A Simple Augmentation Method for Matchings with Applications to Streaming Algorithms

MFCS 2018

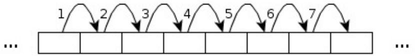
Christian Konrad

University of Bristol

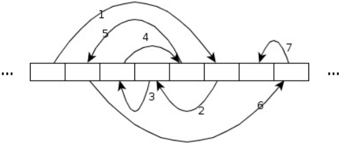
27.08.2018

Streaming Algorithms

sequential access



random access



Streaming (1996 -)

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

  How much RAM is required for the computation of f?

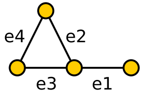
- **Applications:** Massive data sets (e.g. stored on external memory)

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, ...

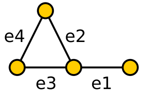
Maximum Matching in Graph Streams:

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, ...

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 $\mathrm{O}(n \log n)$

Matching in Graph Streams (2)

Most Studied Graph Problem in the Streaming Model

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

[Feigenbaum et al., Theo. Comp. Sci. 2005], [McGregor, APPROX 2005], [Epstein et al., STACS 2010], [Ahn, Guha, ICALP 2011], [Eggert et al., Algorithmica 2012], [Konrad et al., APPROX 2012], [Goel et al., SODA 2012], [Zelke, Algorithmica 2012], [Kapralov, SODA 2013], [Crouch, Stubbs, APPROX 2014], [Kapralov et al., SODA 2014], [Esfandiari et al., SODA 2015], [Konrad, ESA 2015], [Assadi et al., SODA 2016], [Kale et al., APPROX 2017], [Cormode et al., ESA 2017] . . .

Matching in Graph Streams (2)

Most Studied Graph Problem in the Streaming Model

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

[Feigenbaum et al., Theo. Comp. Sci. 2005], [McGregor, APPROX 2005], [Epstein et al., STACS 2010], [Ahn, Guha, ICALP 2011], [Eggert et al., Algorithmica 2012], [Konrad et al., APPROX 2012], [Goel et al., SODA 2012], [Zelke, Algorithmica 2012], [Kapralov, SODA 2013], [Crouch, Stubbs, APPROX 2014], [Kapralov et al., SODA 2014], [Esfandiari et al., SODA 2015], [Konrad, ESA 2015], [Assadi et al., SODA 2016], [Kale et al., APPROX 2017], [Cormode et al., ESA 2017] . . .

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 $\mathrm{O}(n^{2-\epsilon})$

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

Relaxations of the One-pass Adversarial Order Model

Today:

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

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

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 (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

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

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

$$\underbrace{(2 - \sqrt{2})}_{0.5857} |M^*| - o(|M^*|) \ .$$

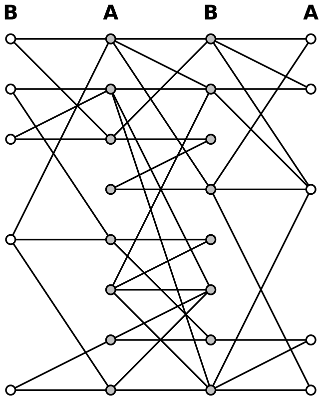
Applications:

- 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

New Augmentation Method for Bipartite Graphs

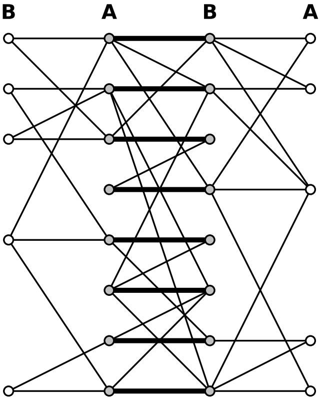
$G = (A, B, E)$ bipartite, $\mathbf{M} \leftarrow \textsc{Greedy}(G)$, $\mathbf{M}^*$ maximum matching



Observations:

New Augmentation Method for Bipartite Graphs

$G = (A, B, E)$ bipartite, $\mathbf{M} \leftarrow \textsc{Greedy}(G)$, $\mathbf{M}^*$ maximum matching

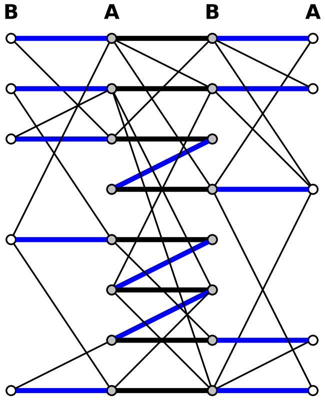


Observations:

- **M** is maximal

New Augmentation Method for Bipartite Graphs

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

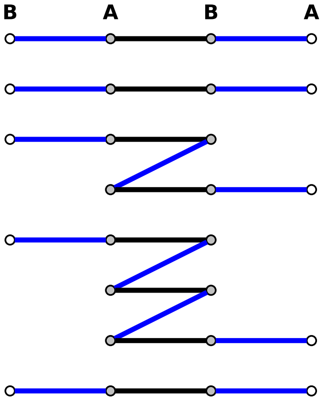


Observations:

- **M** is maximal

New Augmentation Method for Bipartite Graphs

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

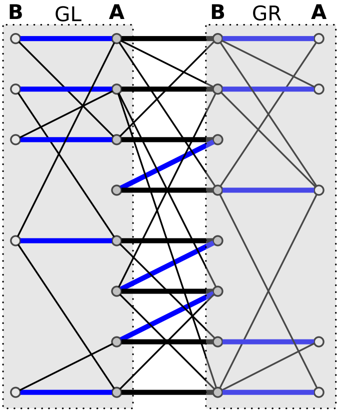


Observations:

- $\mathbf{M}$ is maximal
- $\mathbf{M} \oplus \mathbf{M}^*$: Set of augmenting paths

New Augmentation Method for Bipartite Graphs

$G = (A, B, E)$ bipartite, $\mathbf{M} \leftarrow \textsc{Greedy}(G)$, $\mathbf{M}^*$ maximum matching



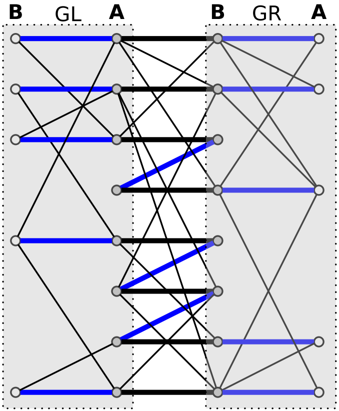
Observations:

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

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

New Augmentation Method for Bipartite Graphs

$G = (A, B, E)$ bipartite, $\mathbf{M} \leftarrow \textsc{Greedy}(G)$, $\mathbf{M}^*$ maximum matching



Observations:

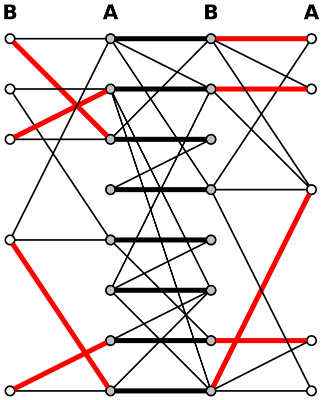
- $\mathbf{M}$ is maximal
- $\mathbf{M} \oplus \mathbf{M}^*$: Set of augmenting paths
- $G_L := G[\underline{A(\mathbf{M})} \cup \overline{B(\mathbf{M})}]$
  $G_R := G[\overline{A(\mathbf{M})} \cup B(\mathbf{M})]$

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

Goal: Compute matchings $\mathbf{M_L}$ in G_L and $\mathbf{M_R}$ in G_R s.t. many edges of $\mathbf{M}$ are incident to edges in both $\mathbf{M_L}$ and $\mathbf{M_R}$

New Augmentation Method for Bipartite Graphs

$G = (A, B, E)$ bipartite, $\mathbf{M} \leftarrow \textsc{Greedy}(G)$, $\mathbf{M}^*$ maximum matching



Observations:

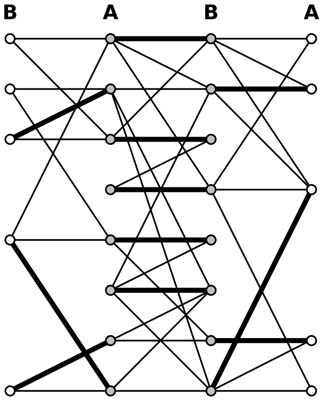
- $\mathbf{M}$ is maximal
- $\mathbf{M} \oplus \mathbf{M}^*$: Set of augmenting paths
- $G_L := G[\underline{A(\mathbf{M})} \cup \overline{B(\mathbf{M})}]$
  $G_R := G[\overline{A(\mathbf{M})} \cup B(\mathbf{M})]$

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

Goal: Compute matchings $\mathbf{M_L}$ in G_L and $\mathbf{M_R}$ in G_R s.t. many edges of $\mathbf{M}$ are incident to edges in both $\mathbf{M_L}$ and $\mathbf{M_R}$

New Augmentation Method for Bipartite Graphs

$G = (A, B, E)$ bipartite, $\mathbf{M} \leftarrow \textsc{Greedy}(G)$, $\mathbf{M}^*$ maximum matching



Observations:

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

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

Goal: Compute matchings $\mathbf{M_L}$ in G_L and $\mathbf{M_R}$ in G_R s.t. many edges of $\mathbf{M}$ are incident to edges in both $\mathbf{M_L}$ and $\mathbf{M_R}$

First Attempt

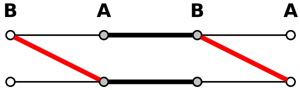
First Attempt: No coordination between $\mathbf{M_L}$ and $\mathbf{M_R}$

- $\mathbf{M_L} \leftarrow \textsc{Greedy}(G_L)$, $\mathbf{M_R} \leftarrow \textsc{Greedy}(G_R)$

First Attempt

First Attempt: No coordination between $\mathbf{M_L}$ and $\mathbf{M_R}$

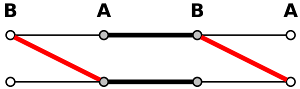
- $\mathbf{M_L} \leftarrow \textsc{Greedy}(G_L)$, $\mathbf{M_R} \leftarrow \textsc{Greedy}(G_R)$
- Half of **M** has left wings, other half of **M** has right wings:



First Attempt

First Attempt: No coordination between $\mathbf{M_L}$ and $\mathbf{M_R}$

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

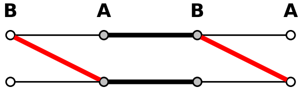


Observation: If $\mathbf{M_L}$ and $\mathbf{M_R}$ were better than $\frac{1}{2}$-approximations then if $|\mathbf{M}| \approx \frac{1}{2}|\mathbf{M}^*|$ then some edges of $\mathbf{M}$ have both left and right wings

First Attempt

First Attempt: No coordination between $\mathbf{M_L}$ and $\mathbf{M_R}$

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



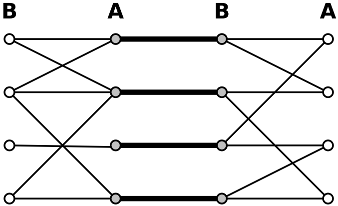
Observation: If $\mathbf{M_L}$ and $\mathbf{M_R}$ were better than $\frac{1}{2}$-approximations then if $|\mathbf{M}| \approx \frac{1}{2}|\mathbf{M}^*|$ then some edges of **M** have both left and right wings

Problems:

- $\textsc{Greedy}$ only guarantees a $\frac{1}{2}$-approximation
- Our overall goal is to obtain a $> \frac{1}{2}$-approximation algorithm...

Main Idea:

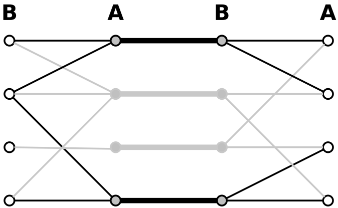
- Attempt to augment only a random subset of $\mathbf{M}$
- $\mathbf{M}' \subseteq \mathbf{M}$ sample where every $e \in \mathbf{M}$ is included with prob. $\sqrt{2} - 1$
- Proceed as before



New Augmentation Method

Main Idea:

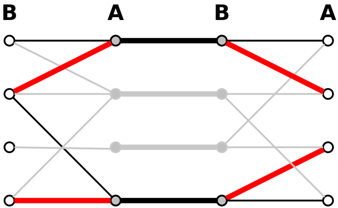
- Attempt to augment only a random subset of $\mathbf{M}$
- $\mathbf{M}' \subseteq \mathbf{M}$ sample where every $e \in \mathbf{M}$ is included with prob. $\sqrt{2} - 1$
- Proceed as before



New Augmentation Method

Main Idea:

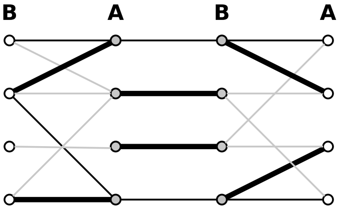
- Attempt to augment only a random subset of **M**
- $\mathbf{M}' \subseteq \mathbf{M}$ sample where every $e \in \mathbf{M}$ is included with prob. $\sqrt{2} - 1$
- Proceed as before



New Augmentation Method

Main Idea:

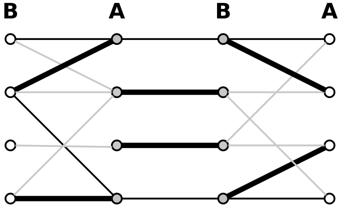
- Attempt to augment only a random subset of $\mathbf{M}$
- $\mathbf{M}' \subseteq \mathbf{M}$ sample where every $e \in \mathbf{M}$ is included with prob. $\sqrt{2} - 1$
- Proceed as before



New Augmentation Method

Main Idea:

- Attempt to augment only a random subset of **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:

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

$\textsc{Greedy}$ is better than $1/2$ when considering a subset of one bipartition!

New Augmentation Method (2)

Two-pass Streaming Algorithm: 0.5857-approximation

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

Comments:

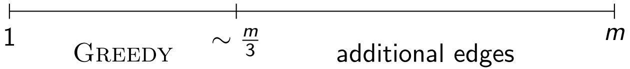
- Theorem by **[Konrad et al., APPROX 2012]** only holds in expectation
- We give a martingale-based analysis that shows that a similar result also holds with high probability
- Much simpler and more efficient than **[Esfandiari et al., ICDMW 2016]**
- Second augmentation round gives 0.6067-approximation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

One Pass Random Order Streaming Algorithm

Random Order

Algorithm by [Konrad et al., APPROX 2012]: 0.505-approximation

- If 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 GREEDY on first $\frac{1}{\log n}$-fraction

More edges available for finding augmenting paths!

Residual Sparsity Property of Greedy

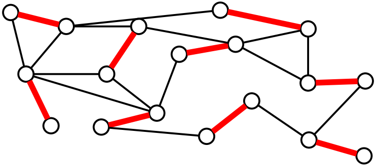
Residual Sparsity Lemma: Run $\textsc{Greedy}$ on $\frac{|E|}{\log n}$ random edges. Then residual graph has at most $\mathrm{O}(n \log^2 n)$ edges

Residual graph: Edges that can be added to the matching

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

Algorithm: 0.5395-approximation

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



Residual Sparsity Property of Greedy

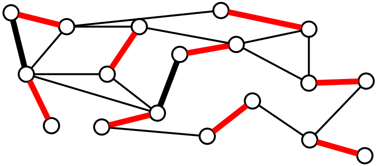
Residual Sparsity Lemma: Run $\textsc{Greedy}$ on $\frac{|E|}{\log n}$ random edges. Then residual graph has at most $\mathrm{O}(n \log^2 n)$ edges

Residual graph: Edges that can be added to the matching

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

Algorithm: 0.5395-approximation

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



Residual Sparsity Property of Greedy

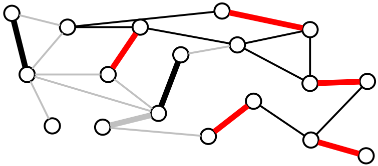
Residual Sparsity Lemma: Run $\textsc{Greedy}$ on $\frac{|E|}{\log n}$ random edges. Then residual graph has at most $\mathrm{O}(n \log^2 n)$ edges

Residual graph: Edges that can be added to the matching

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

Algorithm: 0.5395-approximation

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



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 $\tilde{O}(n)$?

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 $\tilde{\mathrm{O}}(n\sqrt{n})$. Can we achieve a $\frac{2}{3}$-approximation in space $\tilde{\mathrm{O}}(n)$?

Thank you