Streaming Algorithms

**Objective:** compute some function $f(x_1, \ldots, x_n)$ given only sequential access

How much RAM is required for the computation of $f$?

**Applications:** Massive data sets (e.g. stored on external memory)

Sequential access vs. random access in streaming algorithms.
Graph Streams (1999 -)

- Input stream: Sequence of edges of input graph $G = (V, E)$ with $n = |V|$ in arbitrary order
  $$S = e_2 e_1 e_4 e_3$$

- Goal: Few passes (preferably one) algorithms with space $o(n^2)$
- Matchings, independent sets, cuts, graph sparsifiers, random walks, bipartiteness testing, counting triangles/subgraphs, ... 

**Maximum Matching in Graph Streams:**
Matchings in Graph Streams

Graph Streams (1999 -)

- Input stream: Sequence of edges of input graph $G = (V, E)$ with $n = |V|$ in arbitrary order
  
  \[ S = e_2 e_1 e_4 e_3 \]

- Goal: Few passes (preferably one) algorithms with space $o(n^2)$
- Matchings, independent sets, cuts, graph sparsifiers, random walks, bipartiteness testing, counting triangles/subgraphs, \ldots

Maximum Matching in Graph Streams: Greedy Algorithm

- Insert $e$ into initially empty matching $M$ if $M \cup \{e\}$ is a matching
- One-pass $\frac{1}{2}$-approximation streaming algorithm with space $O(n \log n)$
Most Studied Graph Problem in the Streaming Model

Unweighted/weighted, one-pass/multi-pass, adversarial arrival order/random order

Matching in Graph Streams (2)

**Most Studied Graph Problem in the Streaming Model**

Unweighted/weighted, one-pass/multi-pass, adversarial arrival order/random order


**Open Question:** Can we beat 1/2 in one pass?
Relaxations of the One-pass Adversarial Order Model

Today:

**GREEDY** is best one-pass algorithm known, even with space $O(n^{2-\epsilon})$

Relaxations of the One-pass Model: (bipartite graphs)

- **Random Order:** Edges arrive in uniform random order
  $(1/2 + 0.005)$-approximation [Konrad et al., APPROX 2012]

- **Two Passes:** adversarial order
  $(1/2 + 0.083)$-approximation [Esfandiari et al., ICDMW 2016]

Main Technique: Improve **GREEDY** matching

1. $M \leftarrow$ **GREEDY** matching
2. $F \leftarrow$ additional edges
3. return $M$ augmented with edges from $F$
Today:

**GREEDY** is best one-pass algorithm known, even with space $O(n^{2-\epsilon})$

**Relaxations of the One-pass Model:** (bipartite graphs)

- **Random Order:** Edges arrive in uniform random order
  
  $(1/2 + 0.005)$-approximation [Konrad et al., APPROX 2012]

- **Two Passes:** adversarial order
  
  $(1/2 + 0.083)$-approximation [Esfandiari et al., ICDMW 2016]

**Main Technique:** Improve **GREEDY** matching (Two passes)

1. $M \leftarrow **GREEDY** matching (first pass)
2. $F \leftarrow$ additional edges (second pass)
3. **return** $M$ augmented with edges from $F$
Today:

**GREEDY** is best one-pass algorithm known, even with space $O(n^{2-\epsilon})$

Relaxations of the One-pass Model: (bipartite graphs)

- **Random Order**: Edges arrive in uniform random order
  
  
  $(1/2 + 0.005)$-approximation [Konrad et al., APPROX 2012]

- **Two Passes**: adversarial order
  
  
  $(1/2 + 0.083)$-approximation [Esfandiari et al., ICDMW 2016]

Main Technique: Improve **GREEDY** matching (Random order)

1. $M \leftarrow$ **GREEDY** matching (first third of edges)
2. $F \leftarrow$ additional edges (remaining edges)
3. return $M$ augmented with edges from $F$
Our Results

Main Result: New Augmentation Method

\[ G = (A, B, E) \text{ bipartite, } M \leftarrow \text{GREEDY}(G), M^* \text{ maximum matching} \]

There is a random subgraph \( H \subseteq G \) that depends on \( M \) such that w.h.p. \( M \cup \text{GREEDY}(H) \) contains a matching of size

\[
(2 - \sqrt{2}) \left| M^* \right| - o(\left| M^* \right|).
\]

0.5857

Applications:

- Two-pass Streaming: 0.5857-approximation
  (improving on 0.583 [Esfandiari et al., ICDMW 2016])
- One-pass Random Order Streaming: 0.5395-approximation
  (improving on 0.505 [Konrad et al., APPROX 2012])
New Augmentation Method for Bipartite Graphs
$G = (A, B, E)$ bipartite, $M \leftarrow \text{GREEDY}(G)$, $M^*$ maximum matching

Observations:
New Augmentation Method for Bipartite Graphs

\[ G = (A, B, E) \text{ bipartite, } M \leftarrow \text{GREEDY}(G), \ M^* \text{ maximum matching} \]

Observations:
- \( M \) is maximal
New Augmentation Method for Bipartite Graphs

\[ G = (A, B, E) \text{ bipartite, } M \leftarrow \text{GREEDY}(G), \ M^* \text{ maximum matching} \]

Observations:
- \( M \) is maximal
New Augmentation Method for Bipartite Graphs

\[ G = (A, B, E) \text{ bipartite, } M \leftarrow \text{GREEDY}(G), \ M^* \text{ maximum matching} \]

**Observations:**
- \( M \) is maximal
- \( M \oplus M^* \): Set of augmenting paths
New Augmentation Method for Bipartite Graphs

\[ G = (A, B, E) \text{ bipartite, } M \leftarrow \text{GREEDY}(G), \ M^* \text{ maximum matching} \]

Observations:
- \( M \) is maximal
- \( M \oplus M^* \): Set of augmenting paths
- \( G_L := G[A(M) \cup \overline{B(M)}] \)
  \( G_R := G[A(M) \cup B(M)] \)

Lemma:
\( G_L \) and \( G_R \) contain matchings of size
\[ |\text{Aug}| = |M^*| - |M|. \]
\[ G = (A, B, E) \text{ bipartite, } M \leftarrow \text{GREEDY}(G), \ M^* \text{ maximum matching} \]

**Observations:**
- \( M \) is maximal
- \( M \oplus M^* \): Set of augmenting paths
- \( G_L := G[A(M) \cup \overline{B(M)}] \)
  \( G_R := G[A(M) \cup B(M)] \)

**Lemma:**
\( G_L \) and \( G_R \) contain matchings of size
\[ |\text{Aug}| = |M^*| - |M|. \]

**Goal:** Compute matchings \( M_L \) in \( G_L \) and \( M_R \) in \( G_R \) s.t. many edges of \( M \) are incident to edges in both \( M_L \) and \( M_R \)
New Augmentation Method for Bipartite Graphs

$G = (A, B, E)$ bipartite, $M \leftarrow \text{GREEDY}(G)$, $M^*$ maximum matching

**Observations:**
- $M$ is maximal
- $M \oplus M^*$: Set of augmenting paths
- $G_L := G[A(M) \cup \overline{B(M)}]$  
  $G_R := G[A(M) \cup B(M)]$

**Lemma:**
$G_L$ and $G_R$ contain matchings of size $|Aug| = |M^*| - |M|$.

**Goal:** Compute matchings $M_L$ in $G_L$ and $M_R$ in $G_R$ s.t. many edges of $M$ are incident to edges in both $M_L$ and $M_R$
New Augmentation Method for Bipartite Graphs

\[ G = (A, B, E) \text{ bipartite}, \ M \leftarrow \text{GREEDY}(G), \ M^* \text{ maximum matching} \]

Observations:
- \( M \) is maximal
- \( M \oplus M^* \): Set of augmenting paths
- \[ G_L := G[A(M) \cup \overline{B(M)}] \]
- \[ G_R := G[A(M) \cup B(M)] \]

Lemma:
\( G_L \) and \( G_R \) contain matchings of size
\[ |Aug| = |M^*| - |M|. \]

Goal: Compute matchings \( M_L \) in \( G_L \) and \( M_R \) in \( G_R \) s.t. many edges of \( M \) are incident to edges in both \( M_L \) and \( M_R \).
First Attempt: No coordination between $M_L$ and $M_R$

- $M_L \leftarrow \text{GREEDY}(G_L)$, $M_R \leftarrow \text{GREEDY}(G_R)$
First Attempt: No coordination between $M_L$ and $M_R$

- $M_L \leftarrow \text{Greedy}(G_L)$, $M_R \leftarrow \text{Greedy}(G_R)$
- Half of $M$ has left wings, other half of $M$ has right wings:

![Diagram](image.png)
**First Attempt:**** No coordination between $M_L$ and $M_R$

- $M_L \leftarrow \text{Greedy}(G_L)$, $M_R \leftarrow \text{Greedy}(G_R)$
- Half of $M$ has left wings, other half of $M$ has right wings:

![Diagram showing a matching with left and right wings]

**Observation:** If $M_L$ and $M_R$ were better than $\frac{1}{2}$-approximations then if $|M| \approx \frac{1}{2} |M^*|$ then some edges of $M$ have both left and right wings
**First Attempt**: No coordination between $M_L$ and $M_R$

- $M_L \leftarrow \text{GREEDY}(G_L)$, $M_R \leftarrow \text{GREEDY}(G_R)$
- Half of $M$ has left wings, other half of $M$ has right wings:

![Diagram](image)

**Observation**: If $M_L$ and $M_R$ were better than $\frac{1}{2}$-approximations then if $|M| \approx \frac{1}{2}|M^*|$ then some edges of $M$ have both left and right wings

**Problems**:
- **GREEDY** only guarantees a $\frac{1}{2}$-approximation
- Our overall goal is to obtain a $\frac{1}{2}$-approximation algorithm...
New Augmentation Method

Main Idea:
- Attempt to augment only a random subset of $M$
- $M' \subseteq M$ sample where every $e \in M$ is included with prob. $\sqrt{2} - 1$
- Proceed as before

![Diagram]

**Theorem:** [Konrad et al., APPROX 2012]

$G = (A, B, E)$, $A' \subseteq A$ uniform random sample with probability $p$. Then:

$$\text{Greedy}(G[A' \cup B]) \geq p \cdot 1 + p \cdot |M^*|$$

Greedy is better than $\frac{1}{2}$ when considering a subset of one bipartition!
Main Idea:
- Attempt to augment only a random subset of $M$
- $M' \subseteq M$ sample where every $e \in M$ is included with prob. $\sqrt{2} - 1$
- Proceed as before

**Theorem:** [Konrad et al., APPROX 2012]

$G = (A, B, E)$, $A' \subseteq A$ uniform random sample with probability $p$. Then:

$\mathbb{E}_{A'}[\text{Greedy}(G[A' \cup B])] \geq p + p|M^*|.$

Greedy is better than $\frac{1}{2}$ when considering a subset of one bipartition!
Main Idea:
- Attempt to augment only a random subset of $M$
- $M' \subseteq M$ sample where every $e \in M$ is included with prob. $\sqrt{2} - 1$
- Proceed as before

\[ G = (A, B, E), A' \subseteq A \text{ uniform random sample with probability } p. \]

Then:

\[ E_{A'} | \text{Greedy}(G[A' \cup B]) \geq p^{1 + p} |M^*|. \]

Greedy is better than $\frac{1}{2}$ when considering a subset of one bipartition!
Main Idea:
- Attempt to augment only a random subset of $M$
- $M' \subseteq M$ sample where every $e \in M$ is included with prob. $\sqrt{2} - 1$
- Proceed as before

\[ \text{Theorem: [Konrad et al., APPROX 2012]} \]

\[ G = (A, B, E), \quad A' \subseteq A \text{ uniform random sample with probability } p. \]

\[ E_{A'} \mid \text{Greedy}(G[A' \cup B]) \geq p + p |M^*|. \]

Greedy is better than $1/2$ when considering a subset of one bipartition!
Main Idea:
- Attempt to augment only a random subset of $\mathbf{M}$
- $\mathbf{M}' \subseteq \mathbf{M}$ sample where every $e \in \mathbf{M}$ is included with prob. $\sqrt{2} - 1$
- Proceed as before

Theorem: [Konrad et al., APPROX 2012]

$G = (A, B, E)$, $A' \subseteq A$ uniform random sample with probability $p$. Then:

$$\mathbb{E}_{A'} [\text{Greedy}(G[A' \cup B])] \geq \frac{p}{1 + p} |\mathbf{M}^*| .$$

Greedy is better than $1/2$ when considering a subset of one bipartition!
Two-pass Streaming Algorithm: 0.5857-approximation

- **First pass:** $M \leftarrow \text{GREEDY}(G)$
- Sample $M' \subseteq M$
- **Second pass:** Compute matchings $M_L$ and $M_R$
- **return** $M$ augmented with $M_L \cup M_R$

Comments:

- Theorem by [Konrad et al., APPROX 2012] only holds in expectation
- We give a martingale-based analysis that shows that a similar result also holds with high probability
- Much simpler and more efficient than [Esfandiari et al., ICDMW 2016]
- Second augmentation round gives 0.6067-approximation (three passes), improving on 0.605-approx. by [Esfandiari et al., ICDMW 2016]
One Pass Random Order Streaming Algorithm
Algorithm by [Konrad et al., APPROX 2012]: 0.505-approximation

- If \textsc{Greedy} performs poorly then it converges quickly
- Roughly $\frac{2}{3} m$ edges for finding 3-augmenting paths

Improvements:

1. Use our new augmentation method
2. Enough to run \textsc{Greedy} on first $\frac{1}{\log n}$-fraction

More edges available for finding augmenting paths!
**Residual Sparsity Property of Greedy**

**Residual Sparsity Lemma:** Run `GREEDY` on $\frac{|E|}{\log n}$ random edges. Then residual graph has at most $O(n \log^2 n)$ edges.

*Residual graph:* Edges that can be added to the matching

Distributed Computing, Dynamic Algorithms, Streaming Algorithms, …

**Algorithm:** 0.5395-approximation

1. $M \leftarrow \text{GREEDY}(\pi[1, \frac{m}{\log n}])$
2. If $M$ close to maximal then employ new augmentation method
3. Else store $O(n \log^2 n)$ residual edges $E'$, compute $\text{OPT}$ in $M \cup E'$

![Graph with residual edges highlighted in red.]
Residual Sparsity Property of Greedy

**Residual Sparsity Lemma:** Run \textsc{Greedy} on \( \frac{|E|}{\log n} \) random edges. Then residual graph has at most \( O(n \log^2 n) \) edges.

*Residual graph:* Edges that can be added to the matching

Distributed Computing, Dynamic Algorithms, Streaming Algorithms, ...

**Algorithm:** 0.5395-approximation

1. \( M \leftarrow \textsc{Greedy}(\pi[1, \frac{m}{\log n}]) \)
2. If \( M \) close to maximal \textbf{then} employ new augmentation method
3. Else store \( O(n \log^2 n) \) residual edges \( E' \), compute \( \text{OPT} \) in \( M \cup E' \)
Residual Sparsity Property of Greedy

**Residual Sparsity Lemma:** Run $\text{GREEDY}$ on $\frac{|E|}{\log n}$ random edges. Then residual graph has at most $O(n \log^2 n)$ edges.

*Residual graph:* Edges that can be added to the matching.

Distributed Computing, Dynamic Algorithms, Streaming Algorithms, …

**Algorithm:** 0.5395-approximation

1. $M \leftarrow \text{GREEDY}(\pi[1, \frac{m}{\log n}])$
2. If $M$ close to maximal then employ new augmentation method
3. Else store $O(n \log^2 n)$ residual edges $E'$, compute OPT in $M \cup E'$

![Graph diagram](image)
Summary

Results:
- Simple augmentation method that only requires running \textsc{Greedy} on a random subgraph
- Improvement over all known streaming algorithms for matchings that operate in few passes

Open Questions:
- Improve on \textsc{Greedy} in one pass adversarial order setting?
- Exploit additional properties of random order \textsc{Greedy}?
- [Assadi et al., arXiv 2018] recently gave a $\frac{2}{3}$-approximation random order algorithm with space $\tilde{O}(n\sqrt{n})$. Can we achieve a $\frac{2}{3}$-approximation in space $\tilde{O}(n)$?
Summary

Results:
- Simple augmentation method that only requires running GREEDY on a random subgraph
- Improvement over all known streaming algorithms for matchings that operate in few passes

Open Questions:
- Improve on GREEDY in one pass adversarial order setting?
- Exploit additional properties of random order GREEDY?
- [Assadi et al., arXiv 2018] recently gave a $\frac{2}{3}$-approximation random order algorithm with space $\tilde{O}(n\sqrt{n})$. Can we achieve a $\frac{2}{3}$-approximation in space $\tilde{O}(n)$?

Thank you