NYU

Breaking 3-Factor Approximation for Correlation Clustering in Polylogarithmic Rounds

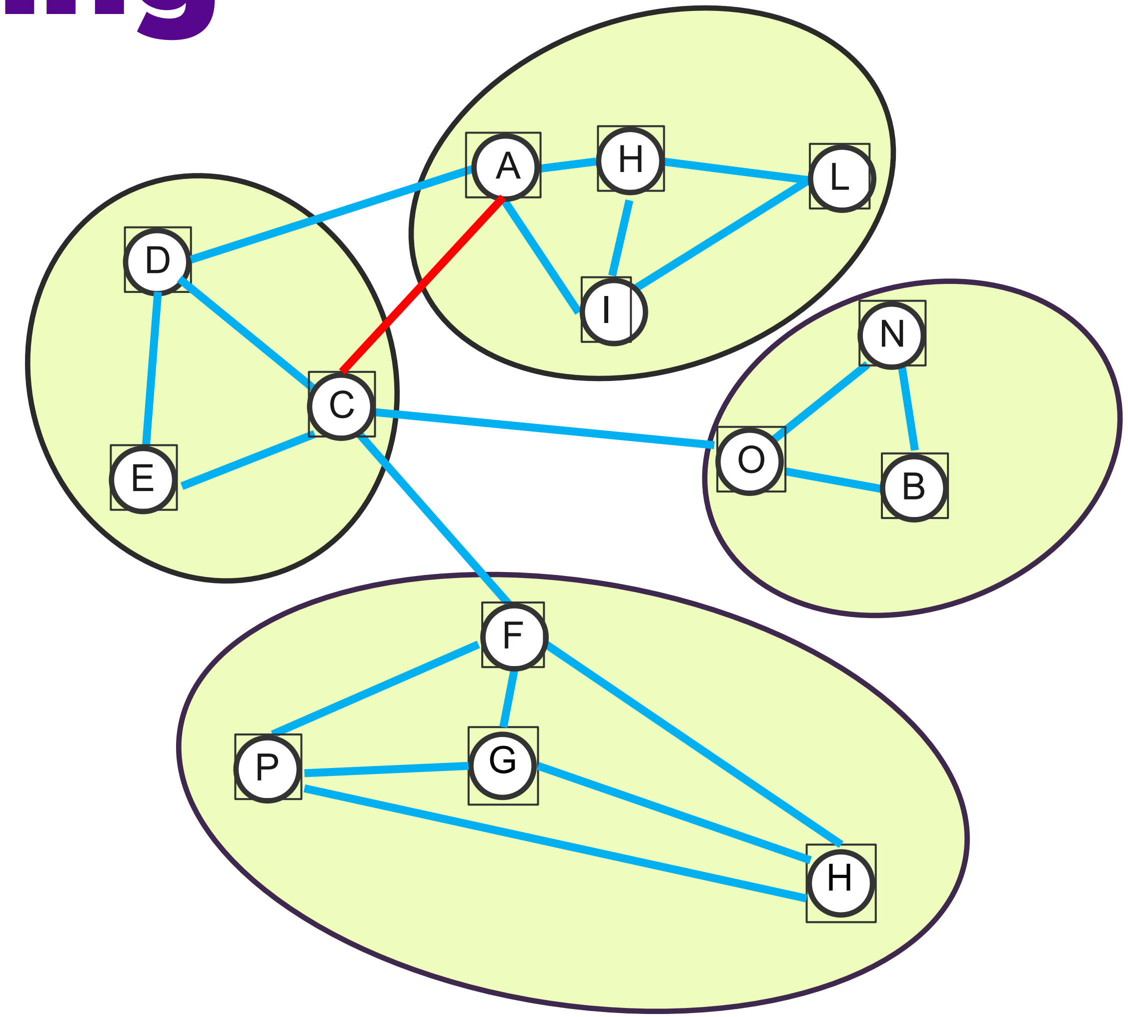
Nairen Cao (NYU Tandon)

Joint work with Shang-En Huang (National Taiwan University) and Hsin-hao Su (Boston College)

Correlation Clustering

- **Input**: Given a complete graph $G = (V, E)$, each edge is either $+$ or $-$. Usually, only $+$ edges are shown

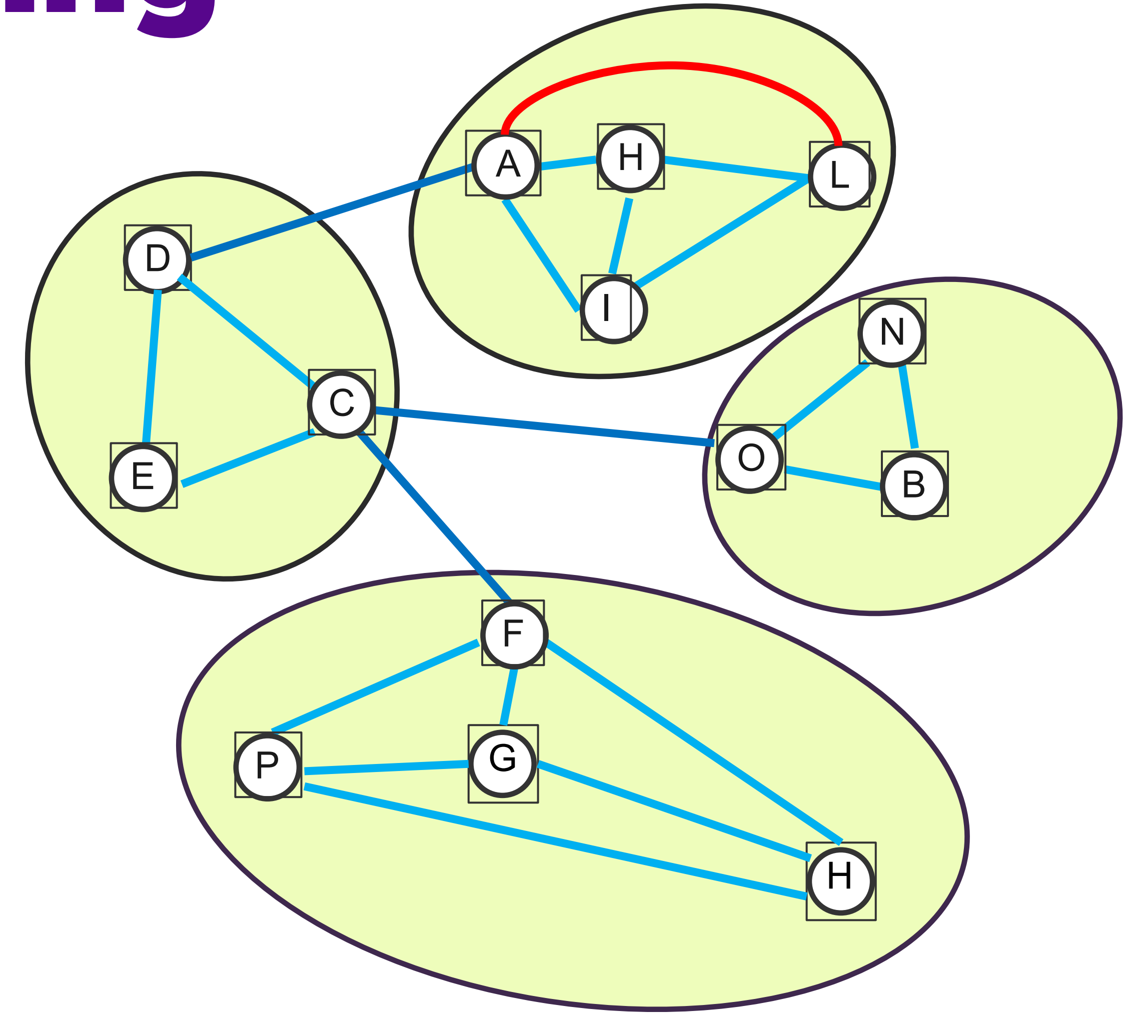
- **Output**: Partition V into $(V_1, , \ldots V_k)$ which minimizes total disagreements:
  - $+$ edges in different clusters
  - $-$ edges in the same cluster
  - k is not given



Correlation Clustering

- **Input**: Given a complete graph $G = (V, E)$, each edge is either + or −. Usually, only + edges are shown.

- **Output**: Partition V into $(V_1, , \ldots V_k)$ which minimizes total disagreements:
  - + edges in different clusters
  - − edges in the same cluster
  - k is not given

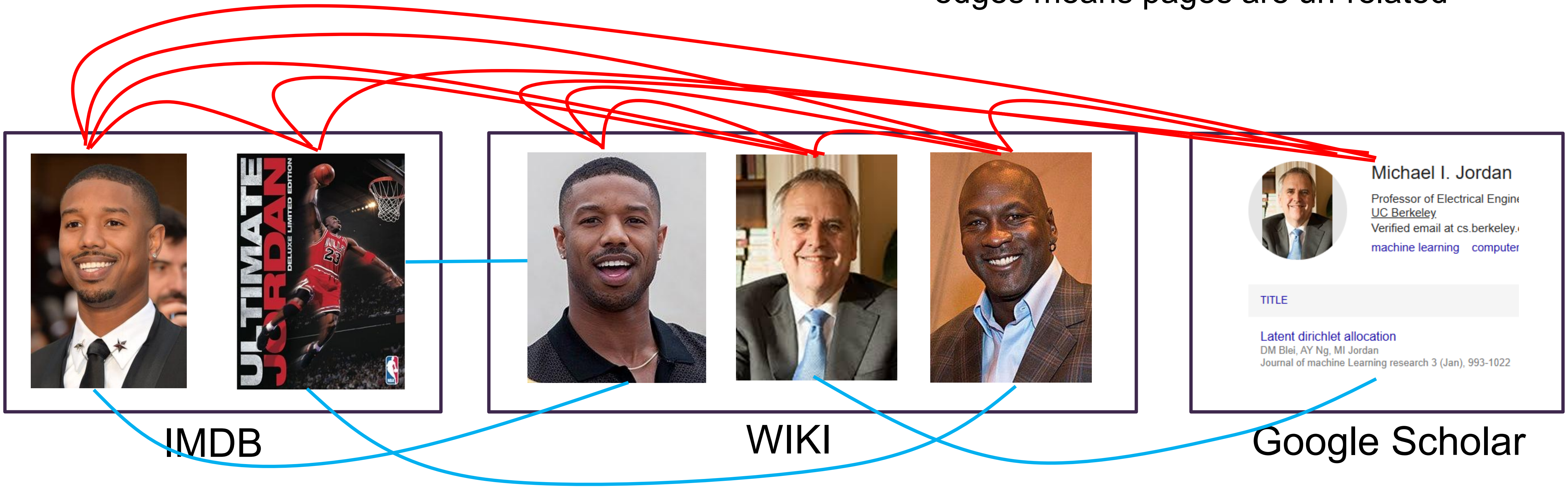


Disagreements:
- DA, CO, FC
- AL

Motivation

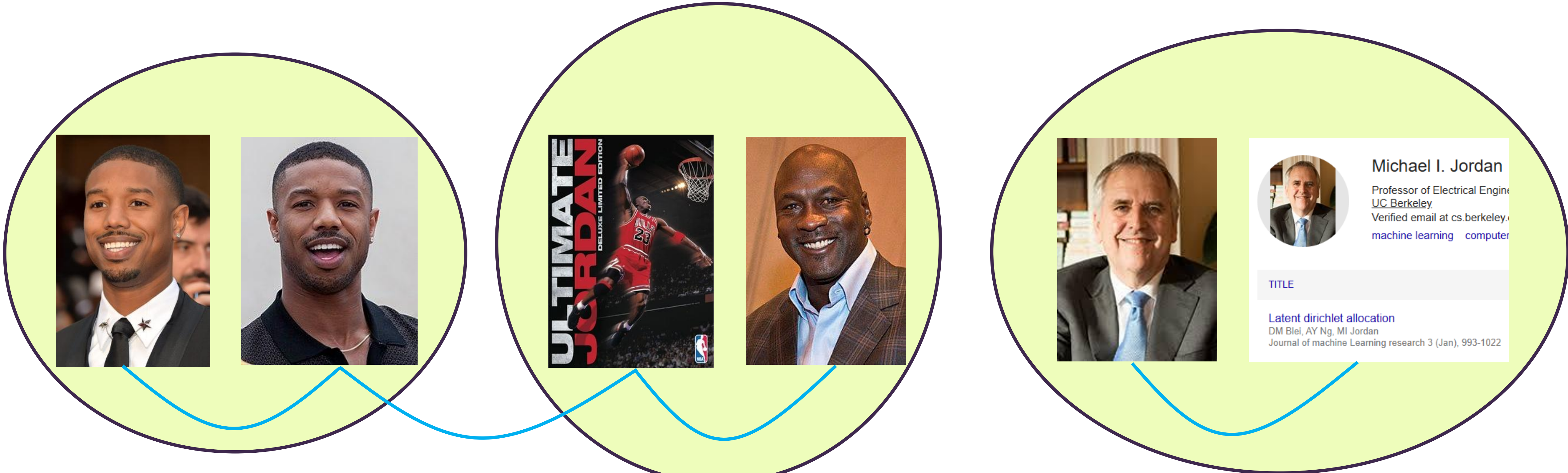
- One application: building an entity graph on the web
  - Many pages on “Michael Jordan”
  - Which pages refer to same person?

- + edges means pages are related
- – edges means pages are un-related



Motivation

- One application: building an entity graph on the web
  - Many pages on “Michael Jordan”
  - Which pages refer to same person?



Different versions

- General weighted graphs?
- Maximizing (# of agreement) instead of minimizing (# of disagreement)?

	COMPLETE	GENERAL
Max	PTAS[GG '05]	Optimally approximated using SDP method[Raghavendra '08]
Min	This talk	No Better than Multicut

History

- Formulated By Bansal, Blum and Chawla ’02

- NP-Hard and APX-hard, shown by Charikar, Guruswami and Wirth ’05
  - No algorithm can achieve 1.01 approximation ratio

- Can be solved within approximation ratio 1.437 in sequential setting, shown by Cao, Cohen-Addad, Lee, Li, Newman, Vogl ’24

History

Low-Rounds Algorithms for Big Graphs

References	Approx.	Rounds	Notes
[Assadi and Wang ’22]	≈ 100000	1	Sublinear MPC
[CLMNPT ’21]	701	$O(1)$	Sublinear MPC
[ACGMW ’21]	3	$O(\log\log n)$	Sublinear MPC
[Cambus, Choo, Miikonen, Uitto ’21]	3	$O(\log\Delta\log\log n)$	Sublinear MPC
[Blelloch, Fineman, Shun ’12]	3	$O(\log^2 n)$	Sublinear MPC
[Chierichetti, Dalvi, Kumar ’14]	$3+\epsilon$	$O(\log n\log\Delta/\epsilon)$	Sublinear MPC
[Fischer and Noever ’20]	3	$O(\log n)$	Sublinear MPC
[Behnezhad, Charikar, Ma, Tan ’22]	$3+\epsilon$	$O(1/\epsilon)$	Sublinear MPC
[Behnezhad, Charikar, Ma, Tan ’23]	5	1	Streaming
[CKLPU ’23]	$3+\epsilon$	1	Linear MPC

- They all have ≥3-approximation ratio.

NYU

Bottleneck in Parallel Algorithms

- They all have ≥3-approximation ratio
  - The 3 or $(3+\epsilon)$ ones are based on parallelizing the PIVOT algorithm of [Ailon, Charikar, Newman '08]

- Does there exist any small-round parallel algorithm that can achieve an approximation factor that is better than 3?

Main Result

- A poly$\left(\frac{1}{\epsilon}, \log n\right)$ rounds $(2.4+\epsilon)$-approximation PRAM algorithm using $\tilde{O}(m^{1.5})$ work, where m is #positive edges.
  - Can also be implemented in MPC with $\tilde{O}(m^{1.5})$ memory
  - poly$\left(\frac{1}{\epsilon}, \log n\right)$ is $\frac{(\log n)^6}{\epsilon^7}$

NYU

History

- Prior work has ≥3-approximation ratio.

Low-Rounds Algorithms for Big Graphs

References	Approx.	Rounds	Notes
[Assadi and Wang '22]	≈ 100000	1	Sublinear MPC
[CLMNPT '21]	701	$O(1)$	Sublinear MPC
[ACGMW '21]	3	$O(\log\log n)$	Sublinear MPC
[Cambus, Choo, Miikonen, Uitto '21]	3	$O(\log\Delta\log\log n)$	Sublinear MPC
[Blelloch, Fineman, Shun '12]	3	$O(\log^2 n)$	Sublinear MPC
[Chierichetti, Dalvi, Kumar '14]	$3+\epsilon$	$O(\log n\log\Delta/\epsilon)$	Sublinear MPC
[Fischer and Noever '20]	3	$O(\log n)$	Sublinear MPC
[Behnezhad, Charikar, Ma, Tan '22]	$3+\epsilon$	$O(1/\epsilon)$	Sublinear MPC
[Behnezhad, Charikar, Ma, Tan '23]	5	1	Streaming
[CKLPU '23]	$3+\epsilon$	1	Linear MPC
[Cao, Huang, Su '24]	$2.4+\epsilon$	$\text{poly}\left(\frac{1}{\epsilon},\log n\right)$	SubLMPC, $\tilde{O}(\mathrm{m}^{1.5})$ total memory
[CLPTYZ '24]	1.847+ϵ	$2^{poly(\frac{1}{\epsilon})}$	subLMP, at least $\left(\frac{1}{\epsilon}\right)^{162}$

Outline

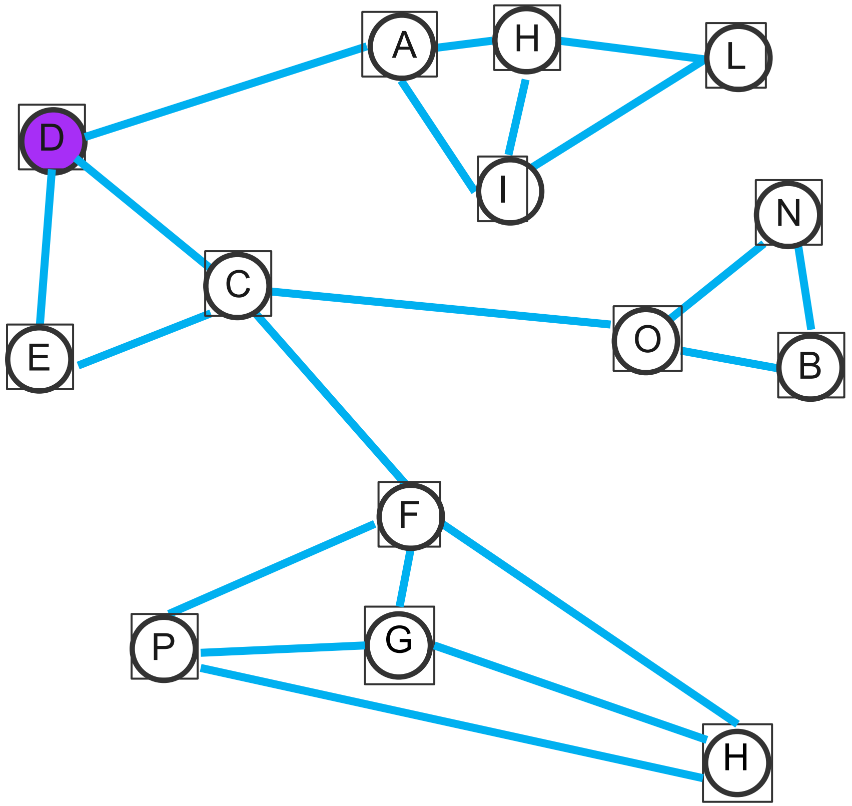
- Previous parallel algorithms get stuck at 3
- How to beat 3 in the sequential setting?
- Our idea

Why previous work get stuck at 3?

- All previous algorithms try to parallelize the PIVOT algorithm.
- PIVOT algorithm is proposed by Ailon, Charikar, Newman ’08

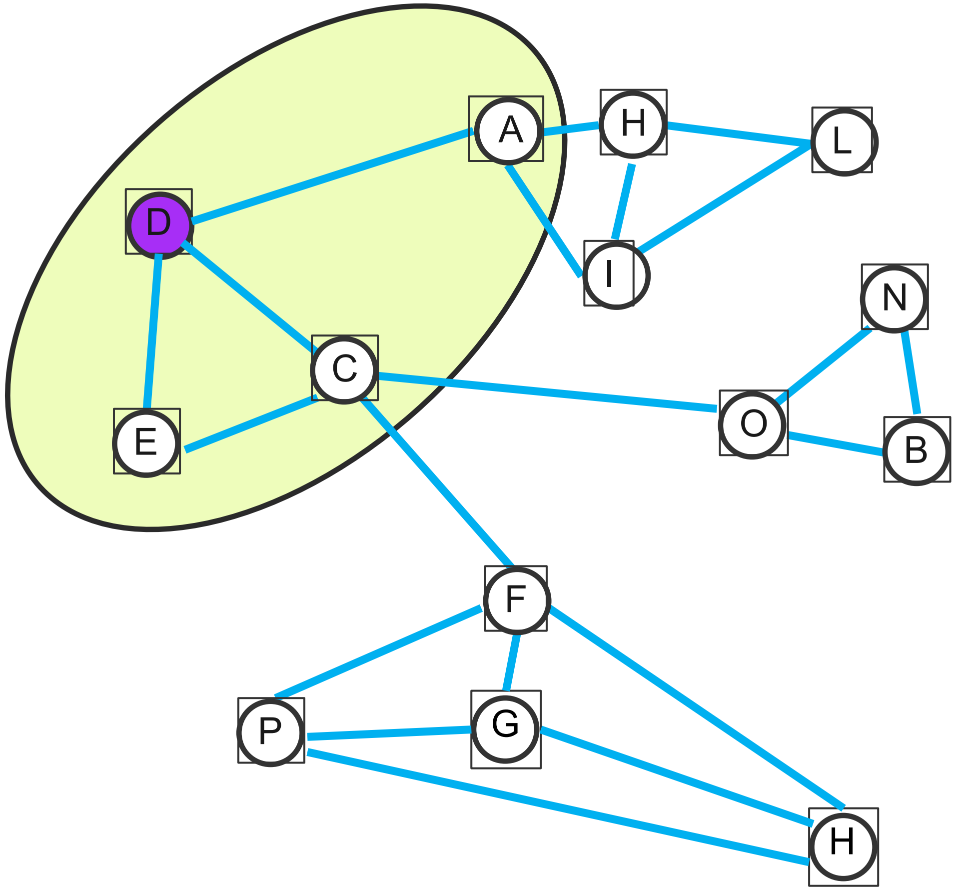
The PIVOT Algorithm [ACN '08]

1. Randomly select a node *u*
2. Put all positive neighbors of *u* in the same cluster as *u*
3. Recurse on the un-clustered nodes



The PIVOT Algorithm [ACN '08]

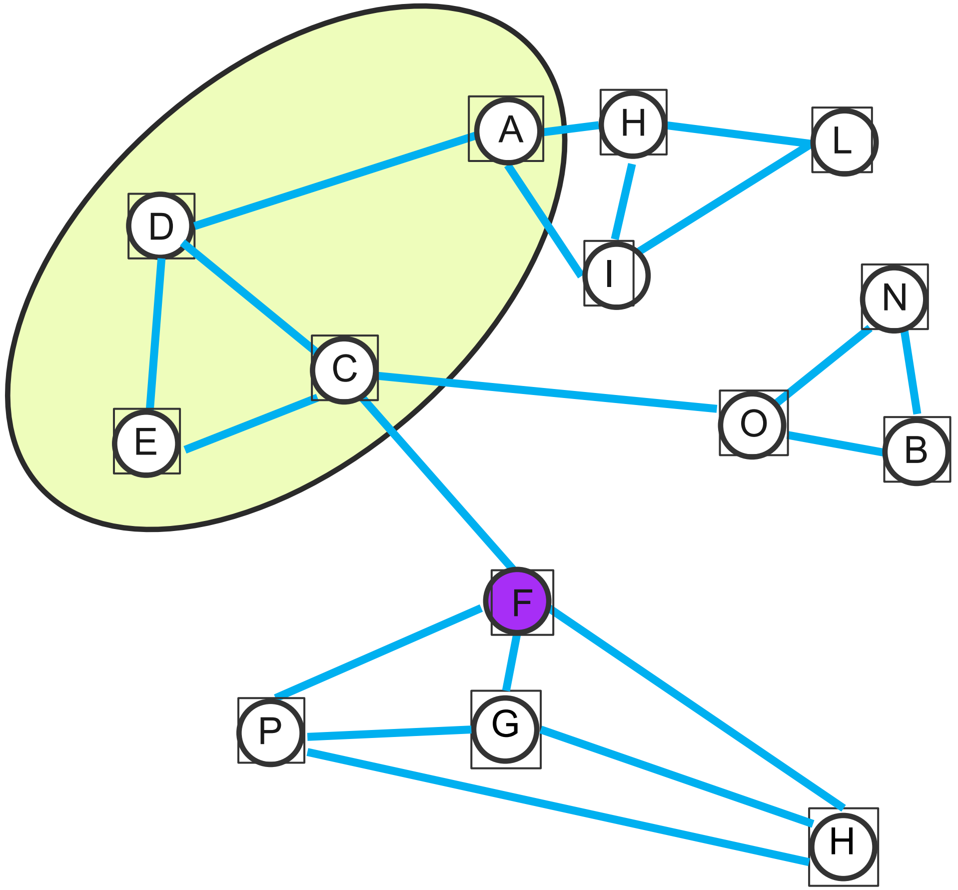
1. Randomly select a node *u*
2. Put all positive neighbors of *u* in the same cluster as *u*
3. Recurse on the un-clustered nodes



NYU

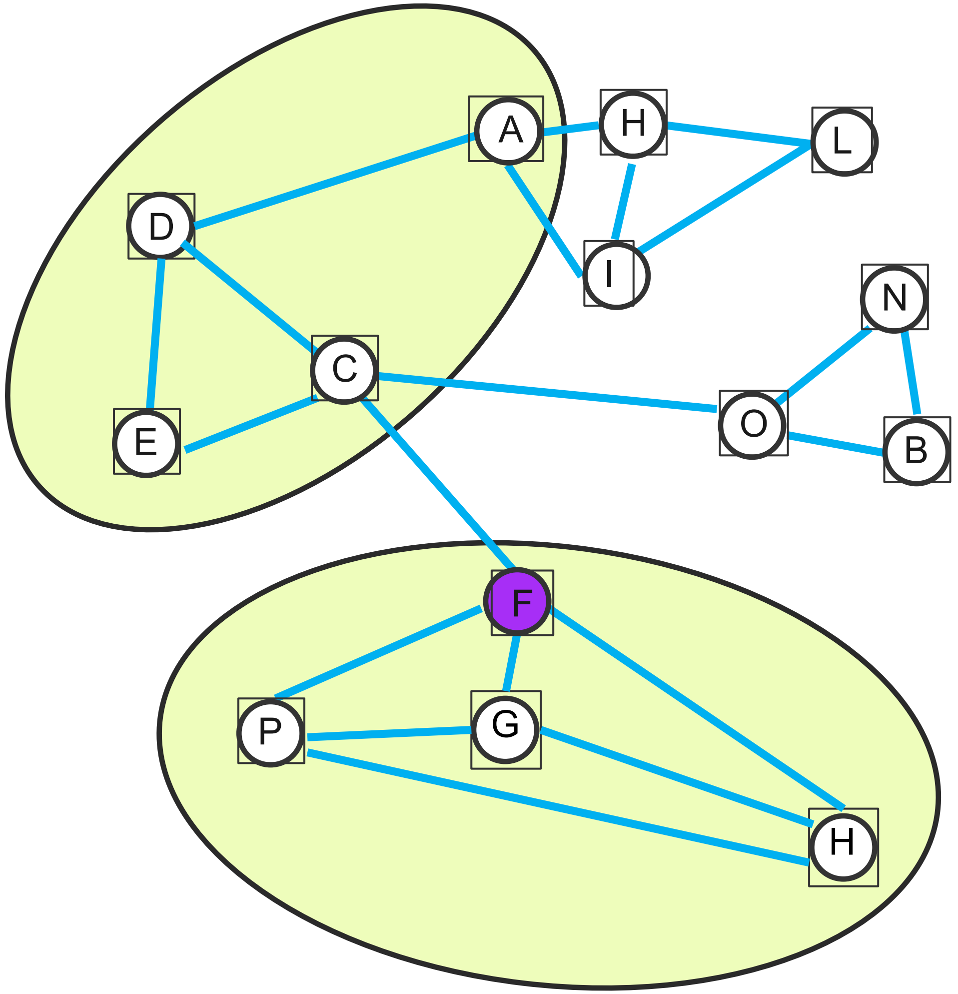
The PIVOT Algorithm [ACN '08]

1. Randomly select a node *u*
2. Put all positive neighbors of *u* in the same cluster as *u*
3. Recurse on the un-clustered nodes



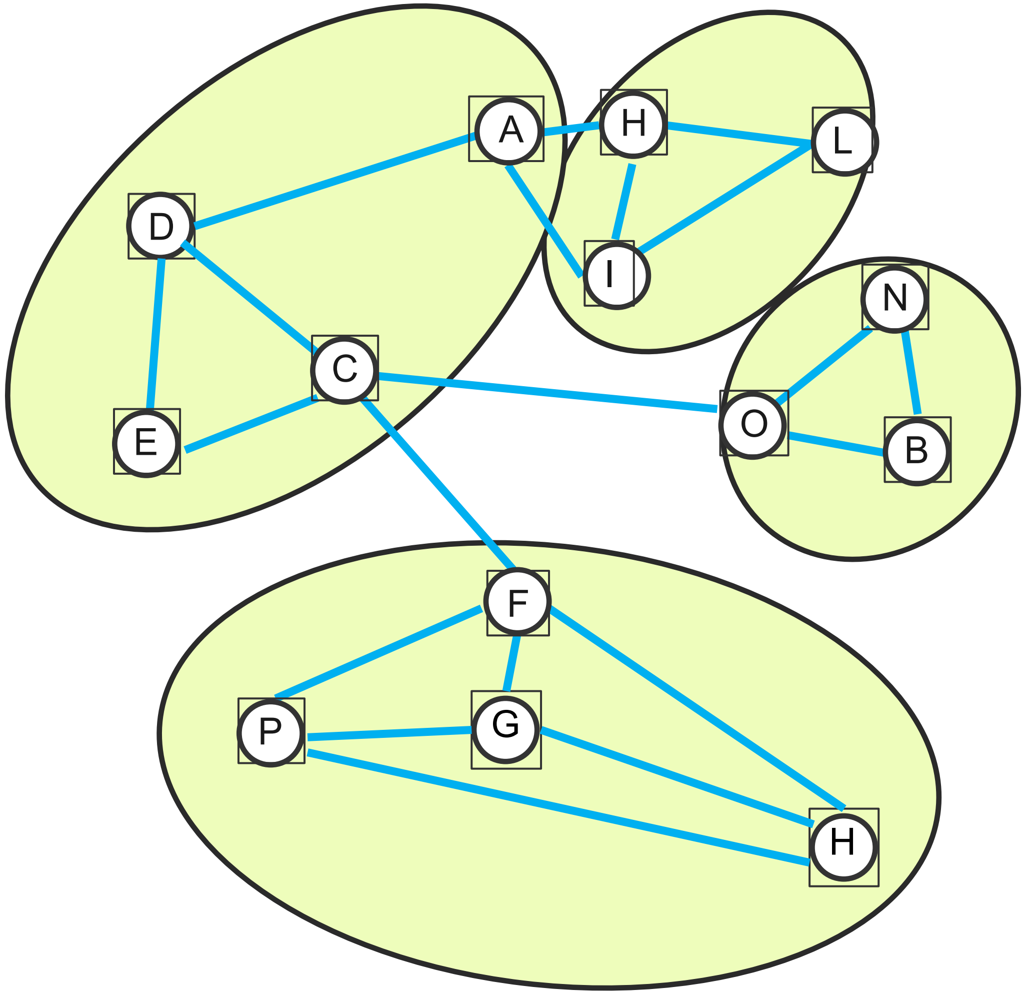
The PIVOT Algorithm [ACN '08]

1. Randomly select a node *u*
2. Put all positive neighbors of *u* in the same cluster as *u*
3. Recurse on the un-clustered nodes



The PIVOT Algorithm [ACN '08]

1. Randomly select a node *u*
2. Put all positive neighbors of *u* in the same cluster as *u*
3. Recurse on the un-clustered nodes

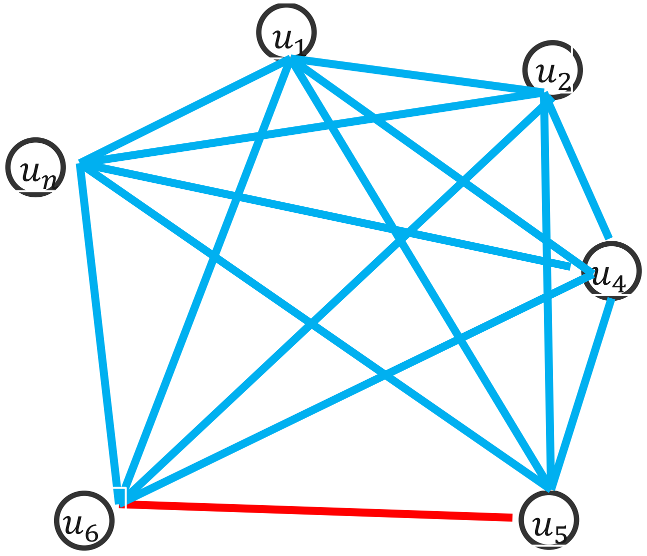


Why previous work get stuck at 3?

- All previous algorithms try to parallelize the PIVOT algorithm.
- Unfortunately, pivot algorithm is tight for 3-approximate ratio.

Why previous work get stuck at 3?

- All previous algorithms try to parallelize the PIVOT algorithm.
- Unfortunately, pivot algorithm is tight for 3-approximate ratio.



- OPT Cost: 1
    - All nodes in one cluster
- PIVOT algorithm:
    - With prob. $1 - 2/n$, cost 1
    - With prob. $2/n$, cost $n - 2$
    - Total cost: 3

How to get below 3 sequentially?

- Sequential algorithms:
  - Solve the following linear program first.

Minimize $\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$

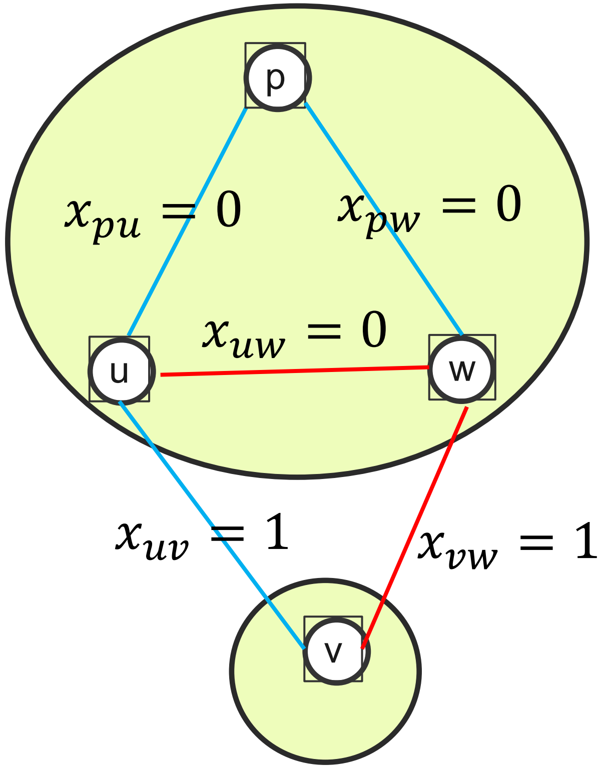
Subject to $x_{uw} + x_{wv} \geq x_{uv} \quad \forall u, w, v \in V$

$x_{uu} = 0 \quad \forall u \in V$

$x_{uv} \in [0,1] \quad \forall u, v \in V$

How to get below 3 sequentially?

- Variables: $\{x_{uv} : u, v \in V^2\}$
- Ideally, $x_{uv} = 0$ if u, v are in the same cluster and 1 otherwise.
- (u, v) becomes wrong when
  - $x_{uv} = 1$ but (u, v) is $+$, pay x_{uv} cost
  - $x_{uv} = 0$ but (u, v) is $-$, pay $1 - x_{uv}$ cost
- Triangle inequality: For any $u, v, \mathrm{w} \in V^3$, if u, v is sperate, then either v, w or u, w should be sperate.



Minimize	$\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$	
Subject to	$x_{uw} + x_{wv} \geq x_{uv}$	$\forall u, w, v \in V$
	$x_{uv} \in \{0,1\}$	$\forall u, v \in V$

How to get below 3 sequentially?

+ Modification [ACN '08, CMSY '15] of the PIVOT algorithm as follows,

1. Solve the LP. Let $\{x_{uv}\}$ be the solution.
2. Randomly select a node *u*
3. For edges *uv* that is adjacent to u
   - Put it in the same cluster as x with prob. $1 - x_{uv}$ if $uv \in E^+$
   - Put it in the same cluster as x with prob. $1 - x_{uv}$ if $uv \in E^-$
4. Recurse on the un-clustered nodes

Difficulty

- Sequential algorithms:
  - Almost all algorithms better than 3 start with a (nearly)-optimal solution of a linear programming relaxation

Minimize $\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$

Subject to $x_{uw} + x_{wv} \geq x_{uv} \quad \forall u, w, v \in V$

$x_{uu} = 0 \quad \forall u \in V$

$x_{uv} \in [0,1] \quad \forall u, v \in V$

- Problem: How to solve LP in poly-log rounds in parallel is unknown

NYU

How to get below 3 in parallel?

- Sequential algorithms:

$$\begin{aligned} &\text{Minimize} \quad \textstyle\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv}) \\ &\text{Subject to} \quad x_{uw} + x_{wv} \geq x_{uv} \quad \forall u, w, v \in V \\ &\qquad\qquad\quad x_{uu} = 0 \quad \forall u \in V \\ &\qquad\qquad\quad x_{uv} \in [0,1] \quad \forall u, v \in V \end{aligned}$$

- Problem: how to solve LP in parallel is unknown,
- Our contribution: How to relax constraints so that the LP is solvable in parallel and still save a good approximation ratio

NYU

Starting Point

- The following two LPs have the same optimal solution [CGW '05]

Triangle View

Minimize

$$\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$$

Subject to

$$x_{uw} + x_{wv} \geq x_{uv} \quad \forall u, w, v \in V$$

$$x_{uu} = 0 \quad \forall u \in V$$

$$x_{uv} \in [0,1] \quad \forall u, v \in V$$

Cut View

Minimize

$$\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$$

Subject to

$$\sum_{e\in P} x_e \geq x_{uv} \quad \forall uv \in E^-, P \in P_{uv}$$

$$x_{uv} \in [0,1] \quad \forall u, v \in V$$

P_{uv}: all simple paths of positive edges between u and v

Starting Point

- Consider the dual problem

PRIMAL

Minimize Cut View

$$\sum_{(u,v)\in E^+} x_{uv} + \sum_{(u,v)\in E^-} (1 - x_{uv})$$

Subject to

$$\sum_{e\in P} x_e \geq x_{uv} \quad \forall uv \in E^-, P \in P_{uv}$$

$$x_{uv} \in [0,1] \qquad \forall u, v \in V$$

DUAL

Maximize Flow View

$$\sum_{P\in \cup_{uv\in E^-} P_{uv}} f_P$$

Subject to

$$\sum_{P\ni e} f_p \leq 1 \qquad \forall e \in E^+$$

$$\sum_{P\in P_{uv}} f_p \leq 1 \qquad \forall uv \in E^-$$

Starting Point

- To solve the flow LP, compute shortest path.
- Main problem: How to compute "long" shortest path in parallel?

Maximize **Flow View**

$$\sum_{P \in \bigcup_{uv \in E^-} P_{uv}} f_P$$

Subject to

$$\sum_{P \ni e} f_p \leq 1 \qquad \forall e \in E^+$$

$$\sum_{P \in P_{uv}} f_p \leq 1 \qquad \forall uv \in E^-$$

NYU

Key Idea: Truncate the LP

Maximize **Flow View**

$$\sum_{P \in \bigcup_{uv \in E^-} P_{uv}} f_P$$

Subject to

$$\sum_{P \ni e} f_p \leq 1 \qquad \forall e \in E^+$$
$$\sum_{P \in P_{uv}} f_p \leq 1 \qquad \forall uv \in E^-$$

Maximize **Length-2 Flow View**

$$\sum_{P \in \bigcup_{uv \in E^-} P_{uv}(2)} f_P$$

Subject to

$$\sum_{P \ni e} f_p \leq 1 \qquad \forall e \in E^+$$
$$\sum_{P \in P_{uv}} f_p \leq 1 \qquad \forall uv \in E^-$$

$P_{uv}(2)$: all length-2 paths of positive edges between u and v

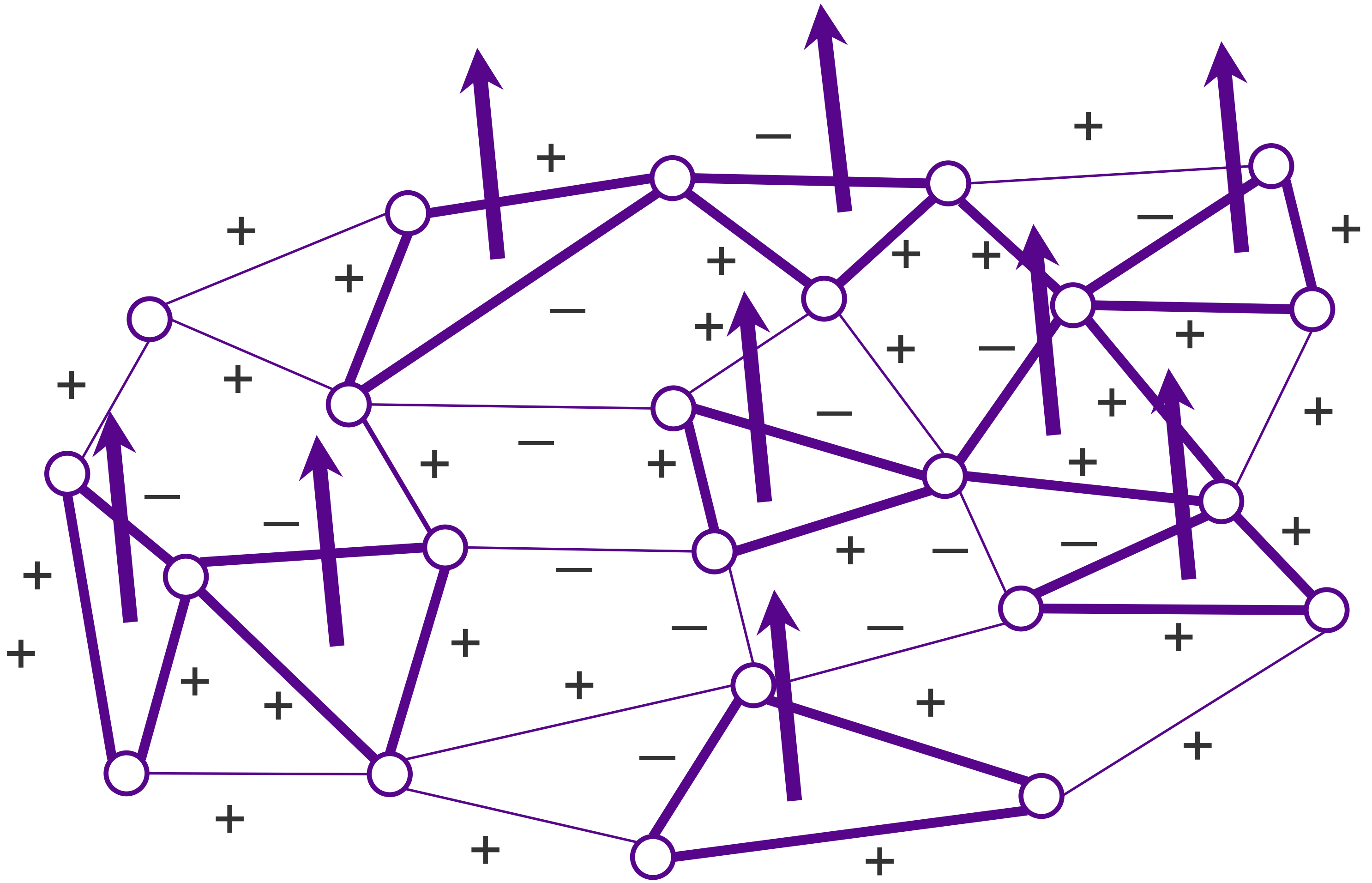
Solving the Truncated LP

- Each iteration involves finding a maximal set of shortest weights edge-disjoint paths (w.r.t. some weight function on the edges) in

$$\bigcup_{uv \in E^-} P_{uv}(2)$$

- Each path in $\bigcup_{uv \in E^-} P_{uv}(2)$ is a (+, +, -) triangle.

Solving the Truncated LP



The Truncated LP

- We give a polylog(n)-round, $\tilde{O}(m^{1.5})$-work algorithm for finding such a maximal set of such triangles, where m is #positive edges
  - Can also be implemented in MPC with $\tilde{O}(m^{1.5})$ total memory
  - This is the only bottleneck towards a $\tilde{O}(m)$-work algorithm

NYU

The Truncated LP

- We give a polylog(n)-round, $\tilde{O}(m^{1.5})$-work algorithm for finding such a maximal set of such triangles, where m is #positive edges
  - Can also be implemented in MPC with $\tilde{O}(m^{1.5})$ total memory
  - This is the only bottleneck towards a $\tilde{O}(m)$-work algorithm

- Why $\tilde{O}(m^{1.5})$?
  - The amount of (+,+,+) triangles is bounded by $O(m^{1.5})$
  - Explore (-,+,+) triangles in a way so that they can be charged to (+,+,+) triangles.

NYU

The Truncated LP: Hardness

- Bad news: Hard to beat $O(m^{1.5})$ under current framework, a reduction to the triangle detection problem
    - [Abboud-Williams '14, Williams-Williams '10] Commonly used assumption: No combinatorial algorithms can beat $O(m^{1.5})$

NYU

Summary

- A poly$\left(\frac{1}{\epsilon}, \log n\right)$ rounds $(2.4+\epsilon)$-approximation PRAM algorithm using $\tilde{O}(m^{1.5})$ work, where m is #positive edges.
  - Can also be implemented in MPC with $\tilde{O}(m^{1.5})$ memory