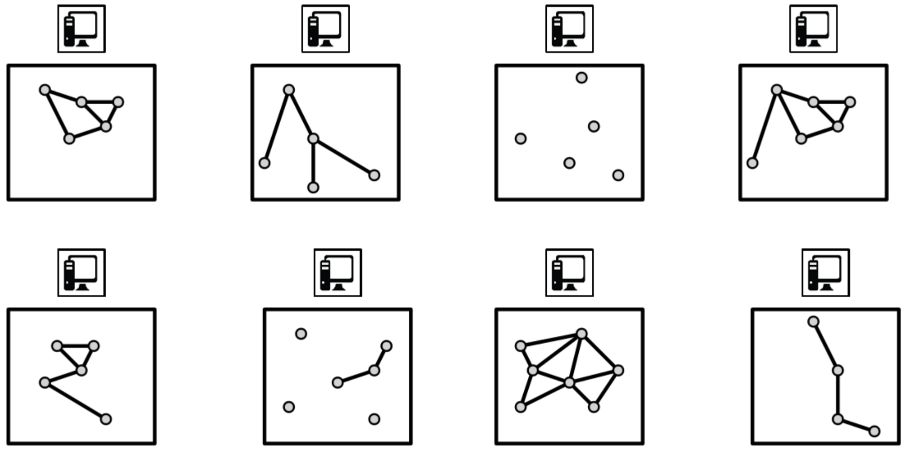
Symmetry Breaking in Massive Graphs

Jara Uitto, Aalto University

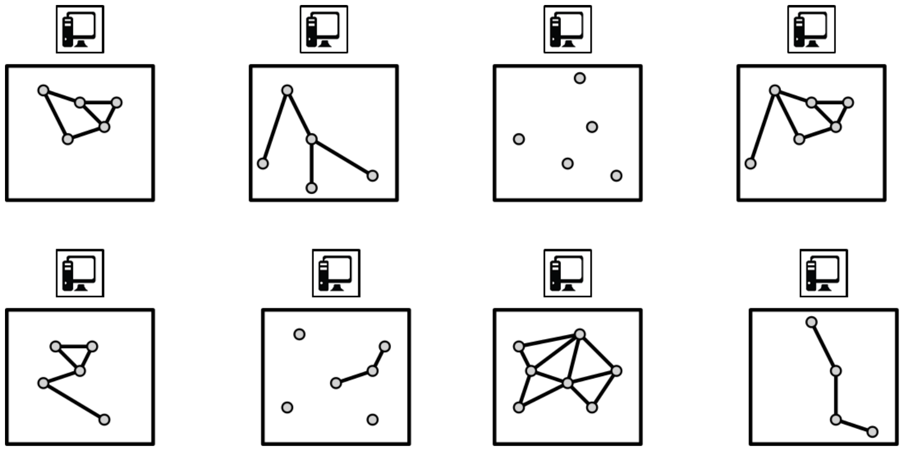
ADGA 2023



On Friday..

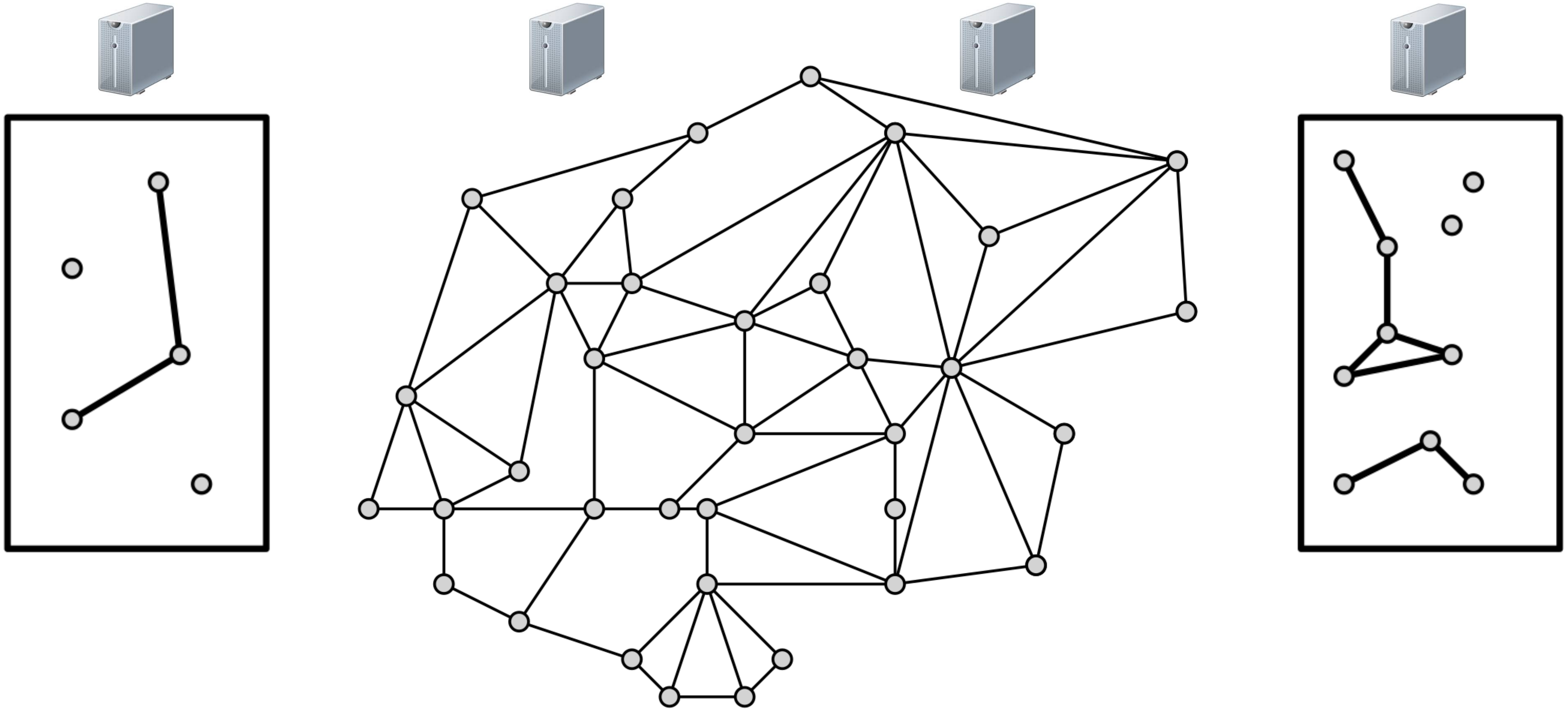
Computer repair team

The Plan



- MPC intro
  - LOCAL VS MPC
  - Locality Barrier
- Known Techniques
  - Sparsification and round compression
  - Derandomization
- New Techniques
  - Careful exponentiation
  - Total space

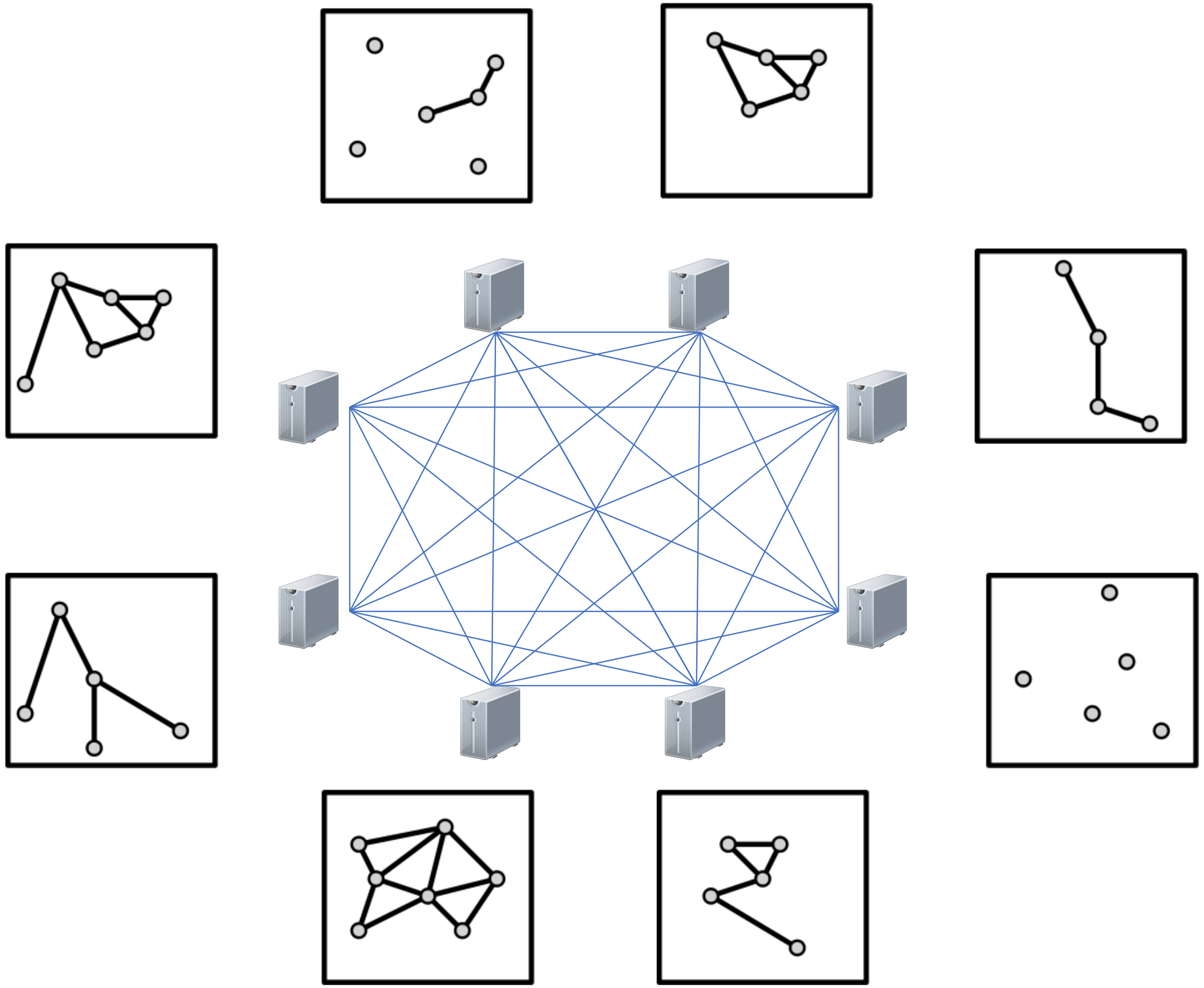
Massively Parallel Computing (MPC)



graph with n nodes and m edges

Massively Parallel Computing (MPC) Model

[Karloff, Suri, Vassilvitskii SODA'10]

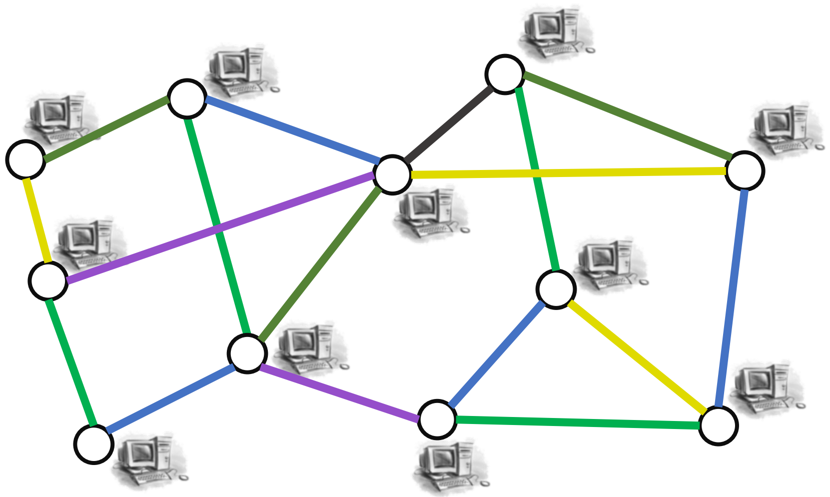


M machines
S memory per machine
Total space $M \cdot S$

Linear space:
$S = \tilde{O}(n)$
A sketch fits onto a single machine

Low-space:
$S = O(n^{\delta}), 0 \leq \delta < 1$
No machine ever sees all the nodes!

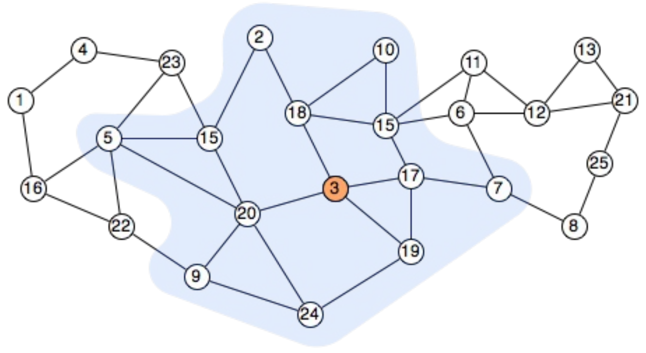
MPC vs Message Passing



MPC and Message Passing
Everyone knows their neighbors in the beginning.
Assume $\Delta < S$.

MPC can simulate T-rounds of message passing as long as $\mathbf{N}^{T}(v) \leq S$

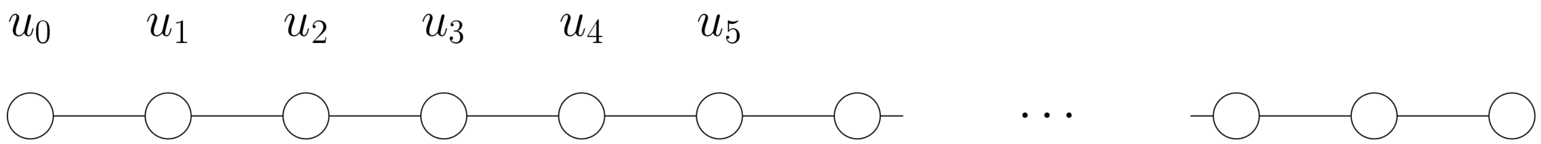
MPC vs Message Passing



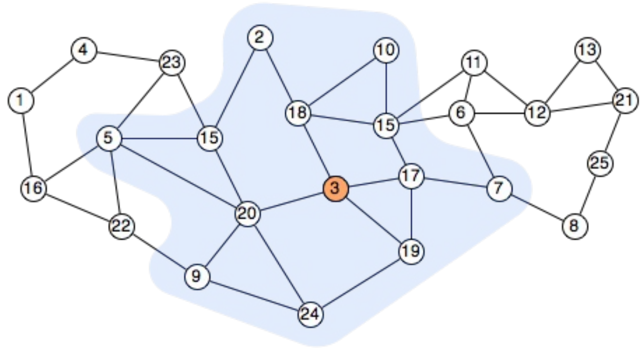
Design pattern: **Graph Exponentation**

Collect the T-hop neighborhoods in $O(\log T)$ rounds.

Simulate the LOCAL algorithm.



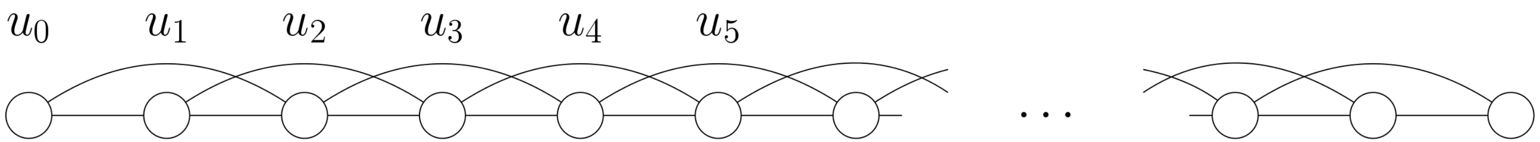
MPC vs Message Passing



Design pattern: Graph Exponentation

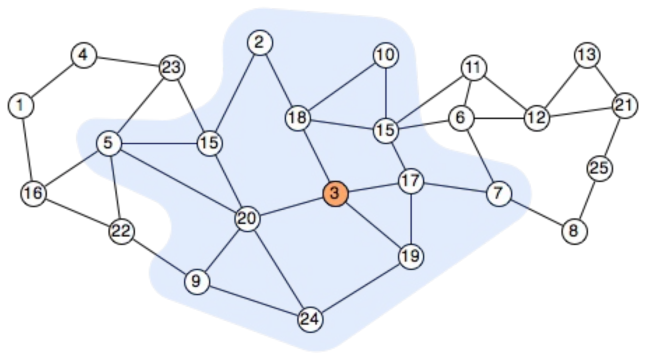
Collect the T-hop neighborhoods in $O(\log T)$ rounds.

Simulate the LOCAL algorithm.



Communicate in G^2

MPC vs Message Passing



Design pattern: **Graph Exponentation**

Collect the T-hop neighborhoods in $\boldsymbol{O}(\log T)$ rounds.

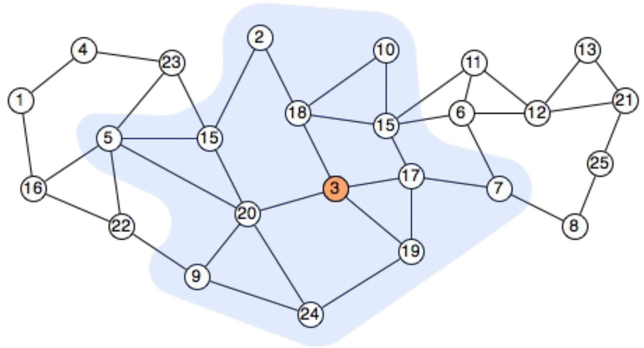
Simulate the LOCAL algorithm.

$(\Delta + 1)$-coloring

LOCAL: poly $\log\log n$ rounds [GG'23]

MPC: $O(\log\log\log n)$ rounds [CDP'21]

MPC vs Message Passing



Design pattern: **Graph Exponentation**

Collect the T-hop neighborhoods in $\boldsymbol{O}(\log T)$ rounds.

Simulate the LOCAL algorithm.

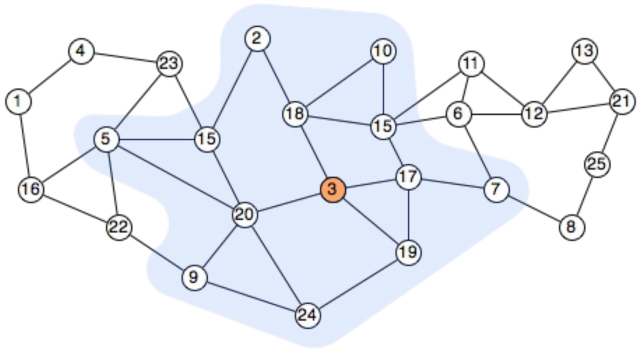
Still restricted by locality!

<u>$(\Delta + 1)$-coloring</u>

LOCAL: poly $\log\log n$ rounds [GG'23]

MPC: $O(\log\log\log n)$ rounds [CDP'21]

MPC vs Message Passing

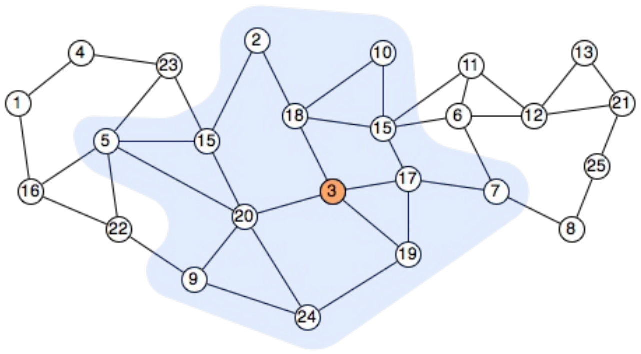


Global Power: Leader Election

Allows probability amplification
- Run $O(\log n)$ parallel repetitions
- Choose the best outcome

Example [KKSS’20, CPD’21]:
Independent sets of size $\Omega(n/\Delta)$

MPC vs Message Passing



Global Power: Leader Election

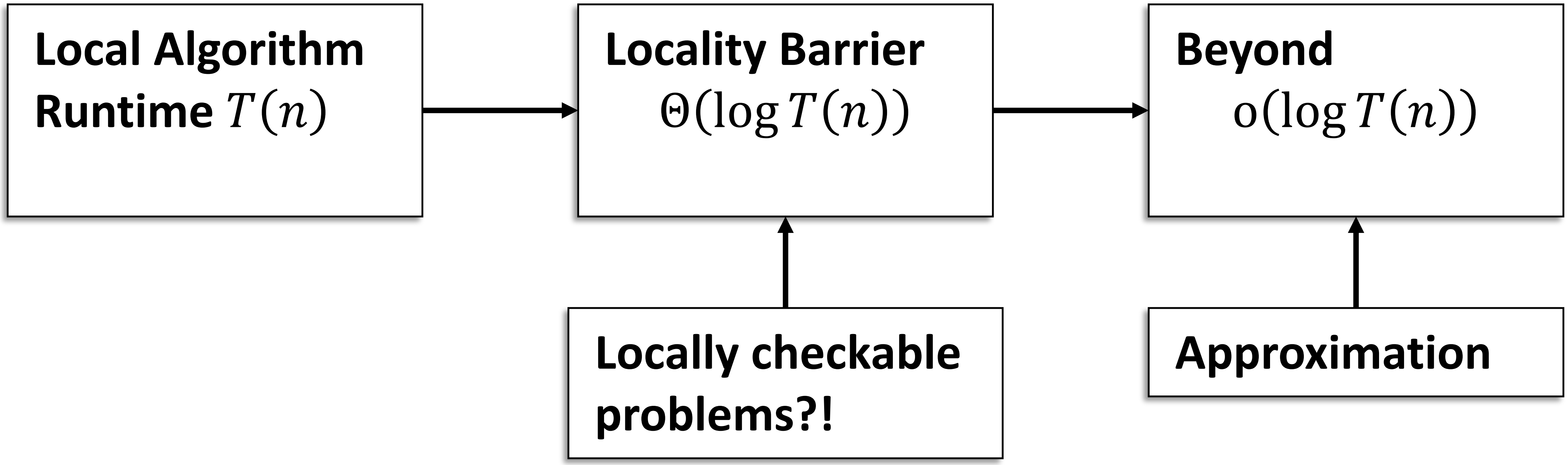
Allows probability amplification
- Run $O(\log n)$ parallel repetitions
- Choose the best outcome

Example [KKSS’20, CPD’21]:
Independent sets of size $\Omega(n/\Delta)$

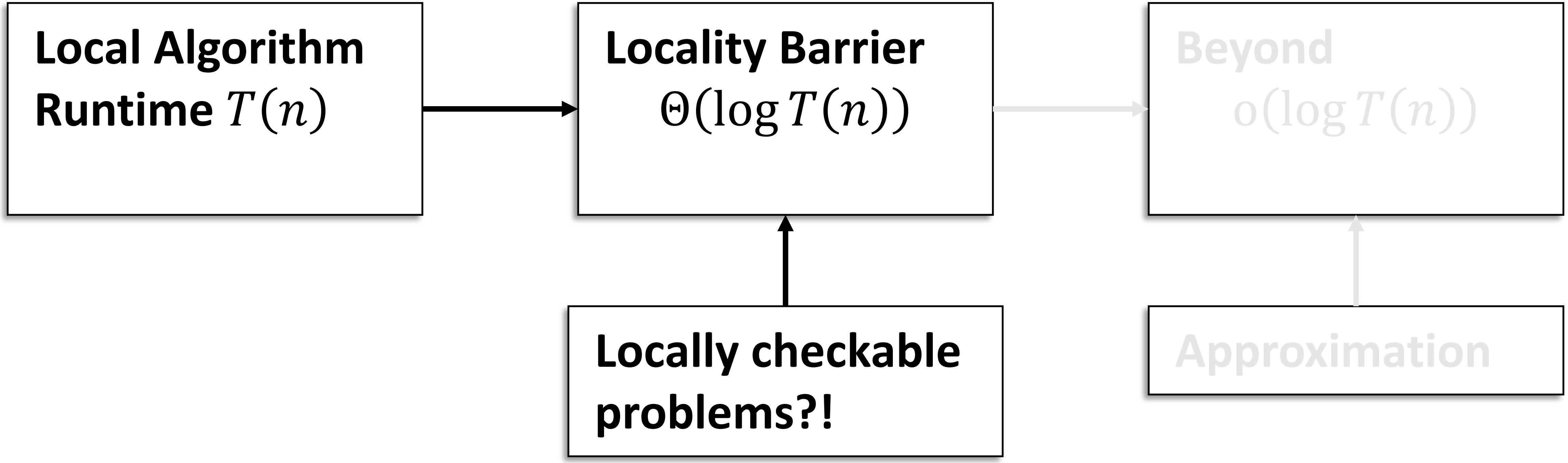
LOCAL: $\Omega(\log^* n)$
MPC: $O(1)$!

Locality Barrier

Exponentiation gets stuck here!



Locality Barrier



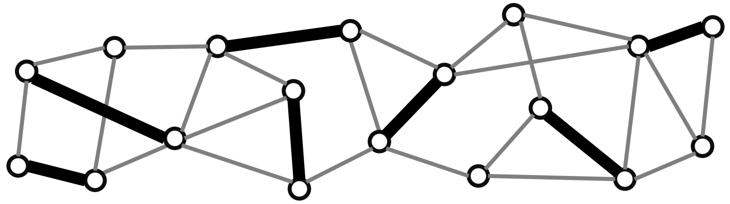
MIS and Maximal Matching

LOCAL [Gh'16]: $O(\log \Delta)$

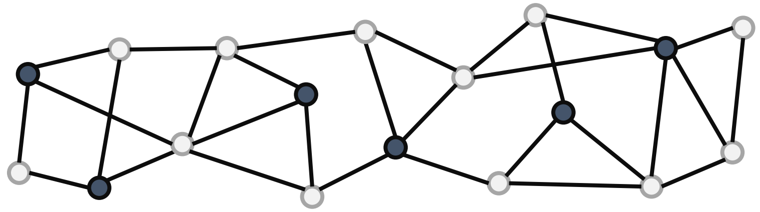
MPC [GU'19]: $\tilde{O}\left(\sqrt{\log \Delta}\right)$

??

Locality barrier: $\Theta(\log \log \Delta)$



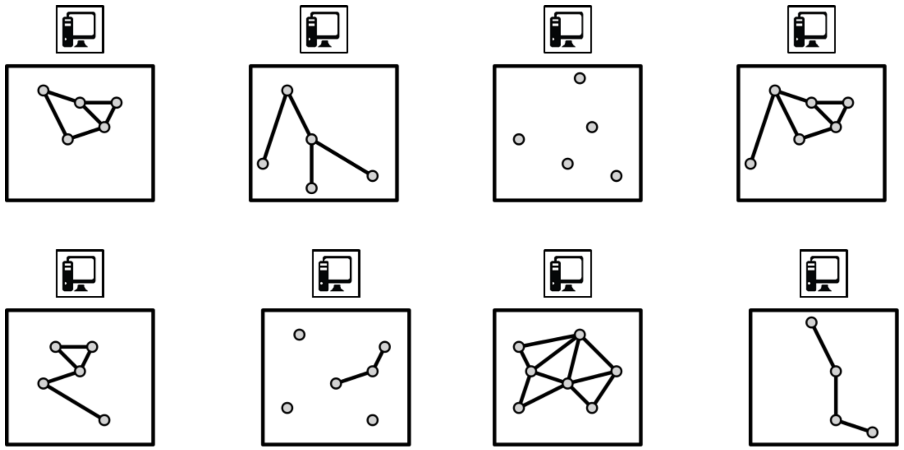
Maximal Matching



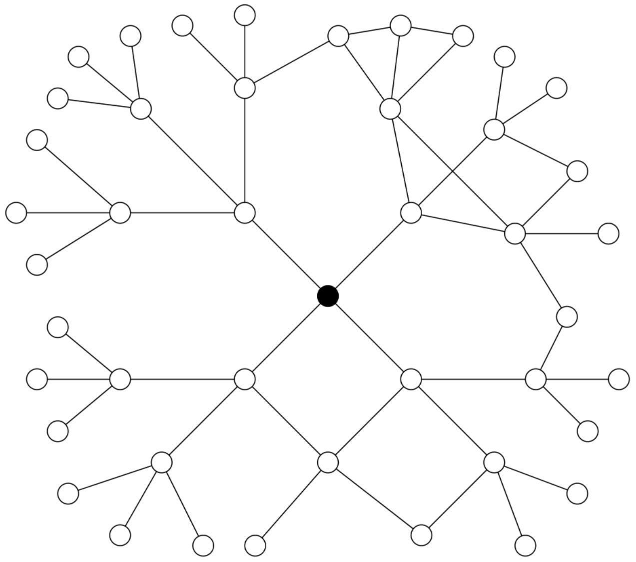
Maximal Independent Set

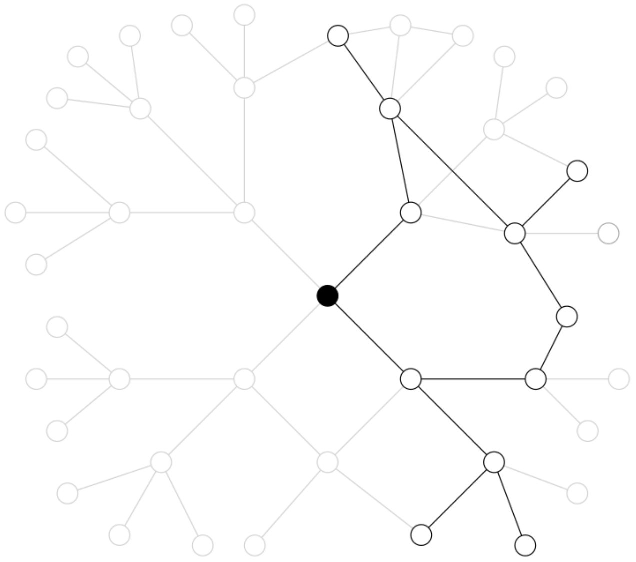
The Plan

- MPC intro
  - LOCAL VS MPC
  - Locality Barrier
- Known Techniques
  - Sparsification and round compression
  - Derandomization
- New Techniques
  - Careful exponentiation
  - Total space



Round Compression and Sparsification





2) Can we solve the problem with a small part of the input?

1) Can we solve the problem efficiently on a sparse graph?

Round Compression and Sparsification

MIS Sparsification Simplified (a lot):

1. Consider Ghaffari's algorithm that runs in $T = O(\log \Delta)$ rounds.
2. Simulate the algorithm on a sparse (low degree) subgraph for $\Omega\left(\sqrt{\log \Delta}\right)$ rounds.
3. Repeat $O\left(\sqrt{\log \Delta}\right)$ times.

$O(\log \Delta \cdot \log \log \Delta)$ in total.

Round Compression and Sparsification

MIS Sparsification Simplified (a lot):

1. Consider Ghaffari's algorithm that runs in $T = O(\log \Delta)$ rounds.
2. Simulate the algorithm on a sparse (low degree) subgraph for $\Omega\left(\sqrt{\log \Delta}\right)$ rounds.
3. Repeat $O\left(\sqrt{\log \Delta}\right)$ times.

$O(\log \Delta \cdot \log \log \Delta)$ in total.

Seems like a fundamental barrier

Black box application of exponentiation.

Shattering

MIS Sparsification Simplified (a lot):
After $O\left(\sqrt{\log \Delta}\right)$ iterations, the graph shatters into $O(\log n)$ sized components.

MPC: gather the components and simulate LOCAL.

Black box application of exponentiation.

Coloring

$(\Delta + 1)$-Coloring in MPC [CFGUZ'19, CDP'21]:
Split the graph into low-degree subgraphs with disjoint color palettes in $O(1)$ rounds.

Post-shattering:
Gather the components and simulate LOCAL.

Black box application of exponentiation.

LLL and Friends

Fischer & Ghaffari:
Find a partial solution to LLL in $O(\Delta^2)$ LOCAL rounds that shatters the graph.

Post-shattering:
Gather the components and simulate LOCAL.

Efficient in sparse graphs.

Black box application of exponentiation.

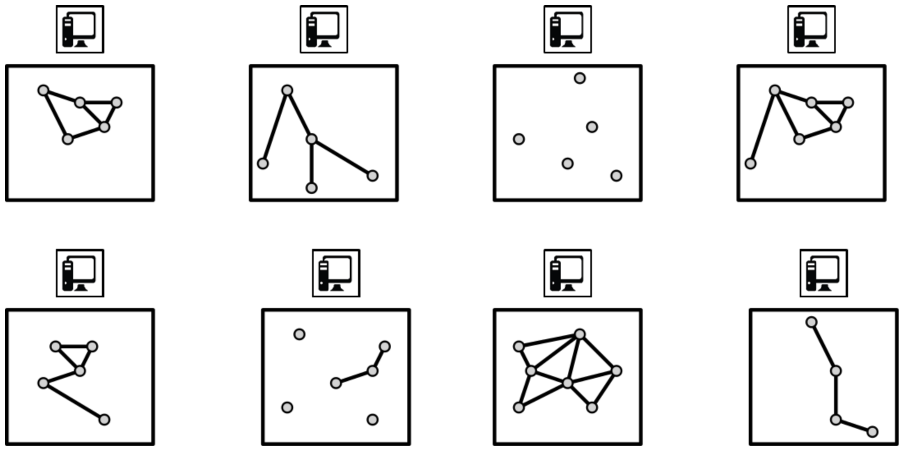
LLL and Friends

Take home:

A lot of naïve exponentiation

The Plan

- MPC intro
  - LOCAL VS MPC
  - Locality Barrier
- Known Techniques
  - Sparsification and round compression
  - Derandomization
- New Techniques
  - Careful exponentiation
  - Total space



Derandomization Tools

A Toolbox for Derandomization

Goal: Reduce the number of required random bits to $O(\log n)$ per node.

Agree globally and deterministically on a common random seed.

Derandomization Tools

A Toolbox for Derandomization

1. Conditional expectations
2. Limited independence
3. Pseudorandom generators

All (at least indirectly) employ naïve exponentiation

MIS: $O(\log \Delta + \log \log n)$

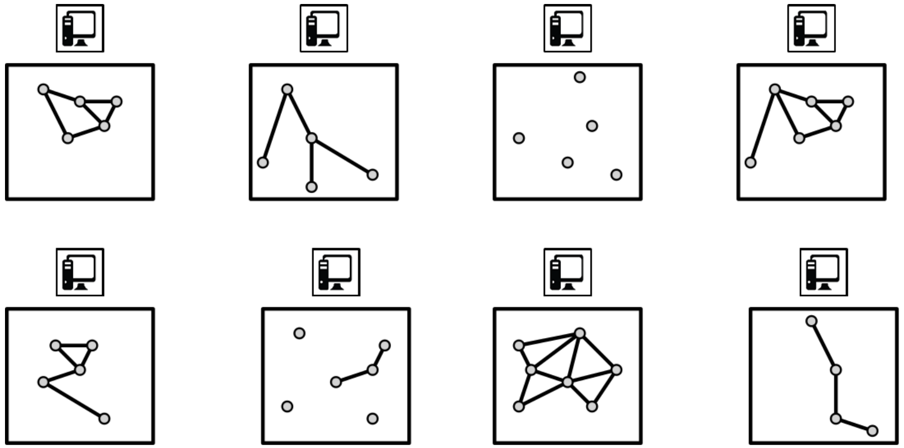
Coloring: $O(\log \log \log n)$

LLL: $O(\operatorname{poly} \Delta + \log \log \log n)$

Connected components (CC'21):
$O(\log \operatorname{diam} + \log \log n)$

The Plan

- MPC intro
  - LOCAL VS MPC
  - Locality Barrier
- Known Techniques
  - Sparsification and round compression
  - Derandomization
- New Techniques
  - Total space
  - Careful exponentiation



Naïve Exponentiation – So What?

- Increases total space demand
  - Connected components is an exception (linear total space). But it’s “slow”.

- Is it worth improving?
  - Linear is prettier (and optimal)
  - Wasted potential?!

Symmetry breaking
Pre-shattering: $n^{1+\Omega(1)}$
Post-shattering: $n^{1+o(1)}$

Naïve Exponentiation – So What?

Wasted potential?!

Example: MIS

Current SOTA seems fundamentally stuck at $\Theta\left(\sqrt{\log \Delta}\right)$.

Some room to improve by smarter exploration (exponentiation)?

Need new ideas!

How to change the game?

Naïve Exponentiation – So What?

Locally Checkable Problems:
Almost all approaches rely, to some extent, on naïve exponentiation.

Yields overhead in total space.

Limit total space to linear (tight)?

Naïve Exponentiation – So What?

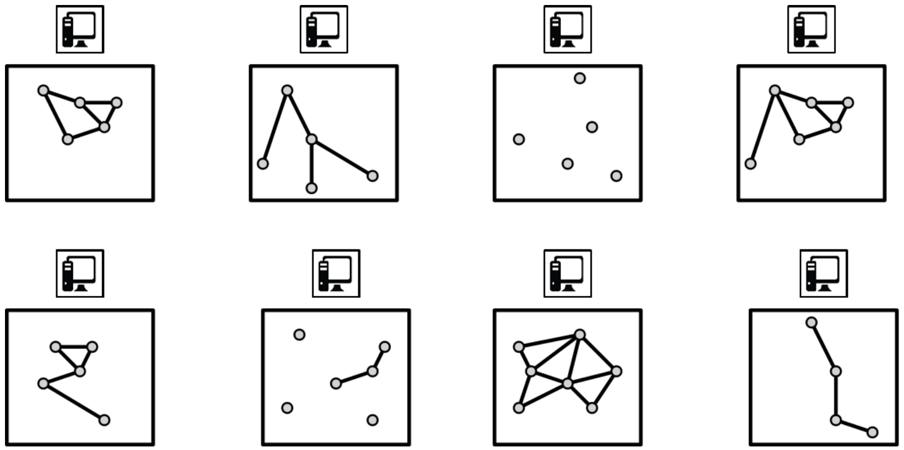
Ideally:
Avoid exponentiation altogether and beat the locality barrier.

At the least:
Come up with new ideas and algorithms.

Probably:
Learn ways to collect local data fast

The Plan

- MPC intro
  - LOCAL VS MPC
  - Locality Barrier
- Known Techniques
  - Sparsification and round compression
  - Derandomization
- New Techniques
  - Total space
  - Careful exponentiation



Locally checkable labeling problems

Careful Exponentiation

Solving LCLs with locality
$\Theta(\log^* n)$

In $O(\log\log^* n)$ rounds of MPC with linear total space.

Meets the locality barrier.

Conditionally optimal with fine print

Global LCLs in Forests

In $O(\log \text{diam})$ MPC rounds with linear total space.

Conditionally optimal

LCLs in the “Tiny” Regime

Theorem [CKP’19]:

Any LCL with deterministic locality $o(\log n)$ can be solved with a canonical (LOCAL) algorithm in $O(\log^* n)$ rounds.

Need a distance-k coloring

Get it by Δ^2-coloring of G^k
Linial: $O(\log^* n)$ local rounds

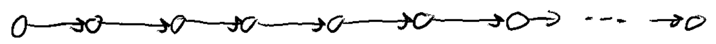
Coloring Pseudo-Forests

Δ^2-coloring of G^k

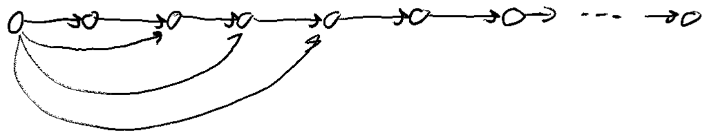
Since Δ and k are constants, can reduce to 3-coloring pseudo-forests and color reduction.

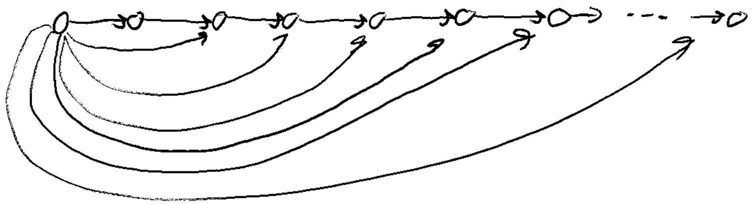
Important: Focus on MPC issues.

A tempting approach:
Gather $O(\log^* n)$-neighborhood
And simulate Linial's



Requires $\Omega(n \log^* n)$ total space!





Coloring a Directed Pseudo-Forest

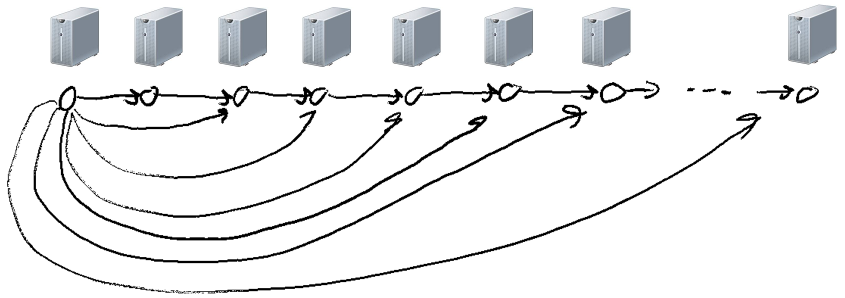
Careful Exploration

Run just one round of Linial's
- Turn IDs into $\log\log n$ -bit colors

Collect a vector of size $O(\log\log n \cdot \log^* n) = O(\log n)$ bits

Total space: $O(n)$ words.

Issue:
Need to store $O(\log^* n)$ machine addresses of $\Omega(\log n)$ bits.



Coloring a Directed Pseudo-Forest

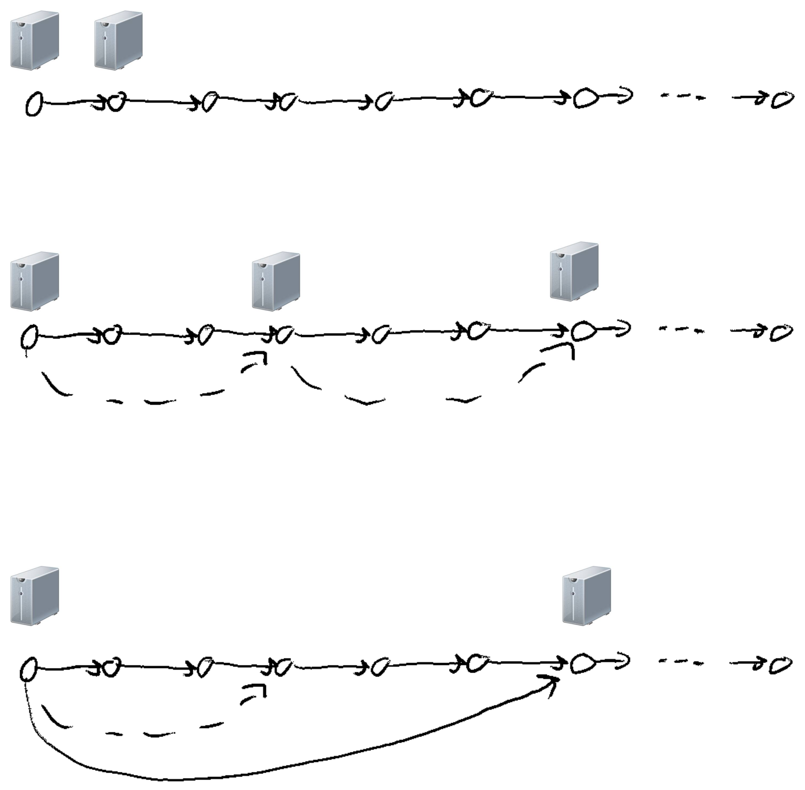
Careful Exploration

Run just one round of Linial's
 - Turn IDs into $\log\log n$ -bit colors

Collect a vector of size $O(\log\log n \cdot \log^* n) = O(\log n)$ bits

Only store the address of farthest machine, $O(\log n)$ bits.

Total space: $O(n)$ words.



LCLs in the "Tiny" Regime

Theorem [BBFLMOU'20]

For any LCL P with locality $o(\log n)$, there is an MPC algorithm that solves P in $O(\log \log^* n)$ rounds.

Nice property:
Optimal in terms of memory parameters.

Nice property:
Goes beyond naïve exponentiation.

Nice property:
Runtime potentially optimal. 🤨

Conditional Lower Bounds

Open Question

Is there a component unstable algorithm that beats the locality barrier for a locally checkable problem?

Affirmative for approximation.

Theorem [GKU'19, CPD'21]

Given the connectivity conjecture, there is no component-stable algorithm that beats $O(\log \text{locality})$

Connectivity Conjecture

It takes $\Omega(\log \text{diam})$ time to find connected components.

Component stable

Outputs on different components are independent

LCLs in the "Tiny" Regime

Theorem [BBFLMOU'20]

For any LCL P with locality $o(\log n)$, there is an MPC algorithm that solves P in $O(\log \log^* n)$ rounds.

Nice property:
Optimal in terms of memory parameters.

Nice property:
Goes beyond naïve exponentiation.

Nice property:
Runtime potentially optimal. 🤨

The Plan

- MPC intro
  - LOCAL VS MPC
  - Locality Barrier
- Known Techniques
  - Sparsification and round compression
  - Derandomization
- New Techniques
  - Total space
  - Careful exponentiation

Locally checkable labeling problems

Connectivity on Forests

Theorem [BLMOU'23]:
There is an MPC algorithm to find the connected components of a forest in $O(\log \text{diam})$ rounds.

Almost directly yields an algorithm to solve all LCLs on forests

Conditionally optimal

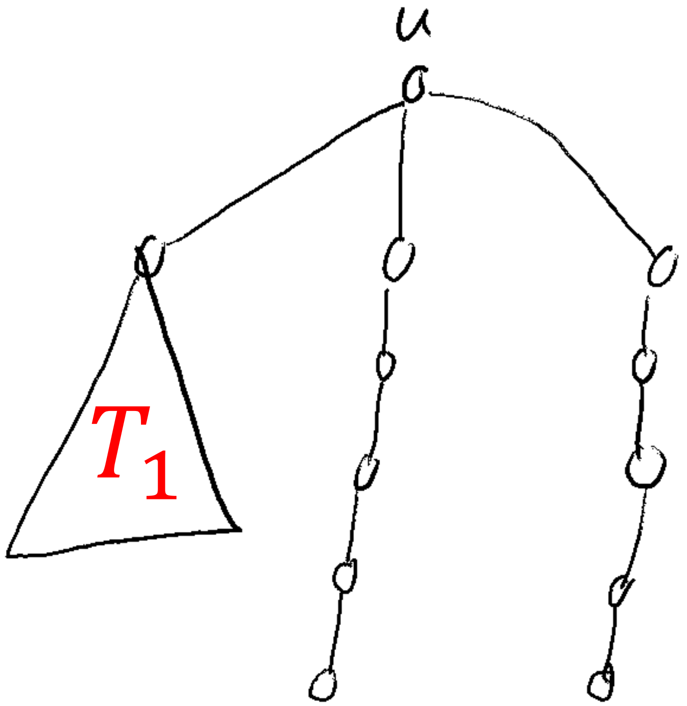
In terms of memory

Holds for component unstable algorithms! 🤨

Finding Leaves

Naïve exponentiation can lead to storing T_1

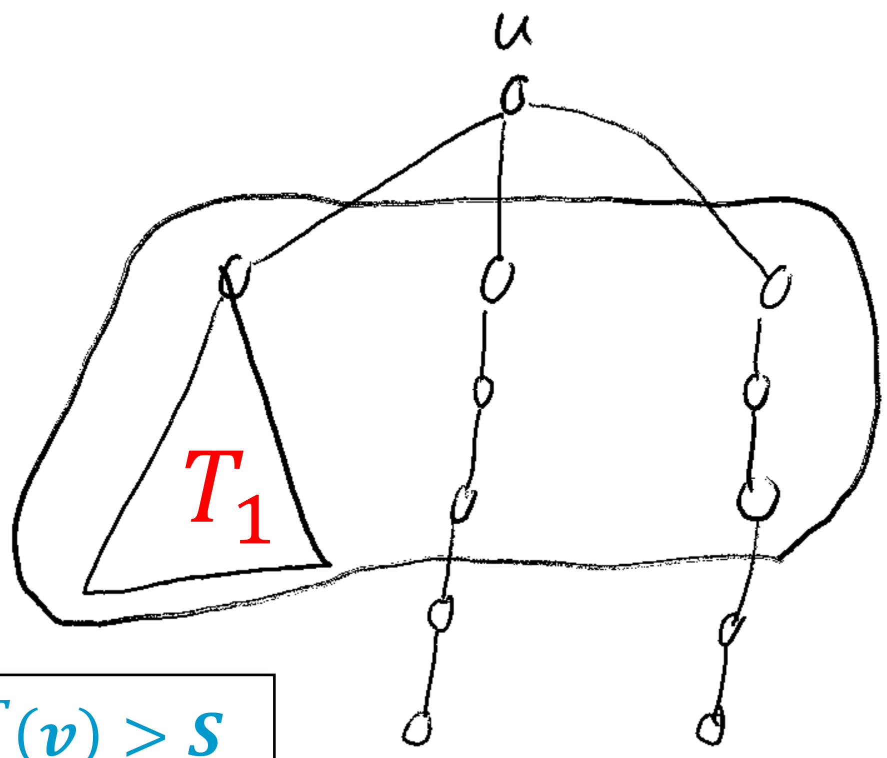
To solve LCLs, need to find leaves from all but one branch.



Finding Leaves

Naïve exponentiation can lead to storing T_1

To solve LCLs, need to find leaves from all but one branch.

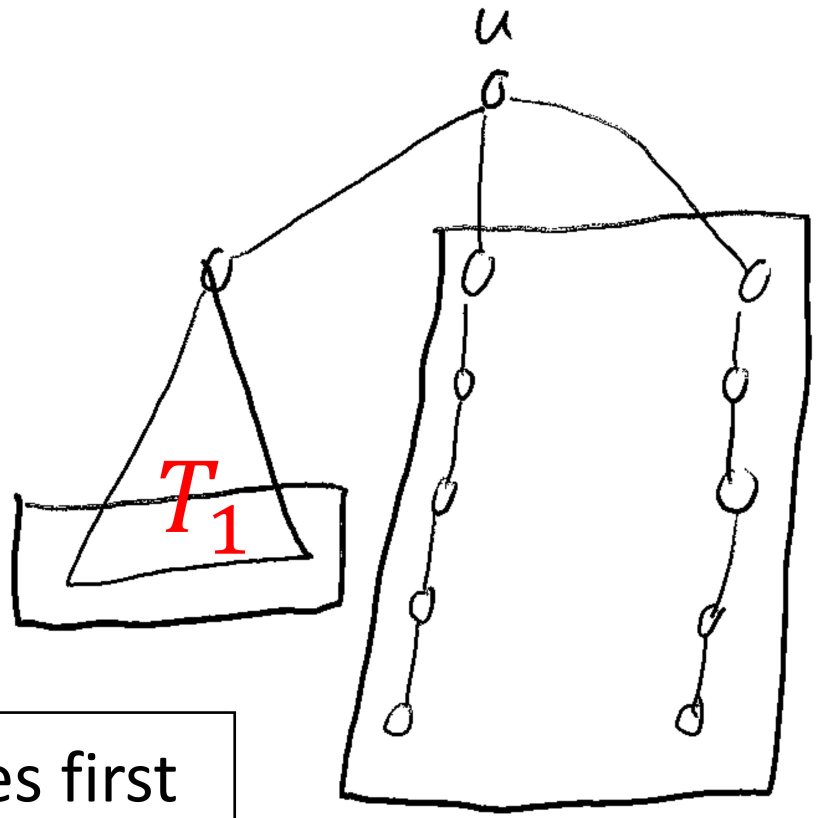


$\mathbf{N}^{T}(v) > S$

Finding Leaves

Naïve exponentiation can lead to storing T_1

To solve LCLs, need to find leaves from all but one branch.

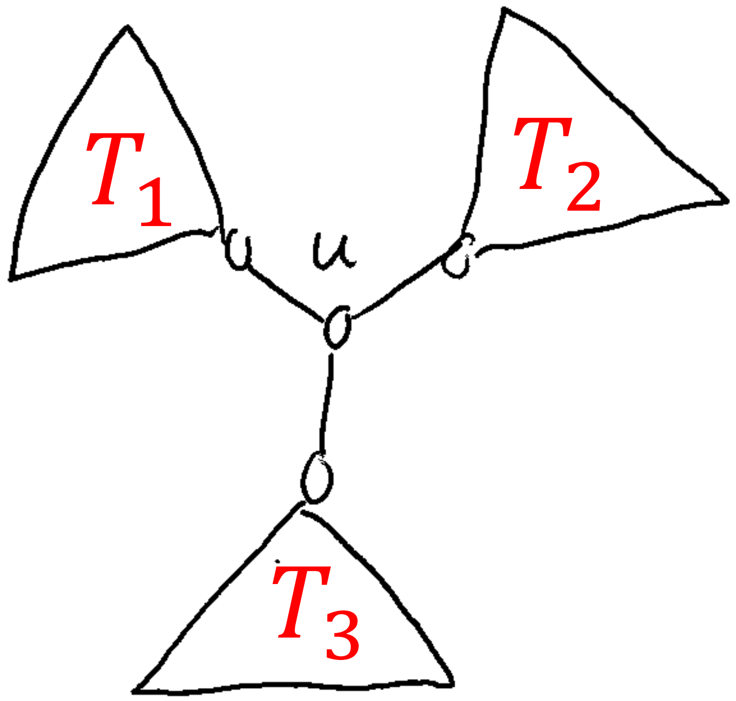


Solve subtrees first

Finding Leaves

To solve LCLs, need to find leaves from all but one branch.

Key idea:
Balanced exploration



Adversary cannot hide leaves to a certain branch

Connectivity on Forests

Theorem [BLMOU'23]:
Connected components of a forest in $O(\log \text{diam})$ rounds.

Almost directly yields an algorithm to solve all LCLs on forests

Nice property:
Global LCLs are hard regardless of component-stability

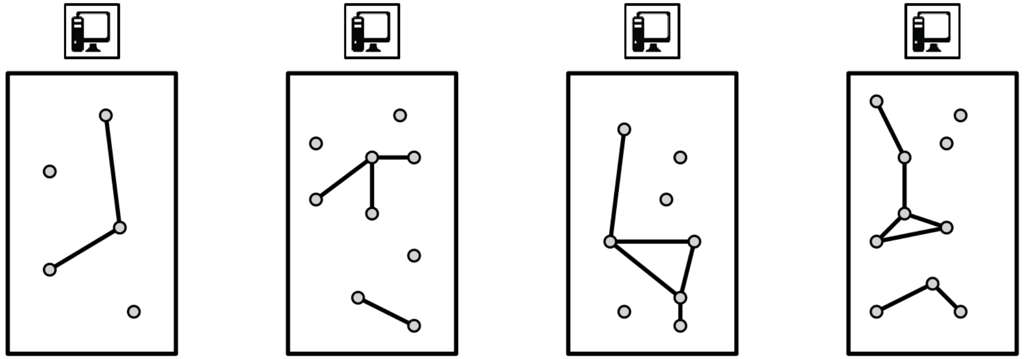
Nice property:
Other connectivity results have a dependency on n

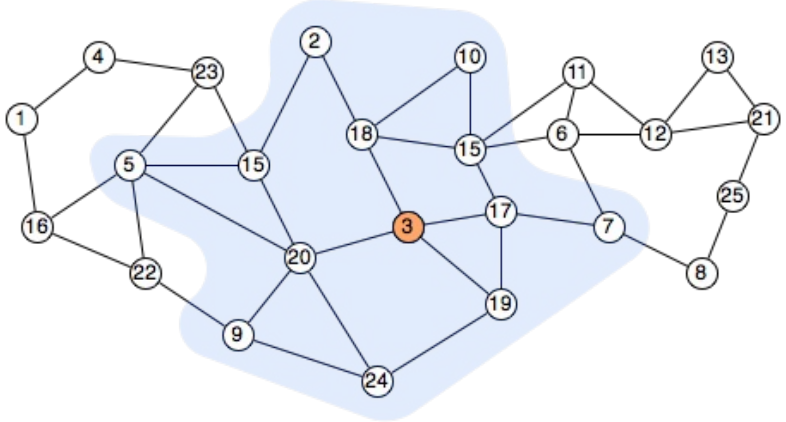
Chicken vs Egg

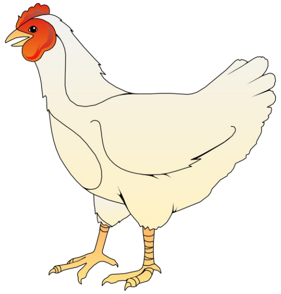
Is there a difference between (?)

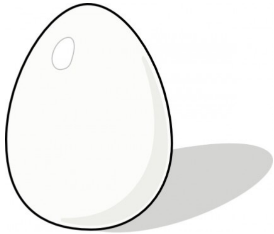
1. First creating a smart subgraph and doing naïve exponentiation

2. Smart exponentiation on the input graph



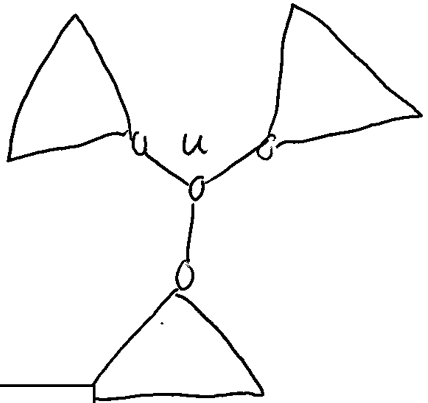


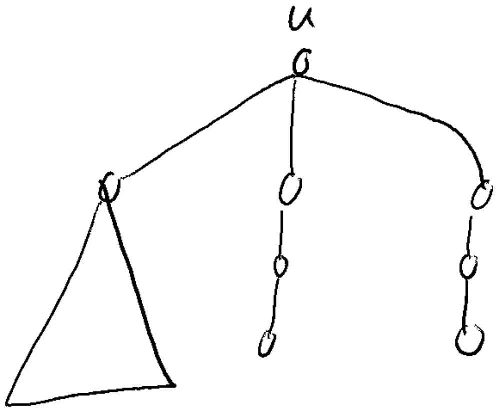




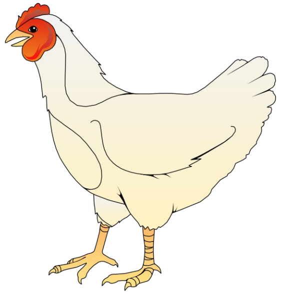
Conclusion

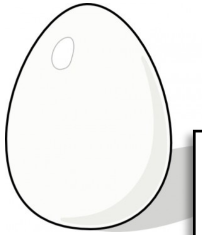
Conditionally optimal algorithms





Graph exponentiation, a necessary evil?





Thank you!
More on Thursday
by Rustam

The Plan

- Known techniques:
  - sampling + solve locally (this is essentially linear space)
    - If I mention this, I should advertise our ruling set talk)
    - Sample and gather by Shreyas (pretty much round compression??)
  - sparsification
    - Round compression + graph exponentiation
  - Deterministic random bits
    - Conditional expectations (check references)
    - Annoying O(1) algorithm for large independent sets
    - PRGs (check references)
  - Shattering
    - Used in combination with round compression of LOCAL
    - post shattering is usually expensive

Where total space is needed

- Finding an MIS of size $\Omega(n/\Delta)$ takes $\Omega(\log^* n)$ in LOCAL and $O(1)$ in low-space MPC.
    - Kawarabayashi, Khoury, Schild, and Schwartzman [KKSS'20]
    - Czumaj, Davies, Parter [CDP'21]
- Derandomization tools for MIS, LLL, etc with $n^{1+o(1)}$ total space [CDP'21]
    - $o(1)$ contains collecting a ball of $\Delta^2 + \mathrm{poly} \log \log n$ radius.
    - $\log \Delta + \log \log \log n$ time for large Δ with $n^{1+\Omega(1)}$ total space
- All randomized algorithms can be derandomized with polynomial number of machines
    - Probability boosting (need to be able to verify correctness)
    - Since success probability is high enough, there is a correct seed (Proof 6.1)

Connectivity

- Randomized algorithms by Andoni, Stein, Song, Wang, Zhong [ASSWZ'18] and Behnezhad, Dhulipala, Esfandiari, Łącki, Mirrokni [BDELM'19] do connectivity in $O(\log D)$ time for dense graphs.
  - Need $\Omega(\log\log n)$ for sparse graph. Needed in order to get concentration.
  - The same bound holds for deterministic algorithms Coy, Czumaj [CC'22]. They use derandomization and hence, at least indirectly, inherently require $\Omega(\log\log n)$.
- Actually, one can shrink the graph by a $\text{poly}\ \log n$ factor in $O(\log\log n)$ rounds which "gives more total space"
  - Nothing wrong with this, but surpassing this bound requires new ideas
  - Explicit disclaimer that I am not saying that this approach cannot lead anywhere
- Sparse graphs are hard?!
  - Can we get $O(\log D)$ in sparse graphs? Topic for another talk?

Locally Checkable Labelings

- Revisit the LOCAL algorithms for LCLs
- On forests
  - Tiny regime: $\Theta(\log^* n)$
  - Mid regime: $\Theta(\log n)$
  - High regime: $\Theta(n^{1/k})$
- Connectivity result gives $O(\log D)$ for forests
  - Leader election
  - Even stronger results through the hierarchical clustering: solve dynamic programming
- How does all of this relate to the LLL (+ other) results by CDP?
  - At least all LCLs do not satisfy any of the LLL criteria

Thinking Inside the Box

Pointer hopping:
Tiny regime of LCLs

Careful exponentiation:
High regime of LCLs

Solving the Tiny Regime

- LOCAL reduction to coloring a directed pseudo-forest

Forest Connectivity

- All LCLs in $O(\log D)$ rounds.
- Careful/balanced exponentiation