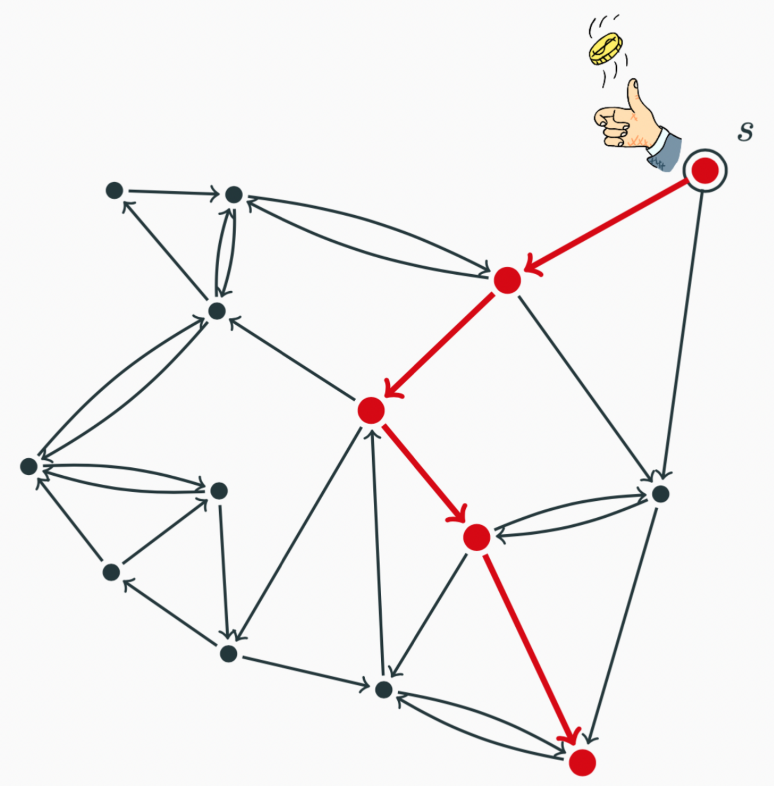
Near-Optimal Two-Pass Streaming Algorithm for Sampling Random Walks over Directed Graphs

Lijie Chen, Gillat Kol, Dmitry Paramonov, Raghuvansh R. Saxena, Zhao Song, Huacheng Yu

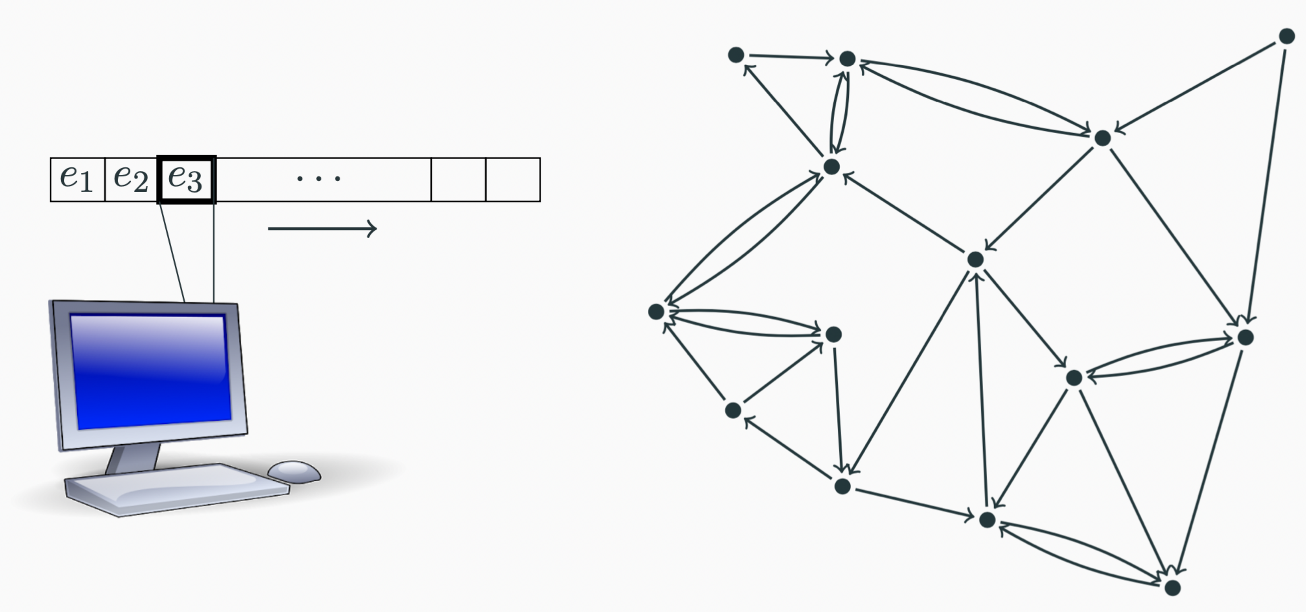
Random Walks on Directed Graphs

n vertices, L steps



Graph Streaming Algorithms [HRR98, FKM+09]

Low memory algorithm can only make few **passes** over the graph.



Prior Work

Known algorithms either use a lot of memory, or a lot of passes.

Reference	# Passes	# Memory	Remarks
Easy	L	$\mathcal{O}(\log n)$	
[SGP11]	$\tilde{\mathcal{O}}(\sqrt{L})$	$\tilde{\mathcal{O}}(n)$	
Easy	1	$\tilde{\Theta}(nL)$	Tight [Jin19]
Our work	**2**	$\boldsymbol{\tilde{\Theta}(n\sqrt{L})}$	

Table 1: L-step random walk on n-vertex graphs.

Our Result

Theorem

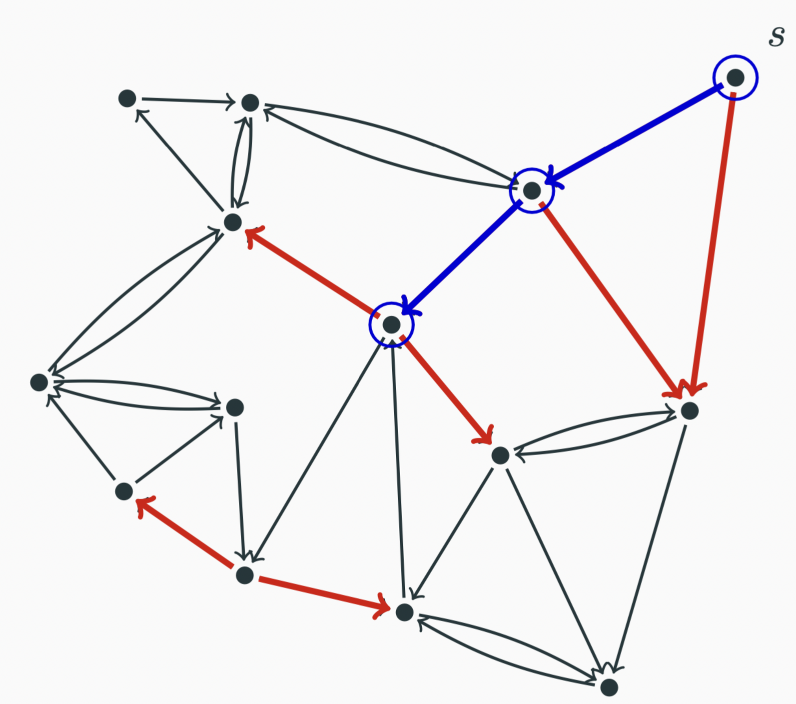
Two pass algorithms need $\tilde{\Theta}(n \cdot \sqrt{L})$ space to find a random walk.

Why it is cool?

- Works in the turnstile model.
- Is starting vertex oblivious (svo).
- Tight for any svo algorithm with any number of passes.
- $\tilde{\Omega}(n \cdot L^{1/p})$-lower bound for p-pass non-svo algorithms.

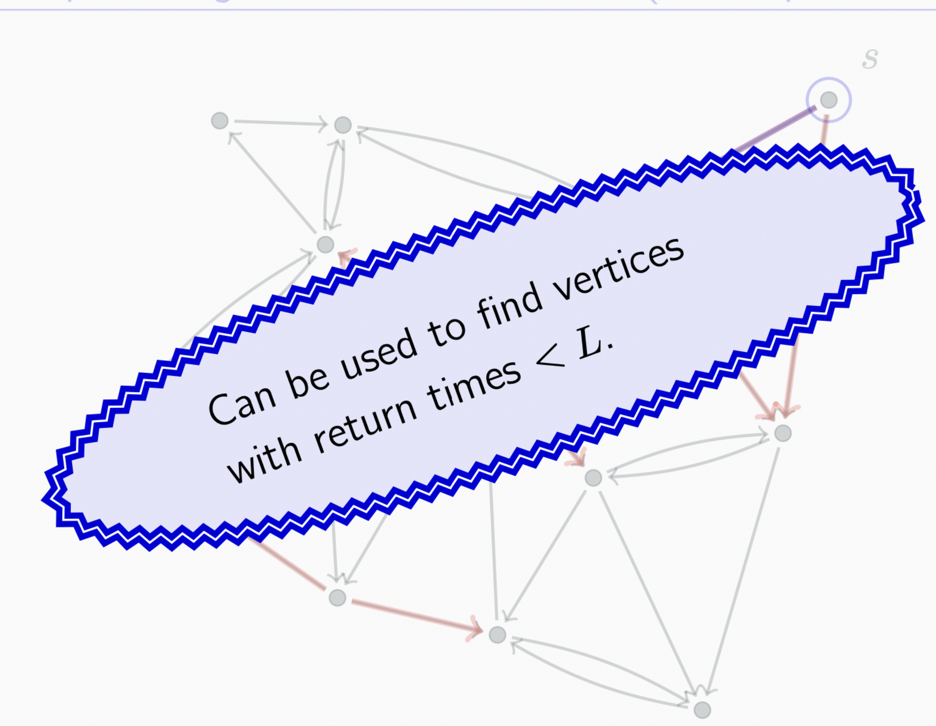
The 1-pass $\mathcal{O}(nL)$-space algorithm

Sample L edges from each vertex! (with replacement)



The 1-pass $\mathcal{O}(nL)$-space algorithm

Sample L edges from each vertex! (with replacement)



Our 2-pass $\mathcal{O}(n \cdot \sqrt{L})$-space algorithm

One pass algorithm tells which vertices have return time $< k$ in $\tilde{\mathcal{O}}(nk)$ space.

First pass

Call a vertex heavy if it has return time $< k$, and light otherwise.

Second pass

Store all edges from heavy vertices, and L/k edges from each light vertex.

How many edges to remember?

Light vertices: Only need to store L/k edges per vertex, at most $\frac{nL}{k}$ overall.

Lemma (Heavy vertices)

The total edges coming out of heavy vertices is $\mathcal{O}(nk)$.

Proof.

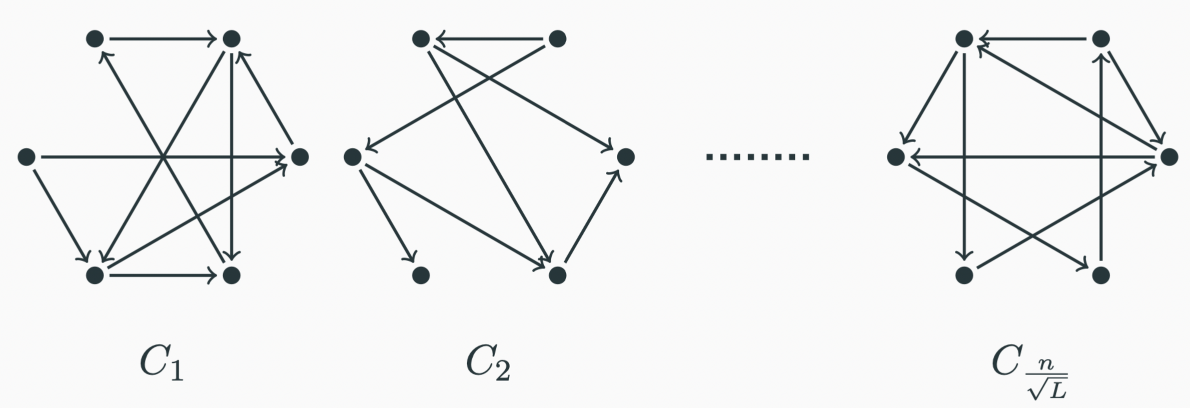
For all v, there are at most $\mathcal{O}(k)$ heavy u that have an edge to v.

- For heavy u, random walk from u will return in $< k$ steps.
- For heavy u with an edge to v, walk from v visits u in $< k$ steps.
- At most k vertices visited in k steps.

□

The Hard Instance

$\frac{n}{\sqrt{L}}$ disjoint random graphs on $\sqrt{L}$-vertices



- L-length walk equivalent to an entire component.
- svo-algorithms do not know which component.