With probability at least $1 - \frac{1}{n}$, algorithm $O(\alpha)$* ***approximation*** *−* ***simple*** *returns an $O(\alpha)$-approximation for the maximum matching in a single pass with $\tilde{O}(\frac{n^2}{\alpha^2})$ space.*

Proof. The space of $O(\frac{n^2}{\alpha})$ is straightforward since we only maintain $\frac{15}{\alpha} \cdot n^2$ ℓ_0 samplers, and each of them only use $O(\text{poly}\log(n))$ bits. The correctness can be shown as the follows: for each edge e_i in the maximum matching M^*, define X_i as the indicator variable for e_i to be sampled. We can further define $X = \sum_{i=1}^{n} X_i$ as the random variable for the number of edges to be sampled from M^*. Observe that $\mathbb{E}[X] = \sum_{i=1}^{n} \mathbb{E}[X_i] = \frac{15n}{\alpha}$. Also, we can show with chernoff bound that

$$\begin{aligned}\Pr(X < \frac{10 \cdot n}{\alpha}) &\leq \exp(-\frac{(\frac{2}{3})^2 \cdot \mathbb{E}(X)}{3}) \\ &\leq \frac{1}{2 \cdot n}\end{aligned}$$

Applying a union bound with the probability of the ℓ_0 samplers concludes the proof. □

4.2 Grouping: An *deterministic* $O(\alpha)$-approximation algorithm with $\tilde{O}(\frac{n^2}{\alpha^2})$ space

For bipartite graph, a natural observation is that for algorithm to get an $O(\alpha)$-approximation, we only need the left side vertices to be matched to unique right side with a rate of *every α vertices*. In this way, we can get $\frac{n}{2\alpha}$ super-vertices (or groups) that are matched to unique right neighbors, and this will be a valid $O(\alpha)$-approximation. We leverage this observation and design the algorithm as the follows:

$O(\alpha)$ **approximation − Grouping** .

- Pre-processing:
  - Parameter: $A = \frac{n}{2\alpha}$
  - Create sets $\mathcal{L}$ and $\mathcal{R}$, which respectively contains A equal-size sets (call them *groups*), and each of them of size α. Group the vertices $u \in L$ and $v \in R$ *deterministically* (maintain the original tier) to create $\mathcal{L}$ and $\mathcal{R}$.
  - For each $L_i \in \mathcal{L}$, assign *every* vertex $R_j \in R$ to it. In other words, we pair each $(L_i, R_j), \forall R_j \in \mathcal{R}$.
- Streaming updates:
  - Maintaining a ℓ_0 sampler between each assigned pair (L_i, R_j).
- Post-streaming phase:
  - Sample one edge from each maintained ℓ_0-sampler and compute a maximum matching M over the sampled edges.

Lemma 14. *Assuming the success of ℓ_0 samplers, algorithm $O(\alpha)$* ***approximation*** *−* ***Grouping*** deterministically *returns an $O(\alpha)$-approximation for the maximum matching in a single pass with $\tilde{O}(\frac{n^2}{\alpha})$ space.*

Proof. Again, the space of $O(\frac{n^2}{\alpha^2})$ is straightforward since we only maintain $\frac{n}{\alpha}$ groups on each side, and this yields in $\frac{n^2}{\alpha^2}$ ℓ_0 samplers, and each of them only use $O(\text{poly}\log(n))$ bits. The correctness follows by the following statement

Claim 15. *Define each group as a super-vertex, and define the multi-graph $G' = (V', E')$ be the graph with V' as the $\frac{n}{\alpha}$ super-vertices and each $\{e_{u,v}\} \in E'$ as the set of α edges between any pair of super-vertices. Assuming the perfect matching for the original graph G, there exists a perfect matching for G'.*

The above claim is a natural corollary of Hall's theorem. To prove this, consider for the purpose of contradiction that there isn't a perfect matching. Therefore, we should have one super vertex on the left not matcheable, and by Hall's theorem, we have $|\mathcal{N}(L')| < |L'|$. However, this also implies $|\mathcal{N}(L)| < |L|$ in the original graph since each edge can be corresponded to one vertex in G. This forms a contradiction.

With the above claim, we can conclude that the ℓ_0 sampler can always return one matching edge in the perfect matching, which proves the correctness. □

4.3 Randomized Grouping: An $O(\alpha)$-approximation algorithm with $\tilde{O}(\frac{n^2}{\alpha^3})$ space

We are now ready to present the optimal algorithm with space $\tilde{O}(\frac{n^2}{\alpha^3})$ space and an $O(\alpha)$-approximation. The idea of the algorithm is, to get an $O(\alpha)$-approximation, we do not need to connect each vertex on the left side to the right. If you form each group uniformly at random, then assigning $\frac{1}{\alpha}$ groups will do us the favor.

$O(\alpha)$ approximation – Optimal .

- Pre-processing:
  - Parameter: $A = \frac{n}{2\alpha}$, $B = \frac{n}{2\alpha^2}\log(n)$, and $\gamma = 4\alpha$
  - Create sets $\mathcal{L}$ and $\mathcal{R}$, which respectively contains A equal-size sets (call them *groups*) for the left vertices, and each of them of size α. Create two γ-wise independent hash functions, and let $u \in L$ hashing to $\mathcal{L}$ and $v \in R$ hashing to $\mathcal{R}$.
  - For each $L_i \in \mathcal{L}$, assign B groups in $\mathcal{R}$ to it (for each (L_i, R_j), we say they are an *active pair*).
- Streaming updates:
  - Maintaining a ℓ_0 sampler between each pair (L_i, R_j).
- Post-streaming phase:
  - Sample one edge from each maintained ℓ_0-sampler and compute a maximum matching M over the sampled edges.

Lemma 16. *The space complexity for algorithm $O(\alpha)$ **approximation – Optimal** is $\tilde{O}(\frac{n^2}{\alpha^3})$.*

Proof. The space complexity comes from two sources: the ℓ_0 samplers and the two 4α-wise independent hash functions. The number of ℓ_0 samplers is $O(\frac{n^2}{\alpha^3})$, and the α-wise independent hash functions can be implemented with $0(\alpha)$ space. The overall space complexity is $\tilde{O}(\alpha + \frac{n^2}{\alpha^3})$; and with the assumption of $\alpha \leq n^{0.5}$, we can get the space complexity of $\tilde{O}(\frac{n^2}{\alpha^3})$. □

To prove the correctness of the algorithm, we need the definition of the 'spanning' groups. Consider the maximum matching M^*, and each $e \in M^*$ is called a *matching edge*. For any pair of groups L_i, R_j, we say they are matcheable if they share at least one matching edge. A group $L_i \in \mathcal{L}$ is said to be spanning if it has matching edges with at least $\frac{1}{3} \cdot \min\{\alpha, \frac{n}{2\alpha}\}$ different groups $R_j \in \mathcal{R}$.

We first show that by randomly hashing vertices on the left to $L_i \in \mathcal{L}$, we will have many spanning groups:

Lemma 17. *With probability at least $\frac{1}{4}$, at least $\frac{1}{3}$ of the $L_i \in \mathcal{L}$ are spanning.*

Proof. The correctness of this lemma relies on the following balls-bins argument:

Claim 18. *If we are given x balls and y bins, and we assign balls to bins uniformly at random. Then, with probability at least $\frac{1}{2}$, the number of non-empty bins is at least $\frac{1}{3} \cdot \min\{x, y\}$.*

We use this claim as a blackbox without proving it. For each group L_i, we can view it as having α 'balls' to be distributed to $\frac{n}{2\alpha}$ 'bins' (groups on the right side). Also recall that each vertex of the left and right sides are hashed 4α-wise independent, which means the 'balls' and 'bins' are sampled uniformly at random and independently. Therefore, with probability at least 0.5, L_i will be spanning. Therefore, the expected number of non-spanning L_i is at most $\frac{1}{2} \cdot \frac{n}{2\alpha} = \frac{n}{4\alpha}$. By a Markov bound, the probability for the non-spanning groups to be more than $\frac{n}{3\alpha}$ is at most $3/4$. That means, with probability at least $1/4$, the fraction of spanning groups will be $\frac{n/2\alpha - n/3\alpha}{n/2\alpha} = \frac{1}{3}$. □

We observe that if the number of neighbors of a group L_i is large, then the probability for it to contain edges in M^* will increase. Specifically, we say a group L_i *preserves* an edge if it has at least one edge $e \in M^*$ that connects to a group $R_j \in \mathcal{R}$. We show that with high probability, the spanning group will preserve at least one matching edge:

Lemma 19. *With probability at least* $1 - \frac{1}{n}$*, every spanning group* $L_i \in \mathcal{L}$ *preserves an edge in* M^**.*

Proof. We show that in both cases (L_i indent to $\frac{\alpha}{3}$ or $\frac{n}{6\alpha}$ groups), the probability for not having matching edge at all is small. Specifically, we can fix a spanning group L_i; if it is indent to $\frac{n}{6\alpha}$ groups, then if we randomly pick a group R_j from $\mathcal{R}$, we have:

$$\Pr(R_j \text{ not mathchable by } L_i) \leq (1 - \frac{n/6\alpha}{n/2\alpha}) = \frac{2}{3}$$

On the other hand, if L_i is indent to $\frac{\alpha}{3}$ groups, the probability becomes:

$$\Pr(R_j \text{ not mathcheable by } L_i) \leq (1 - \frac{\alpha/3}{n/2\alpha}) = (1 - \frac{2}{3} \cdot \frac{\alpha^2}{n})$$

In either case, by raise the quantity to the power of the parameter $B = \frac{n}{2\alpha^2} \log(n)$, the probability of failure is at most $\frac{1}{n^2}$. □

Now we have shown that with constant probability, we will have at least $\frac{1}{3}$ of the groups on the left to have matching edges to the right. There is only one concern remains: it is possible for multiple groups on the left to be indent to the same group on the right. We subsequently show that this is not likely to happen:

Lemma 20. *With constant probability, at least* $\Omega(\frac{n}{\alpha})$ *groups in* $\mathcal{R}$ *should be matchable to* distinct $L_i \in \mathcal{L}$ *groups.*

Proof. To show this lemma, it is sufficient to show that there exists at least $\Omega(\frac{n}{\alpha})$ vertices that are matched by the matching edges to *spanning* groups $L_i \in \mathcal{L}$. If this is true, then the spanning groups are maintaining $\Omega(\frac{n}{\alpha})$ distinct groups in $\mathcal{R}$, which is equivalent to the statement of the lemma.

To prove the above, we can look into the number of matching edges that connect the spanning groups. If $\alpha > \sqrt{\frac{n}{2}}$, then each spanning group $L_i \in \mathcal{L}$ will be matched to $\frac{n}{6\alpha}$ R_i groups in $\mathcal{R}$, and that means we can do a simple Markov bound to get the conclusion with constant probability. Therefore, we are mostly interested in the hard case, when $\alpha \leq \sqrt{\frac{n}{2}}$, where each spanning group $L_i \in \mathcal{L}$ is connected to $\frac{\alpha}{3}$ groups by the matching edges.

Conditioning on the event of Lemma 17, there are at least $\frac{\alpha}{3} \cdot \frac{n}{2\alpha} \cdot \frac{1}{3} = \frac{n}{18}$ edges. We call the graph with these edges as M'. Note that each group in $\mathcal{R}$ has α matching edges. Therefore, by a counting argument, we can say at least $\frac{35}{36}$ fraction of the groups $R_i \in \mathcal{R}$ must have $\frac{\alpha}{35}$ matching edges in M'. To see this, note that $\frac{1}{36}$ fraction of the groups can only provide:

$$\frac{1}{36} \cdot \frac{n}{2\alpha} \cdot \alpha = \frac{n}{72}$$

matching edges to M'. Also, even if the rest of each group provides $\frac{\alpha}{35}$ matching edges, the cumulative matching edges in M' is at most $\frac{n}{2\alpha} \cdot \frac{35}{36} \cdot \frac{\alpha}{35} + \frac{n}{72} = \frac{n}{36} < \frac{n}{18}$.

Denote the above $\frac{35}{36}$ fraction of the groups as $\mathcal{R}'$. Consider, if we sample edges between $R \in \mathcal{R}'$ and spanning group $L_i \in \mathcal{L}$, the probability for a matching edge to be sampled is at least $\frac{1}{\alpha}$. Therefore, for any $R \in \mathcal{R}'$, we have

$$\begin{aligned}\Pr(R \text{ does not have matching edge with any spanning } L \in \mathcal{L}) &\leq (1 - \frac{1}{\alpha})^{\frac{\alpha}{35}} \\ &\leq \exp(-\frac{1}{35})\end{aligned}$$

Hence, in expectation, there are $(1 - \exp(-\frac{1}{35})) \cdot \frac{35}{36} \cdot \frac{n}{\alpha}$ groups in $\mathcal{R}$ that keeps matching edge with distinct spanning $L \in \mathcal{L}$. Applying the Markov bound can give us the conclude with constant probability. □

Finally, we can wrap up the above argument, and conclude the algorithm with the following statement:

Theorem 21. *With high constant probability, algorithm $O(\alpha)$* ***approximation*** *−* ***optimal*** *returns an $O(\alpha)$-approximation for the maximum matching in a single pass with $\tilde{O}(\frac{n^2}{\alpha^3})$ space.*

References

[1] K. J. Ahn, S. Guha, and A. McGregor. Analyzing graph structure via linear measurements. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 459–467. SIAM, 2012. 1

[2] Y. Ai, W. Hu, Y. Li, and D. P. Woodruff. New characterizations in turnstile streams with applications. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 20:1–20:22, 2016. 2

[3] N. Alon, A. Moitra, and B. Sudakov. Nearly complete graphs decomposable into large induced matchings and their applications. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1079–1090, 2012. 4

[4] S. Assadi, S. Khanna, Y. Li, and G. Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1345–1364. SIAM, 2016. 1, 8

[5] J. Dark and C. Konrad. Optimal lower bounds for matching and vertex cover in dynamic graph streams. In *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, pages 30:1–30:14, 2020. 1, 3

[6] A. Goel, M. Kapralov, and S. Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 468–485, 2012. 3

[7] H. Jowhari, M. Saglam, and G. Tardos. Tight bounds for lp samplers, finding duplicates in streams, and related problems. In *PODS*, 2011. 4

[8] J. Kallaugher and E. Price. Separations and equivalences between turnstile streaming and linear sketching. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1223–1236, 2020. 2

[9] Y. Li, H. L. Nguyen, and D. P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 174–183, 2014. 2