The Complexity of (Δ+1) Coloring in Congested Clique, Massively Parallel Computation, and Centralized Local Computation

Yi-Jun Chang

Manuela Fischer

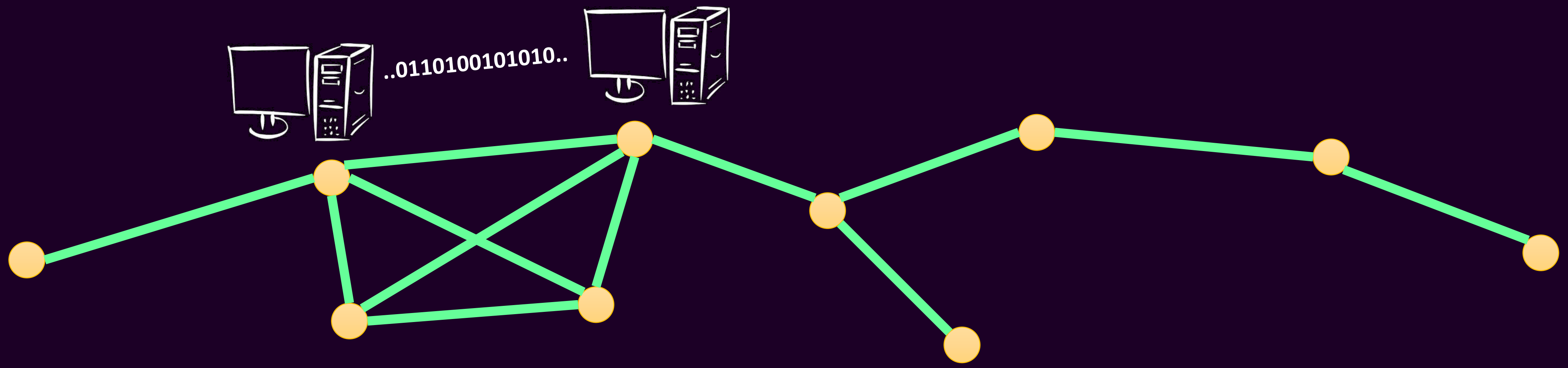
Mohsen Ghaffari

Jara Uitto

Yufan Zheng

(Δ+1) Coloring

- Easy in the sequential setting.
  - A simple sequential greedy algorithm in linear time and space.
- What about the distributed setting?



Two Types of Distributed Models

- Type 1: computer network = input graph
  - LOCAL, CONGEST

With locality

- Type 2: computer network ≠ input graph
  - CONGESTED-CLIQUE, MPC

Without locality

Distributed Models

- LOCAL:

  Can only communicate with neighbors.

  Unbounded message size.

  Locality

- CONGEST:

  Can only communicate with neighbors.

  $O(\log n)$-bit message size.

  Bandwidth constraint

Other features: Synchronous rounds & unbounded local computation power

Distributed Models

- LOCAL: Can only communicate with neighbors. Unbounded message size.
- CONGEST: Can only communicate with neighbors. $O(\log n)$-bit message size.
- Congested Clique: Allow all-to-all communication. $O(\log n)$-bit message size.

Locality

Bandwidth constraint

Distributed Models

- Alternative definition of CONGESTED-CLIQUE:
  - In each round each processor can send and receive up to $O(n)$ messages of $O(\log n)$ bits.
  - Number of processors = n.
  - Initially each processor knows the set of neighbors of a vertex.

(in view of Lenzen's routing)

Distributed Models

- Alternative definition of CONGESTED-CLIQUE:
  - In each round each processor can send and receive up to $O(n)$ messages of $O(\log n)$ bits.
  - Number of processors = n.
  - Initially each processor knows the set of neighbors of a vertex.

- MPC (Massively Parallel Computation) model:
  - A scalable variant of CONGESTED-CLIQUE.
  - Memory per processor = $S = n^{\delta}$ for some $\delta = \Theta(1)$.
  - Number of processors = $\tilde{O}(m \,/\, S)$.
  - Input graph is distributed arbitrarily (can be sorted in O(1) rounds).

(Δ+1)-coloring in the LOCAL Model

- (Rand.) $O(\log n)$ — Luby (STOC'85) and Alon, Babai and Itai (JALG'86)
- (Det.) $2^{O(\sqrt{\log n})}$ — Panconesi, Srinivasan (JALG'96)

- (Rand.) $O(\log \Delta) + 2^{O(\sqrt{\log\log n})}$ — Barenboim, Elkin, Pettie, Schneider (FOCS 2012)
- (Rand.) $O(\sqrt{\log \Delta}) + 2^{O(\sqrt{\log\log n})} = O(\sqrt{\log n})$ — Harris, Schneider, Su (STOC 2016)
- (Rand.) $O(\log^* \Delta) + 2^{O(\sqrt{\log\log n})} = 2^{O(\sqrt{\log\log n})}$ — **Chang**, Li, Pettie (STOC 2018)

Pre-shattering Post-shattering

(There are many more!)

(Δ+1)-coloring in the LOCAL Model

- (Rand.) $O(\log n)$ — Luby (STOC'85) and Alon, Babai and Itai (JALG'86)
- (Det.) $2^{O(\sqrt{\log n})}$ — Panconesi, Srinivasan (JALG'96)

- (Rand.) $O(\log \Delta) + 2^{O(\sqrt{\log \log n})}$ — Barenboim, Elkin, Pettie, Schneider (FOCS 2012)
- (Rand.) $O(\sqrt{\log \Delta}) + 2^{O(\sqrt{\log \log n})} = O(\sqrt{\log n})$ — Harris, Schneider, Su (STOC 2016)
- (Rand.) $O(\log^* \Delta) + 2^{O(\sqrt{\log \log n})} = 2^{O(\sqrt{\log \log n})}$ — **Chang**, Li, Pettie (STOC 2018)

Pre-shattering   Post-shattering

(There are many more!)

What about MPC / CONGESTED-CLIUQUE?

(Δ+1) Coloring in MPC

- "Sublinear Algorithms for (Δ+1) Vertex Coloring" by Sepehr Assadi, Yu Chen, Sanjeev Khanna [SODA 2019]

- Sample $O(\log n)$ colors for each vertex independently and uniformly at random from the $\Delta + 1$ colors.
- With high probability, the graph is colorable using the selected colors.
- This leads to an $O(1)$-round MPC algorithm.

(Δ+1) Coloring in MPC

- “Sublinear Algorithms for (Δ+1) Vertex Coloring” by Sepehr Assadi, Yu Chen, Sanjeev Khanna [SODA 2019]

- Sample $O(\log n)$ colors for each vertex independently and uniformly at random from the $\Delta + 1$ colors.
- With high probability, the graph is colorable using the selected colors.
- This leads to an $O(1)$-round MPC algorithm.

Two issues:

(i) costs polylogarithmic rounds in CONGESTED CLIQUE.

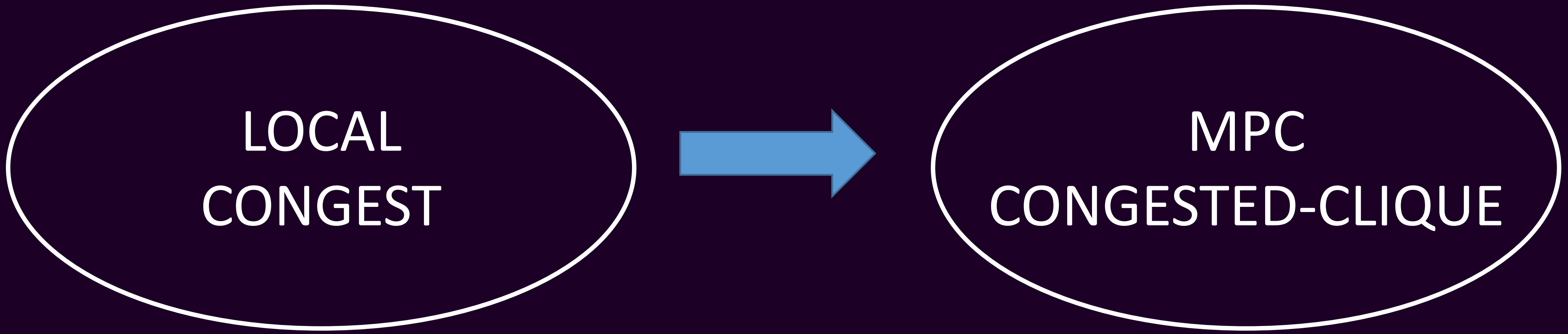
(ii) memory per processor must be $\widetilde{\Omega}(n)$.

We will later see that our approach does not have these issues.

Our Results

- $O(1)$-round CONGESTED-CLIQUE algorithm.
- $O(\sqrt{\log \log n})$-round MPC algorithm in the small memory regime.

- Our approach: transformation from Chang-Li-Pettie algorithm for (Δ+1)-coloring in the LOCAL Model



(Δ+1) Coloring in CONGESTED-CLIQUE

How to implement this algorithm in CONGESTED-CLIQUE?

- $O(\log^* \Delta) \quad + \quad 2^{O(\sqrt{\log\log n})} \quad = \quad 2^{O(\sqrt{\log\log n})}$

Chang, Li, Pettie (STOC 2018)

Pre-shattering

Post-shattering

At this stage, the remaining graph has O(n) edges, so we can send them to one processor. This part can be implemented in CONGESTED-CLIQUE in $O(1)$ rounds.

For this part, some node has to receive messages of size $O(\Delta^2)$, so a naïve simulation works only when $\Delta < \sqrt{n}$.

Prior works

- $O(\log\log n)$ rounds
  - Merav Parter – "(Delta+1) Coloring in the Congested Clique Model" ICALP 2018
  - (the cost for reducing the general case to the $\Delta < \sqrt{n}$ case)
- $O(\log^* \Delta)$ rounds
  - Merav Parter & Hsin-Hao Su – "(Delta+1)-Coloring in O(log* Delta) Congested-Clique Rounds" DISC 2018
  - (modify the internal details of the CLP coloring algorithm to increase the threshold from $\Delta < \sqrt{n}$ to $\Delta < n^{5/8}$)

Our Approach (high-deg case)

- A simple algorithm that deals with the case $\Delta > \log^5 n$ in $O(1)$ rounds.
- Decompose the vertex set and the color set randomly into $\sqrt{\Delta}$ parts: $B_1, B_2, \ldots, B_{\sqrt{\Delta}}$.
  - Each part has $O(n/\sqrt{\Delta})$ vertices and max-deg $\mathrm{O}(\sqrt{\Delta})$.
  - Each part is associated with $\mathrm{O}(\sqrt{\Delta})$ colors.
- We want to color each part with its associated colors.
- But there will be a gap of $\approx \Delta^{1/4}$ between max-degree and # colors.

Our Approach (high-deg case)

- We want to color each part with its associated colors.
- But there will be a gap of $\approx \Delta^{1/4}$ between max-degree and # colors.

- Solution: adjust the probabilities to decrease the max-deg of each part B_1, B_2, …, $B_{\sqrt{\Delta}}$ by $\approx \Delta^{1/4}$, and this leads to a new part L whose size is $\approx n/\Delta^{1/4}$ with max-deg $\approx \Delta^{3/4}$

- Now each of B_1, B_2, …, $B_{\sqrt{\Delta}}$ is colorable with their colors. After coloring them, we can recurse on L.

Our Approach (high-deg case)

- Recall: each of $B_1, B_2, \ldots, B_{\sqrt{\Delta}}$ has $O(n/\sqrt{\Delta})$ vertices and max-deg $O(\sqrt{\Delta})$, so they have $O(n)$ edges. We can send each of them to a processor to construct the coloring locally. This takes $O(1)$ rounds in CONGESTED-CLIQUE.
- A simple calculation shows that when $\Delta > \log^5 n$, after $O(1)$ depth of recursions, the size of L also decreases to $O(n)$ edges.
- This gives us an $O(1)$-round CONGESTED-CLIQUE algorithm.

Our Approach (low-deg case)

- How about the case $\Delta < \log^5 n$? Recall:

  - $O(\log^* \Delta) \;+\; 2^{O(\sqrt{\log\log n})} \;=\; 2^{O(\sqrt{\log\log n})}$

Chang, Li, Pettie (STOC 2018)

Pre-shattering

Post-shattering

At this stage, the remaining graph has O(n) edges, so we can send them to one processor. This part can be implemented in CONGESTED-CLIQUE in $O(1)$ rounds.

For this part, some node has to receive messages of size $O(\Delta^2)$, so a naïve simulation works only when $\Delta < \sqrt{n}$.

Our Approach (low-deg case)

- How about the case $\Delta < \log^5 n$? Recall:
  - $O(\log^* \Delta) + 2^{O(\sqrt{\log\log n})} = 2^{O(\sqrt{\log\log n})}$

Chang, Li, Pettie (STOC 2018)

Pre-shattering Post-shattering

$O(\log^* \Delta) + O(1)$ <- Straightforward simulation

$O(\log\log^* \Delta) + O(1)$ <- Graph exponentiation

After the i-th round, each vertex gathers all information within its radius 2^i neighborhood. We can do this because the degree is small.

Our Approach (low-deg case)

- Can we do better?

- Let's say we have a T-round LOCAL algorithm that we wish to run in the CONGESTED CLIQUE.

$O(T)$ <- Straightforward simulation (when degree and message size is sufficiently small)

$O(\log T)$ <- Graph exponentiation ($\Delta^T < n$)

$O(1)$ <- Straightforward information gathering (# edges = $O(n)$)

Our Approach (low-deg case)

- "Opportunistic" information gathering:
  - Each vertex v sends its edges to random destinations, and it wishes that someone will gather enough information to simulate the algorithm at v.

- $\Pr[\ e \text{ is sent to } u\ ] = p$
  - Need $p = 1/\Delta$ so that each node received only O(n) words.
  - Recall: # edges = $O(n\Delta)$.

- $\Pr[\ v \text{ is successfully simulated by } u\ ] = p^{\Delta^T}$ (need this to be $\gg 1/n$)

This idea is implicit in:
Tomasz Jurdzinski and Krzysztof Nowicki - "MST in O(1) rounds of congested clique" in SODA 2018.

Our Approach (low-deg case)

- For example, it works when $T = O(\log^* n)$ and $\Delta = \text{poly} \log \log n$.
- We "sparsify" the pre-shattering phase of the CLP algorithm to reduce the effective degree from $\Delta = O(\log^5 n)$ to $\Delta = \text{poly} \log \log n$.
- This leads to an $O(1)$-round algorithm in CONGESTED CLIQUE.

The idea of sparsifying local algorithms to obtain better MPC / CONGESTED CLIQUE algorithms appears in:
Mohsen Ghaffari and Jara Uitto – "Sparsifying Distributed Algorithms with Ramifications in Massively Parallel Computation and Centralized Local Computation" in SODA 2019.

Adaptation to MPC

- One issue: memory per processor is $S = n^{\delta}$

- When # edges = $O(n)$, cannot gather all information to one processor.

- Need to recurse on $B_1, B_2, \ldots, B_{\sqrt{\Delta}}$, until they have small degree.
  - depth of recursion is still $O(1)$.

bottleneck

- Post-shattering phase cannot be done in $O(1)$ rounds.
  - Apply graph exponentiation to attain the round complexity of $O(\sqrt{\log \log n})$.

Adaptation to MPC

Conditional lower bound: Mohsen Ghaffari, Fabian Kuhn, and Jara Uitto – "Conditional Hardness Results for Massively Parallel Computation from Distributed Lower Bounds" in FOCS 2019.

- One issue: memory per processor is $S = n^{\delta}$
- When # edges = $O(n)$, cannot gather all information to one processor.
- Need to recurse on $B_1, B_2, \ldots, B_{\sqrt{\Delta}}$, until they have small degree.
  - depth of recursion is still $O(1)$.
- Post-shattering phase cannot be done in $O(1)$ rounds.
  - Apply graph exponentiation to attain the round complexity of $O(\sqrt{\log \log n})$.

bottleneck

Adaptation to MPC

Conditional lower bound: Mohsen Ghaffari, Fabian Kuhn, and Jara Uitto – "Conditional Hardness Results for Massively Parallel Computation from Distributed Lower Bounds" in FOCS 2019.

- One issue: memory per processor is $S = n^{\delta}$
- When # edges = $O(n)$, cannot gather all information to one processor.
- Need to recurse on $B_1, B_2, \ldots, B_{\sqrt{\Delta}}$, until they have small degree.
  - depth of recursion is still $O(1)$.
- Post-shattering phase cannot be done in $O(1)$ rounds.
  - Apply graph exponentiation to attain the round complexity of $O(\sqrt{\log \log n})$.

bottleneck

Thanks for your attention