Approximate Matchings in Dynamic Graph Streams

Sanjeev Khanna

University of Pennsylvania

Joint work with Sepehr Assadi (Penn), Yang Li (Penn), and Grigory Yaroslavtsev (Penn).

The Streaming Model

Introduced in the seminal work of [Alon, Matias, Szegedy'96].

- Input is presented as a data stream, for instance, as a sequence of edges in case of a graph input.
- Algorithm sees the entire input once but has only a small space to store information about the input.
- At the end of the sequence, the algorithm outputs a solution using the stored information.

Power of Streaming Algorithms

- Surprisingly, many interesting things can be computed by streaming algorithms using a very small amount of space.

- Two key ingredients to tackle adversarial arrangement of data in small space: randomization and approximation.

This talk: sub-linear space streaming algorithms for computing approximate matchings.

Matchings in the Streaming Model

How much space is needed to compute an α–approximate matching?

Insertions-only Streams

- Edges of the graph arrive one by one in a stream.

Dynamic Graph Streams

- Edges of the graph are inserted/deleted one by one in a stream, where no edge is deleted before it is inserted.

We will focus only on single-pass streams.

Matching in Graph Streams

Insertion-only streams:

- Exact computation requires $\Omega(n^2)$ space [Feigenbaum, et.al '05].
- 2-approximation in $O(n)$ space is trivial but no better than 2-approximation is known in $o(n^2)$ space.
  - In the random order model, $(2-\delta)$-approximation ($\delta \approx 0.02$) in $O(n)$ space [Konrad, Maginez, Mathew '12].
- Beating $(e/e-1)$-approximation requires $n^{1+\Omega(1/\log\log n)}$ space [Goel, Kapralov, K '12],[Kapralov '13].

Dynamic graph streams:

- Until recently, no non-trivial results were known for single-pass dynamic streams.

A Toy Problem

Suppose you are given a sequence of edge insert and delete operations.

How much space do you need to output a matching of size one in the remaining graph?

Linear Sketches

- For a graph G with n vertices:
  - Let f denote the n^2-dimensional vector of edge multiplicities.
  - Let A be an $r \times n^2$-dimensional matrix (possibly randomly chosen) for some parameter r.
  - We refer to Af as a linear sketch of G – this is an r-dimensional vector.
  - Space needed to store the graph is reduced from $O(n^2)$ to $O(r)$.

Linear Sketches

Application to dynamic graph streams

- Algorithm dynamically maintains a linear sketch Af of the graph as it is being revealed.
- On each update, i.e, insertion or deletion of an edge e: $Af = Af \pm A1_e$.
- Space requirement is $O(r)$ (+ random bits for implicitly storing A).
- At the end of the stream, the algorithm applies an arbitrary function to Af, to compute the final answer.

Essentially all existing dynamic graph streaming algorithms are linear sketching algorithms.

Our Results

We study the power of linear sketching algorithms for approximating matchings in dynamic graph streams.

- For any $0 < \varepsilon \leq 1/2$, there is an $\tilde{O}(n^{2-3\varepsilon})$ space randomized linear sketching algorithm to compute an n^{ε}-approximate matching in dynamic graph streams. For each edge insertion/deletion, the update time is $\tilde{O}(1)$.

- For any $\varepsilon > 0$, any (randomized) linear sketch that can be used to recover an n^{ε}-approximate maximum matching requires $n^{2-3\varepsilon-o(1)}$ space.

Recent Related Work

Two recent results obtained independently and concurrently.

- [Konrad '15] For (randomized) linear sketches of n^{ε}-approximate maximum matching:
  - $O(n^{2-2\varepsilon})$ space is sufficient.
  - $\Omega(n^{3/2 - 4\varepsilon})$ space is necessary.

- [Chitnis, et.al '15] For any $0 < \varepsilon \leq 1/2$, there is an $O(n^{2-3\varepsilon})$-space randomized linear sketching algorithm to compute an n^{ε}-approximate matching.
  - Update time is $O(n)$ in contrast to $\tilde{O}(1)$ update time achieved in our work.

n^{ε}-Approximation for Matchings

Theorem For any $0 < \varepsilon \leq 1/2$, there is an $\tilde{O}(n^{2-3\varepsilon})$ space algorithm to find an n^{ε}-approximate matching in dynamic graph streams.

- The algorithm maintains a linear sketch.
- We can restrict our attention to bipartite graphs w.l.o.g.
- For simplicity, assume there is a perfect matching M^* in the input bipartite graph $G(L, R, E)$.

ℓ_0-Sampler

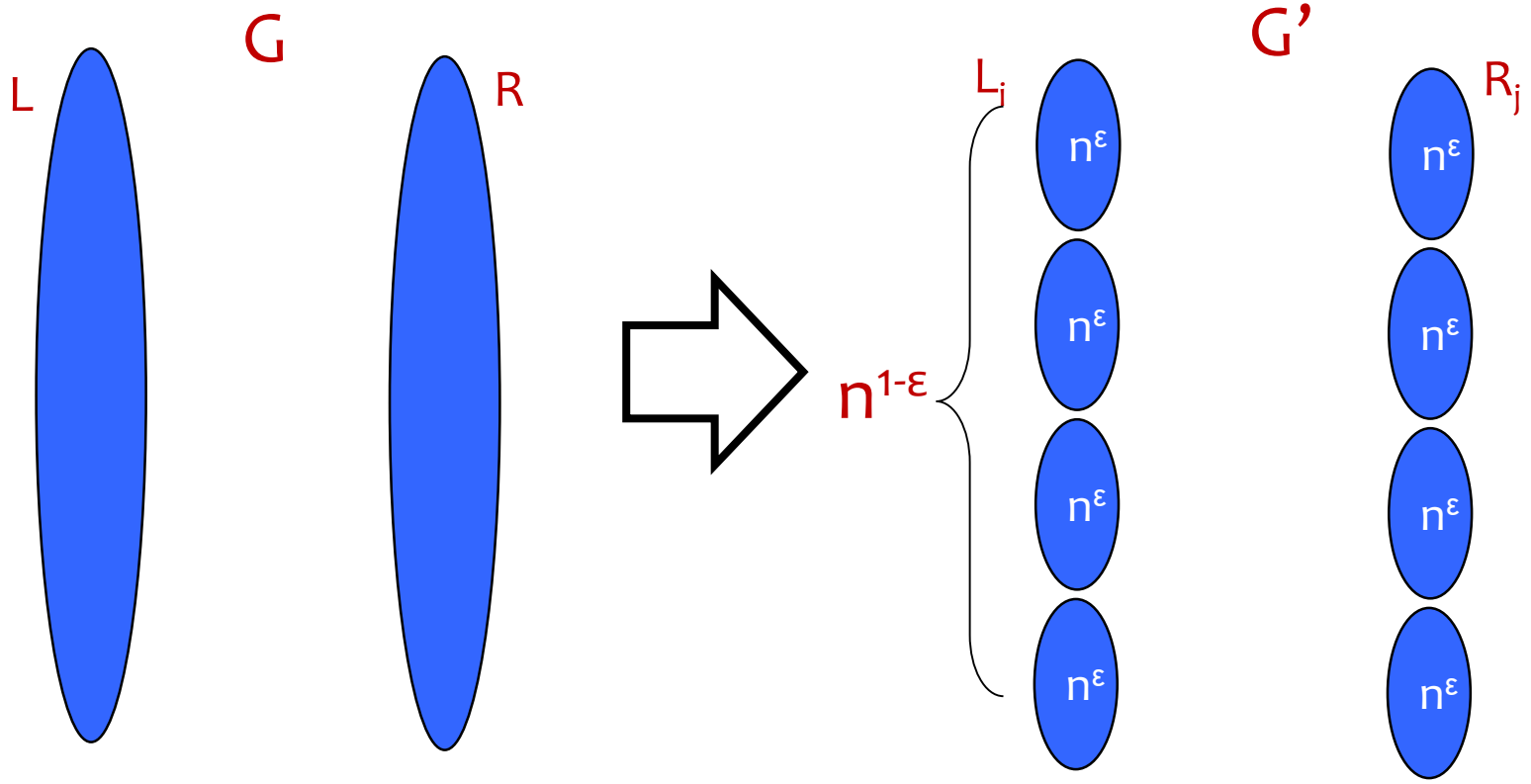
Input: A stream of insertions and deletions over a set of elements (e.g., edges of a graph).
Goal: Among all the elements whose ℓ_0-norm of the multiplicities is nonzero, output one uniformly at random.

Theorem [Jowhari, Sağlam, Tardos '11] For any $0 < \delta < 1$, there is a linear sketching implementation of ℓ_0-sampler for a set of n elements, with probability of success $1 - \delta$, using $O(\log^2 n \cdot \log(\delta^{-1}))$ bits of space.

Plan: Maintain a sample of edges that has a large matching.

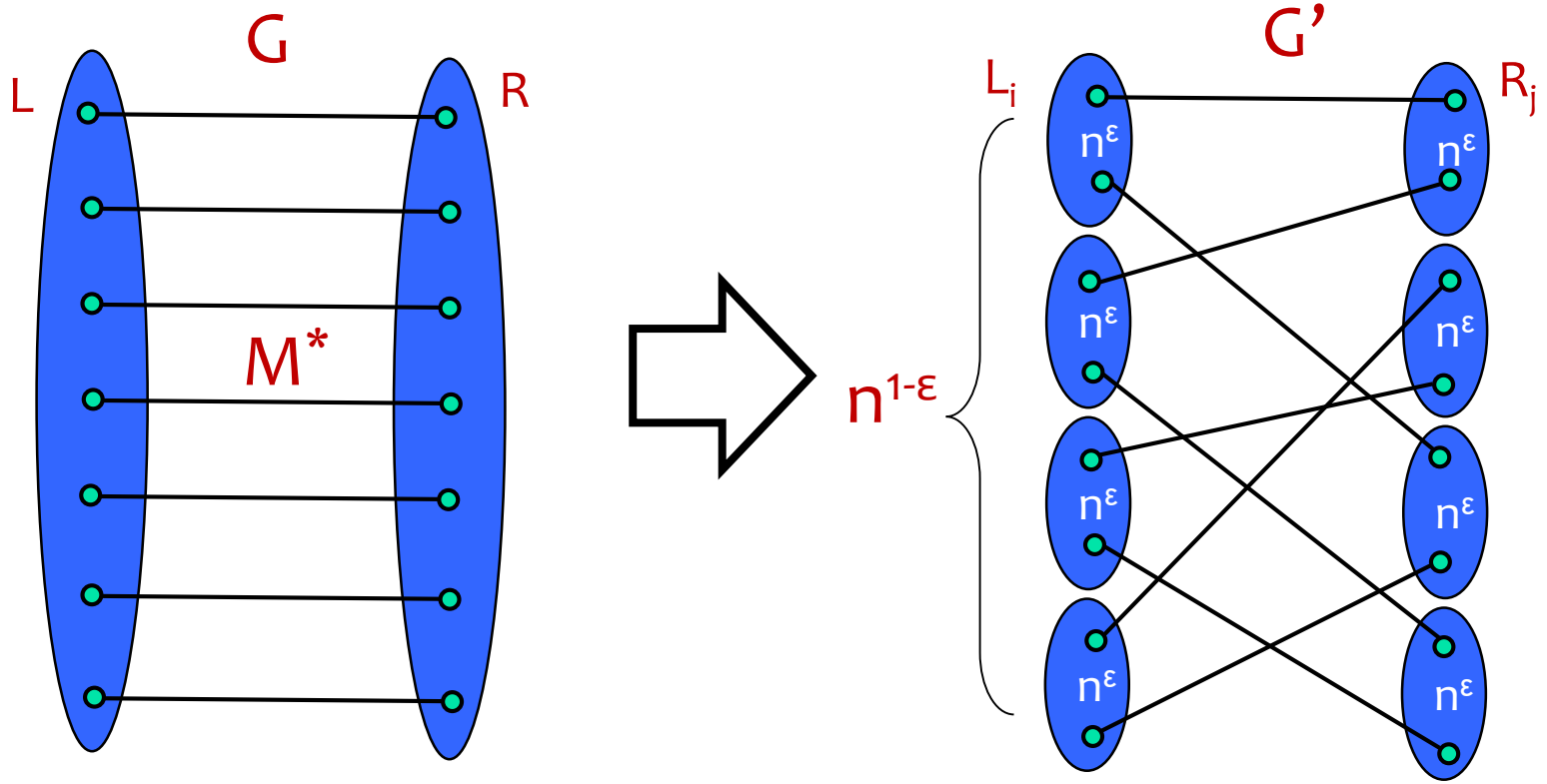
Warm-up: an $\tilde{O}(n^{2-2\varepsilon})$ space Algorithm

- Randomly group the vertices in L (resp. R) into groups L_i 's (resp. R_j 's)of size n^{ε} each. Treat each group as a vertex -- this leads to a new graph G'.



Warm-up: an $\tilde{O}(n^{2-2\varepsilon})$ space Algorithm

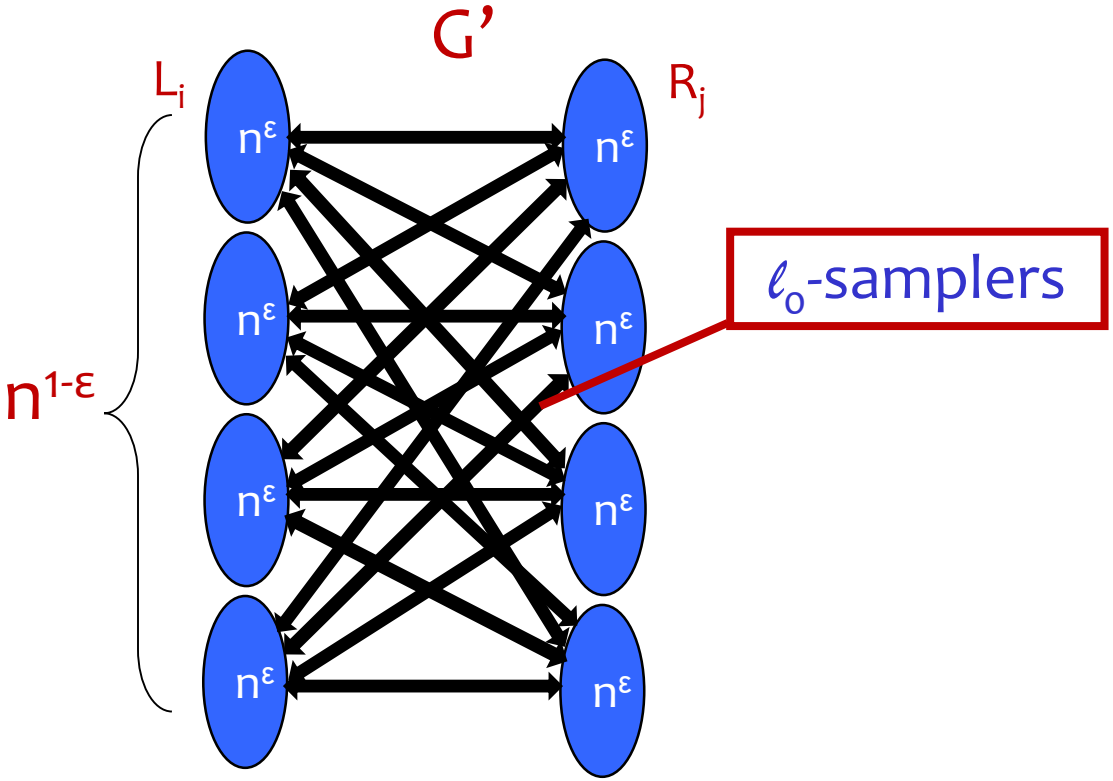
G' has a perfect matching: the perfect matching M^* in G forms an n^{ε}-regular bipartite (multi-)graph in G' and hence G' must contain a perfect matching of size $n^{1-\varepsilon}$.



Warm-up: an $\tilde{O}(n^{2-2\varepsilon})$ space Algorithm

Find a perfect matching in G': For each pair of groups, maintain an ℓ_0-sampler for edges between them.

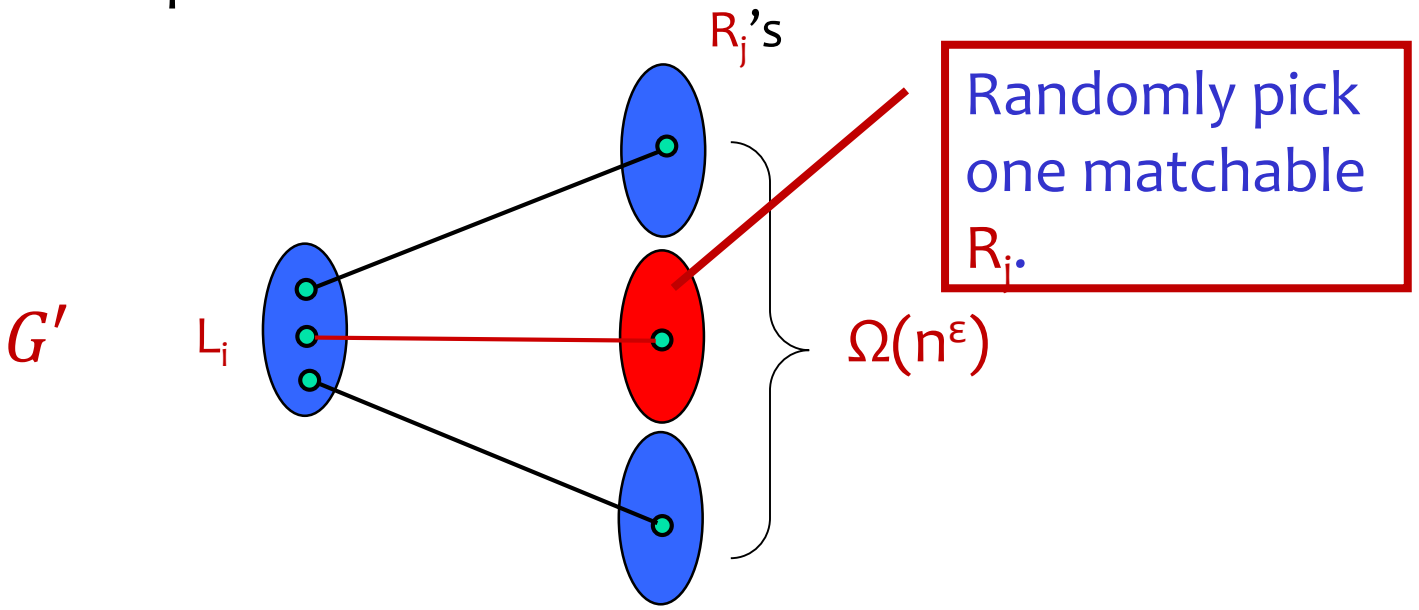
- This requires $\tilde{O}(n^{2-2\varepsilon})$ space.
- Note that so far, *random* grouping was not necessary.



Improving to $\tilde{O}(n^{2-3\varepsilon})$ space

For each L_i, it suffices to find one R_j uniformly at random from the R_j's that are matchable to L_i (connected by an edge in M^*).

- For the n^{ε}-regular bipartite graph induced by M^*, for each vertex, picking one neighbor uniformly at random gives a matching of size $\Omega(n^{1-\varepsilon})$.
- How do we implement this?



Improving to $\tilde{O}(n^{2-3\varepsilon})$ space

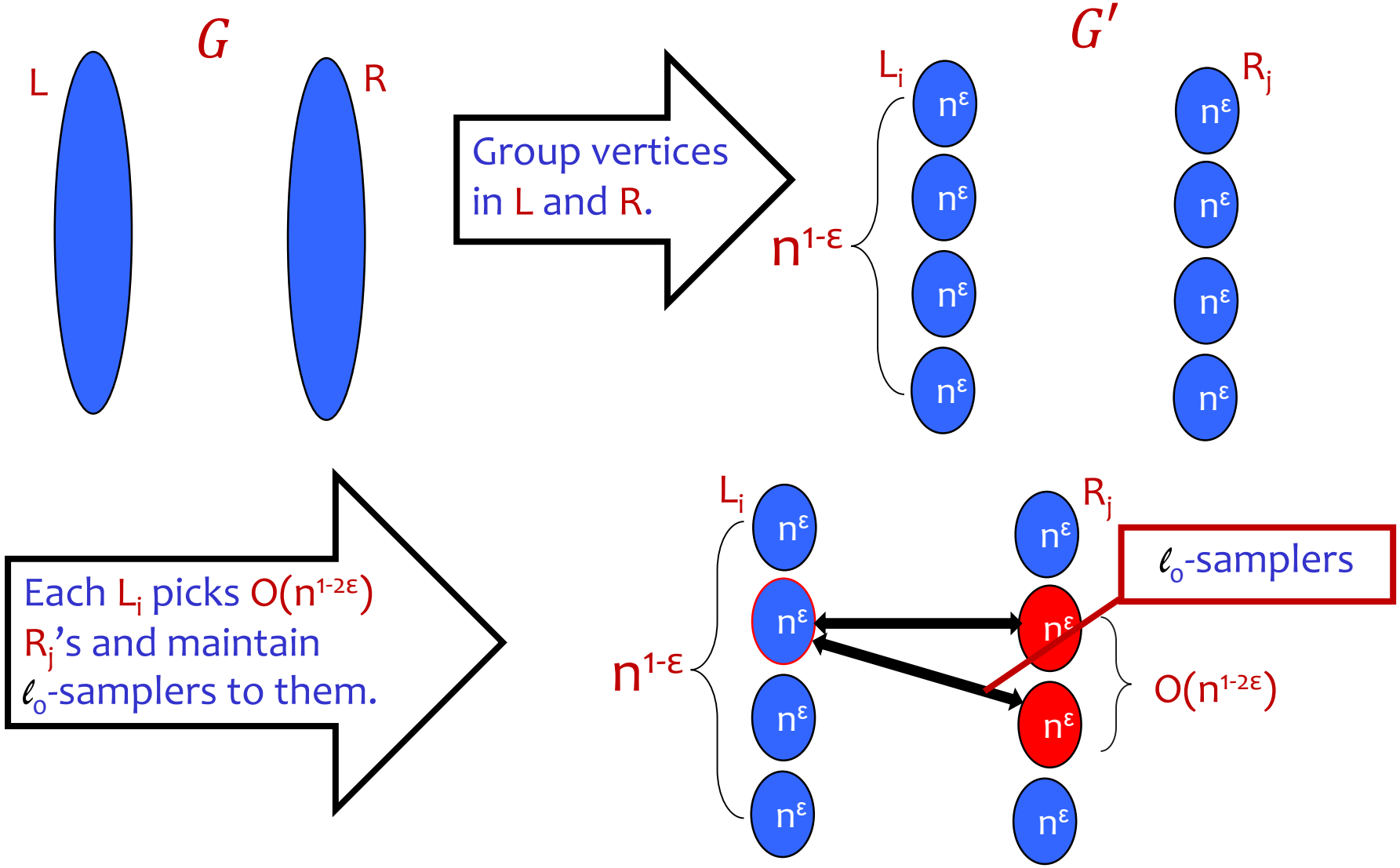
Algorithm:

- For each L_i, pick $O(n^{1-2\varepsilon})$ R_j's uniformly at random, and maintain an ℓ_0-sampler for the edges between L_i and each picked R_j.

Analysis:

- For each L_i, $\Omega(n^{\varepsilon})$ R_j's are matchable.
- Conditioned on the event that L_i picked at least one matchable R_j, the matchable R_j chosen by L_i is uniformly at random.
- For each L_i, choosing $O(n^{1-2\varepsilon})$ R_j's uniformly at random ensures that with constant probability at least one matchable R_j is picked.

Recap: An $\tilde{O}(n^{2-3\varepsilon})$ Space Algorithm



Lower Bound for n^{ε}-Approximation

Theorem. For any $\varepsilon > 0$, any randomized linear sketch that can be used to recover an n^{ε}-approximate matching of a bipartite graph requires $n^{2-3\varepsilon-o(1)}$ space.

Our Approach

- We prove this lower bound, using simultaneous communication complexity:
  - The graph is partitioned between k players $P^1,\ldots,P^k$.
  - There exists another party, called the coordinator.
  - Players $P^1,\ldots,P^k$ simultaneously send a message to the coordinator.
  - Communication measure: maximum # of bits sent by any player.
  - Players have access to public random coins.

A communication lower bound in this model implies an identical space lower bound for linear sketching algorithms.

Connection to Linear Sketches

Proposition [folklore]. If there exists a linear sketch A of size s for a problem P, then simultaneous communication complexity of P is at most $O(s)$.

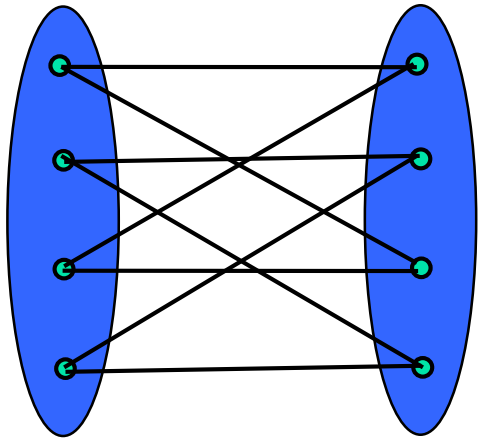
Proof.

1. Players construct A using public random coins.
2. Let x_i denote the input of player P^i. Each player P^i computes $A(x_i)$ and sends it to the coordinator.
3. Coordinator computes $A(x) = A(x_1 + \ldots + x_k) = A(x_1) + \ldots + A(x_k)$ (by linearity) and uses $A(x)$ to solve P.

Ruzsa-Szemerédi graphs

(r,t)-RS graphs: A graph whose edges can be partitioned into t induced matchings of size r.

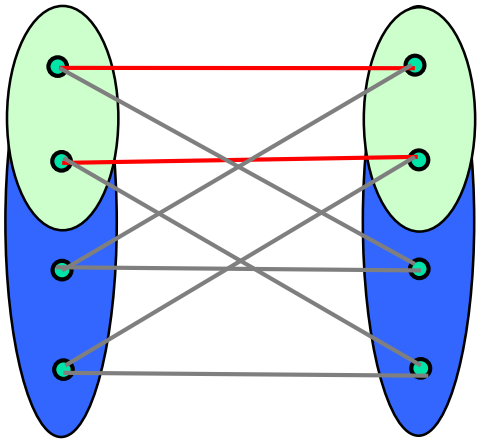
Example: A (2,4)-RS graph on 8 vertices.



Ruzsa-Szemerédi Graphs

(r,t)-RS graphs: A graph whose edges can be partitioned into t induced matchings of size r each.

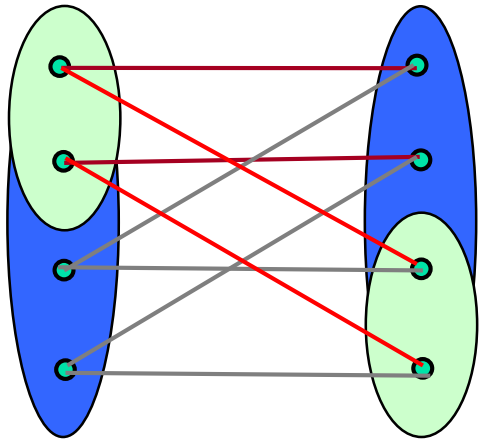
Example: A (2,4)-RS graph on 8 vertices.



Ruzsa-Szemerédi Graphs

(r,t)-RS graphs: A graph whose edges can be partitioned into t induced matchings of size r each.

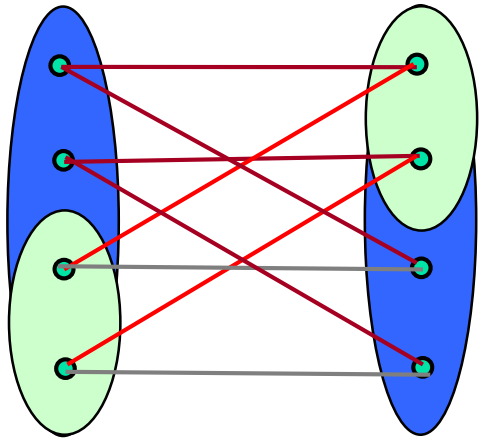
Example: A (2,4)-RS graph on 8 vertices.



Ruzsa-Szemerédi Graphs

(r,t)-RS graphs: A graph whose edges can be partitioned into t induced matchings of size r each.

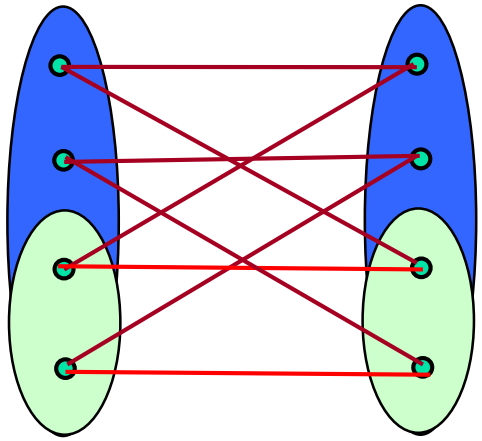
Example: A (2,4)-RS graph on 8 vertices.



Ruzsa-Szemerédi Graphs

(r,t)-RS graphs: A graph whose edges can be partitioned into t induced matchings of size r each.

Example: A (2,4)-RS graph on 8 vertices.



Ruzsa-Szemerédi Graphs

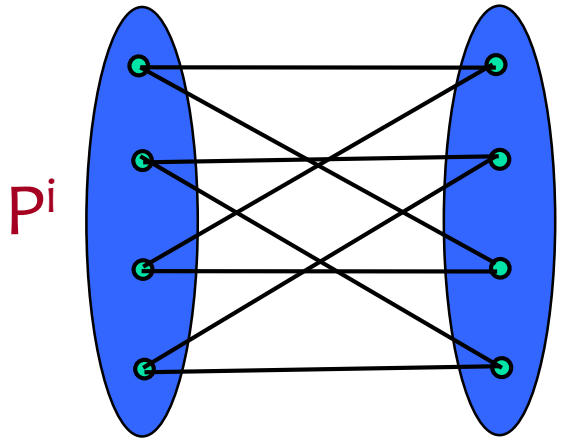
Theorem [Alon, Moitra, Sudakov '12] There exists an (r,t)-RS graph on N vertices and $\Omega(N^2)$ edges with $r = N^{1-o(1)}$ and $t = \Omega(N)$.

$n^{2-3\varepsilon-o(1)}$ Lower Bound: Distribution

($k = n^{\varepsilon+o(1)}$, $N \approx n^{1-\varepsilon}$, $r = N^{1-o(1)}$, $t = \Omega(N)$.)

Hard distribution:

1. Each of the k players is given an (r,t)-RS graph on N vertices with half the edges dropped randomly.



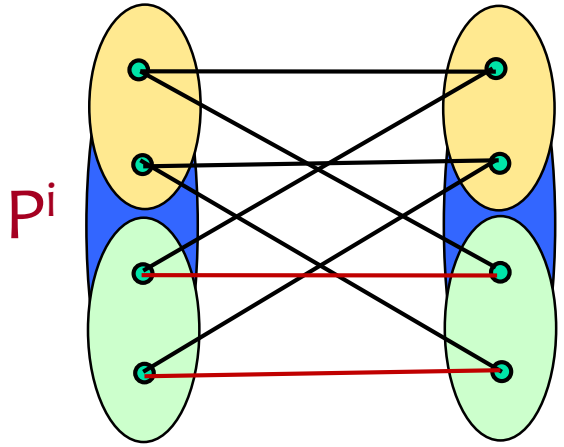
Local view

$n^{2-3\varepsilon-o(1)}$ Lower Bound: Distribution

($k = n^{\varepsilon+o(1)}$, $N \approx n^{1-\varepsilon}$, $r = N^{1-o(1)}$, $t = \Omega(N)$.)

Hard distribution:

1. Each of the k players is given an (r,t)-RS graph on N vertices with half the edges dropped randomly.
2. One of the induced matchings (red edges) is special, unknown to the player.



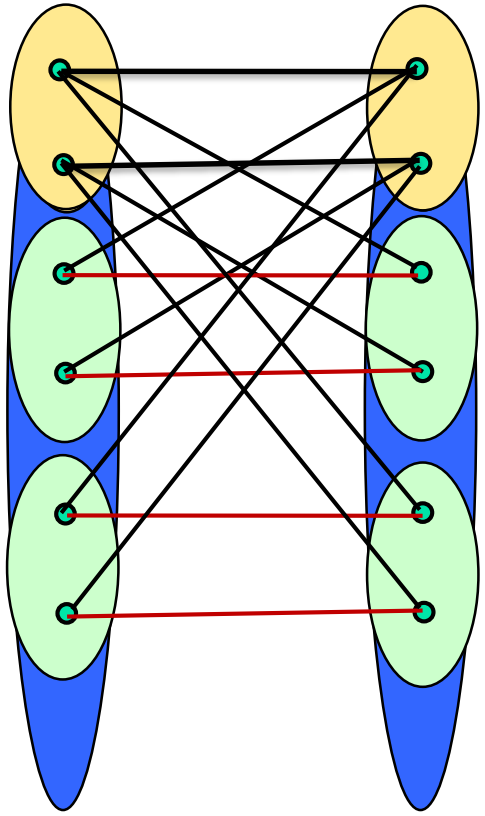
Hidden matching

$n^{2-3\varepsilon-o(1)}$ Lower Bound: Distribution

($k = n^{\varepsilon+o(1)}$, $N \approx n^{1-\varepsilon}$, $r = N^{1-o(1)}$, $t = \Omega(N)$.)

Hard distribution:

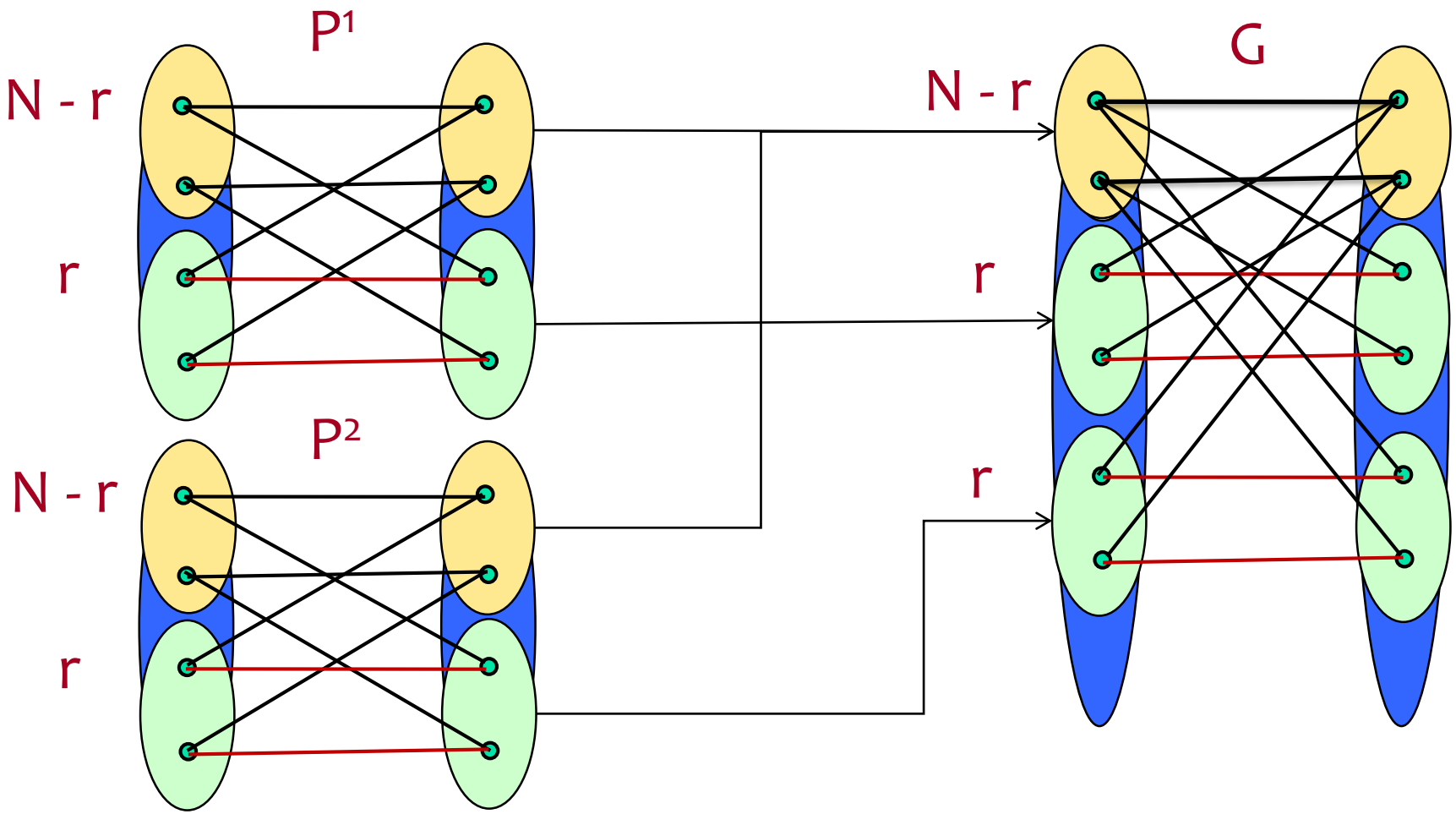
1. Each of the k players is given an (r,t)-RS graph on N vertices with half the edges dropped randomly.
2. One of the induced matchings (red edges) is special, unknown to the player.
3. Across the players, vertices in the special matchings are unique, while other vertices are shared.



Global view

$n^{2-3\varepsilon-o(1)}$ Lower Bound: Distribution

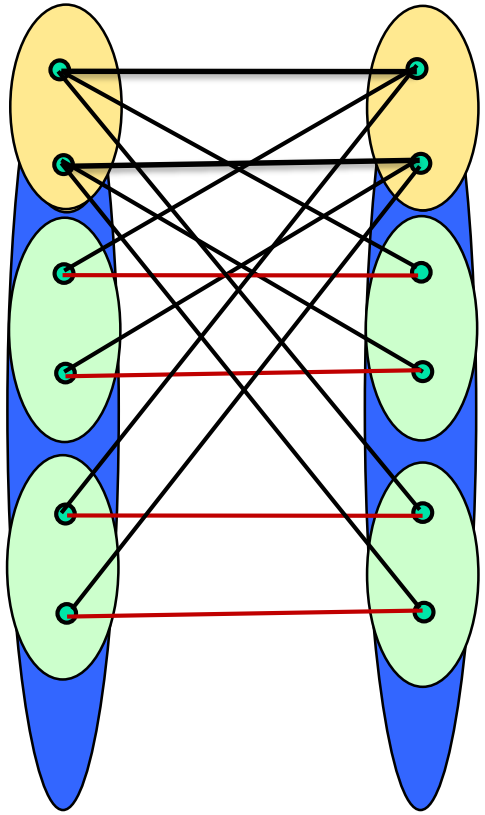
($k = n^{\varepsilon+o(1)}$, $N \approx n^{1-\varepsilon}$, $r = N^{1-o(1)}$, $t = \Omega(N)$.)



$n^{2-3\varepsilon-o(1)}$ Lower Bound: Proof Sketch

($k = n^{\varepsilon+o(1)}$, $N \approx n^{1-\varepsilon}$, $r = N^{1-o(1)}$, $t = \Omega(N)$.)

- Each player's graph has $n^{2-2\epsilon}$ edges.
- If the max communication budget is restricted to $o(n^{2-3\epsilon})$, then each player has to compress their graph by more than n^{ϵ} factor.
- Since players are oblivious to the identity of their special matching, the compression discards more than $1/n^{\epsilon}$-fraction of edges in special matching.



Global view

Concluding Remarks

- For any $0 < \varepsilon \leq 1/2$, we showed that $n^{2-3\varepsilon \pm o(1)}$ space is both sufficient and necessary for any linear sketching algorithm that computes an $O(n^{\varepsilon})$-approximate maximum matching in dynamic graph streams.

- For any $1/2 < \varepsilon \leq 1$, $n^{1-\varepsilon \pm o(1)}$ space is both sufficient and necessary for any linear sketching algorithm that computes an $O(n^{\varepsilon})$-approximate maximum matching in dynamic graph streams.

Concluding Remarks

- Recent work of [Li, Nguyen, Woodruff '14] and of [Ai, Hu, Woodruff '15] show that our lower bounds also imply a lower bound for any dynamic graph streaming algorithm.

- Combined together, these results resolve space complexity of approximating matchings in single-pass dynamic graph streams.

- Is there a sublinear space single-pass algorithm that gives better than a 2-approximation for insertions-only stream?

Thank you