Peer-to-peer clustering

Po-Shen Loh
Carnegie Mellon University

Joint work with Eyal Lubetzky
**Question**

Each of us has a number. How fast can we calculate the sum?

5  2  3  1  2  4  3  5
**How to Add**

**Question**

Each of us has a number. How fast can we calculate the sum?

```
5  2  3  1  2  4  3  5
11 14
25
7  4  6  8
log_2 n
```
How to add

**Question**
Each of us has a number. How fast can we calculate the sum?

```
5 2 3 1 2 4 3 5
11 14
25
7 4 6 8
log n^2
```

**Main issue**
- Creating the “clustering” tree may take a long time.
- Can it be done in a distributed manner?
Simplest parallelization (D. Malkhi)

- Start with $n$ clusters of size 1.
- Every round:
  - Each cluster flips coin to decide state: req or acc.
  - Each req cluster sends request to random cluster.
  - Each acc chooses random incoming request to merge.
Clustering protocol

Simplest parallelization (D. Malkhi)

- Start with $n$ clusters of size 1.
- Every round:
  - Each cluster flips coin to decide state: req or acc.
  - Each req cluster sends request to random cluster.
  - Each acc chooses random incoming request to merge.

Peer-to-peer context:

- Clusters cannot have full knowledge of who is in other clusters.
- Basic operation: sample random atom.
Clustering protocol

Simplest parallelization (D. Malkhi)

- Start with $n$ clusters of size 1.
- Every round:
  - Each cluster flips coin to decide state: req or acc.
  - Each req cluster sends request to random cluster.
  - Each acc chooses random incoming request to merge.

Peer-to-peer context:

- Clusters cannot have full knowledge of who is in other clusters.
- Basic operation: sample random atom.
- Atoms keep track of their parent.
  If no parent, then atom is root.
Simplest parallelization (D. Malkhi)

- Start with \( n \) clusters of size 1.
- Every round:
  - Each cluster flips coin to decide state: \text{req} or \text{acc}.
  - Each \text{req} cluster sends request to random cluster.
  - Each \text{acc} chooses random incoming request to merge.

Peer-to-peer context:

- Clusters cannot have full knowledge of who is in other clusters.
- Basic operation: sample random \textit{atom}.
- Atoms keep track of their \textit{parent}.
  If no parent, then atom is \textit{root}. 
Simplest parallelization (D. Malkhi)

- Start with $n$ clusters of size 1.
- Every round:
  - Each cluster flips coin to decide state: **req** or **acc**.
  - Each **req** cluster sends request to random cluster.
  - Each **acc** chooses random incoming request to merge.

Peer-to-peer context:

- Clusters cannot have full knowledge of who is in other clusters.
- Basic operation: sample random **atom**.
- Atoms keep track of their **parent**.
  - If no parent, then atom is **root**.
- Clusters sampled **proportional to size**.
Clustering protocol

Simplest parallelization (D. Malkhi)

- Start with \( n \) clusters of size 1.
- Every round:
  - Each cluster flips coin to decide state: req or acc.
  - Each req cluster sends request to random cluster.
  - Each acc chooses random incoming request to merge.

Peer-to-peer context:

- Clusters cannot have full knowledge of who is in other clusters.
- Basic operation: sample random atom.
- Atoms keep track of their parent. If no parent, then atom is root.
- Clusters sampled proportional to size.
- acc choose uniformly over incoming.
Clustering protocol

Empirical observation

In simulations, distributed protocol takes $O(\log n)$ time to finish.
Empirical observation

In simulations, distributed protocol takes $O(\log n)$ time to finish.

Challenge to analysis:

$n - \sqrt{n}$
Empirical observation

In simulations, distributed protocol takes $O(\log n)$ time to finish.

Challenge to analysis:

- Singleton #1 sends request to #2 with probability $\frac{1}{n}$.
- Number of pairs of singletons is $(\sqrt{n})^2$.
- Each round, only constant number of singletons merge.
- Running time could be polynomial.
Clustering protocol

Empirical observation
In simulations, distributed protocol takes $O(\log n)$ time to finish.

Challenge to analysis:

- Singleton #1 sends request to #2 with probability $\frac{1}{n}$.
- Number of pairs of singletons is $(\sqrt{n})^2$.
- Each round, only constant number of singletons merge.
- Running time could be polynomial.
If requesters contact *uniformly random* clusters, then the process completes in $O(\log n)$ rounds.
**Random-Mate Algorithm**

If requesters contact *uniformly random* clusters, then the process completes in $O(\log n)$ rounds.

**Sketch:** Let $\kappa$ be number of clusters.

- Each cluster receives $\text{Bin}[\kappa, \frac{1}{2\kappa}]$ requests.
- Number of clusters reduces by constant factor every round. □

---

**Analogy to connectivity in graph processes**

Above: Merge two uniformly sampled components.

Erdős-Rényi: Sample two components, proportional to size.

Peer-to-peer clustering: Sample one uniform component, and one proportional to size.
Related processes

**Random-Mate algorithm**

If requesters contact *uniformly random* clusters, then the process completes in $O(\log n)$ rounds.

**Sketch:** Let $\kappa$ be number of clusters.
- Each cluster receives $\text{Bin}\left[\kappa, \frac{1}{2\kappa}\right]$ requests.
- Number of clusters reduces by constant factor every round. □

**Analogy to connectivity in graph processes**
- **Above:** Merge two uniformly sampled components.
- **Erdős-Rényi:** Sample two components, proportional to size.
**Random-Mate algorithm**

If requesters contact *uniformly random* clusters, then the process completes in $O(\log n)$ rounds.

**Sketch:** Let $\kappa$ be number of clusters.
- Each cluster receives Bin$[\kappa, \frac{1}{2\kappa}]$ requests.
- Number of clusters reduces by constant factor every round. □

**Analogy to connectivity in graph processes**

- **Above:** Merge two uniformly sampled components.
- **Erdős-Rényi:** Sample two components, proportional to size.
- **Peer-to-peer clustering:** Sample one uniform component, and one proportional to size.
The distributed protocol finishes in $O(\sqrt{n})$ time, and takes at least $\log_2 n$ time.

Conjecture (Schramm)

The distributed protocol takes $\omega(\log n)$ time to complete.

Theorem (L., Lubetzky)

The distributed protocol takes at least $\log n \cdot \log \log n \log \log \log n$ time with high probability (whp).
**Results**

**Previous bounds**

The distributed protocol finishes in $O(\sqrt{n})$ time, and takes at least $\log_2 n$ time.

**Conjecture (Schramm)**

The distributed protocol takes $\omega(\log n)$ time to complete.
PREVIOUS BOUNDS
The distributed protocol finishes in $O(\sqrt{n})$ time, and takes at least $\log_2 n$ time.

CONJECTURE (SCHRAMM)
The distributed protocol takes $\omega(\log n)$ time to complete.

THEOREM (L., LUBETZKY)
The distributed protocol takes at least $\log n \cdot \frac{\log \log n}{\log \log \log n}$ time whp.
Theorem (L., Lubetzky)

If accepters choose their *smallest* incoming request,* then the process completes in $O(\log n)$ rounds whp.

* ignoring requests from clusters larger than themselves
Theorem (L., Lubetzky)
If accepters choose their *smallest* incoming request,* then the process completes in $O(\log n)$ rounds \textit{whp}.

* ignoring requests from clusters larger than themselves
Theorem (L., Lubetzky)

If accepters choose their \textit{smallest} incoming request,\footnote{ignoring requests from clusters larger than themselves} then the process completes in \(O(\log n)\) rounds \textit{whp}.

Implementation details:

- Roots know their cluster size.
Size-biased protocol

**Theorem (L., Lubetzky)**

If accepters choose their *smallest* incoming request,* then the process completes in $O(\log n)$ rounds whp.

* ignoring requests from clusters larger than themselves

**Implementation details:**

- Roots know their cluster size.
- Add at merge, pass to new root.
Theorem (L., Lubetzky)

If accepters choose their *smallest* incoming request,* then the process completes in $O(\log n)$ rounds whp.

* ignoring requests from clusters larger than themselves

Implementation details:

- Roots know their cluster size.
- Add at merge, pass to new root.

Remarks

- Easier to select smallest incoming, rather than uniform.
- Size-biased protocol faster in practice as well.
Proof Techniques

Challenge

Tracking the number of clusters alone is not enough. Also need control over the cluster size distribution.

Definition

Normalized sizes $c_1, \ldots, c_\kappa$; let the susceptibility be $\chi = \sum c_i^2$. Remarks: This is the expected size of the cluster containing a uniformly sampled atom; the initial value of $\chi$ is $1\kappa$. In the Erdős-Rényi random graph process, adding an edge typically increases $\chi$ by:

$$\Delta \chi = (a + b)^2 - a^2 - b^2 = 2ab \approx 2\chi^2.$$
Challenge
Tracking the number of clusters alone is not enough. Also need control over the cluster size distribution.

Definition
Normalized sizes $c_1, \ldots, c_\kappa$; let the susceptibility be $\chi = \sum c_i^2$.

Remarks:
- This is the expected size of the cluster containing a uniformly sampled atom; the initial value of $\chi$ is $\frac{1}{\kappa}$. 
**Challenge**

Tracking the number of clusters alone is not enough. Also need control over the cluster size distribution.

**Definition**

Normalized sizes $c_1, \ldots, c_\kappa$; let the **susceptibility** be $\chi = \sum c_i^2$.

**Remarks:**

- This is the expected size of the cluster containing a uniformly sampled atom; the initial value of $\chi$ is $\frac{1}{\kappa}$.
- In the Erdős-Rényi random graph process, adding an edge typically increases $\chi$ by:

$$\Delta \chi = (a + b)^2 - a^2 - b^2 = 2ab$$
**Challenge**

Tracking the number of clusters alone is not enough. Also need control over the cluster size distribution.

**Definition**

Normalized sizes $c_1, \ldots, c_\kappa$; let the susceptibility be $\chi = \sum c_i^2$.

**Remarks:**

- This is the expected size of the cluster containing a uniformly sampled atom; the initial value of $\chi$ is $\frac{1}{\kappa}$.
- In the Erdős-Rényi random graph process, adding an edge typically increases $\chi$ by:

\[
\Delta \chi = (a + b)^2 - a^2 - b^2 = 2ab \approx 2\chi^2.
\]
Claim
The susceptibility does not grow beyond \((\text{constant}) \times \frac{1}{\kappa}\).

Idea:
- A cluster which becomes too large receives many requests.

Conditioned on their number, the incoming requests at a given cluster are uniformly distributed over all clusters. Larger clusters have higher probability of receiving a very small cluster, so they grow more slowly.

Claim About \(\chi \kappa\)-fraction of clusters merge at each round.

Idea:
- If \(\chi\) is bounded, there can be large clusters, but only few. Distribution is leveled (controlled by \(\chi\)).
Claim

The susceptibility does not grow beyond \((\text{constant}) \times \frac{1}{\kappa}\).

Idea:

- A cluster which becomes too large receives many requests.
- Conditioned on their number, the incoming requests at a given cluster are uniformly distributed over all clusters.
- Larger clusters have higher probability of receiving a very small cluster, so they grow more slowly.
**Claim**

The susceptibility does not grow beyond \((\text{constant}) \times \frac{1}{\kappa} \). 

**Idea:**

- A cluster which becomes too large receives many requests.
- Conditioned on their number, the incoming requests at a given cluster are uniformly distributed over all clusters.
- Larger clusters have higher probability of receiving a very small cluster, so they grow more slowly.

**Claim**

About \(\frac{1}{\chi^\kappa}\)-fraction of clusters merge at each round.
Intuition for size-biased protocol

Claim
The susceptibility does not grow beyond \((\text{constant}) \times \frac{1}{\kappa}\).

Idea:
- A cluster which becomes too large receives many requests.
- Conditioned on their number, the incoming requests at a given cluster are uniformly distributed over all clusters.
- Larger clusters have higher probability of receiving a very small cluster, so they grow more slowly.

Claim
About \(\frac{1}{\chi \kappa}\)-fraction of clusters merge at each round.

Idea:
- If \(\chi\) is bounded, there can be large clusters, but only few.
- Distribution is leveled (controlled by \(\chi\)).
Problem:

- Susceptibility is not bounded by $O\left(\frac{1}{\kappa}\right)$. 
Problem:

- Susceptibility is not bounded by $O\left(\frac{1}{\kappa}\right)$.
- Carefully define event $E$ such that on the level of expectation:
  - On $E$, $\chi\kappa$ drops.
  - On $\overline{E}$, $\chi\kappa$ doesn’t grow much, but $\kappa$ shrinks by a constant factor.
- Track potential function: $\chi\kappa + 10^7 \log \kappa$. 

Tools:
- Talagrand’s concentration inequality for certifiable random variables on product spaces.
- Optional Stopping Theorem for martingales.
- Freedman’s $L_2$ martingale tail inequality.
**Problem:**

- Susceptibility is not bounded by $O\left(\frac{1}{\kappa}\right)$.
- Carefully define event $E$ such that on the level of expectation:
  - On $E$, $\chi_{\kappa}$ drops.
  - On $\overline{E}$, $\chi_{\kappa}$ doesn’t grow much, but $\kappa$ shrinks by a constant factor.
- Track potential function: $\chi_{\kappa} + 10^7 \log \kappa$.

**Tools:**

- Talagrand’s concentration inequality for *certifiable* random variables on product spaces.
- Optional Stopping Theorem for martingales.
- Freedman’s $L^2$ martingale tail inequality.
Challenge:
Evolution of susceptibility depends on more than its previous value.

Approach (Schramm):
Track all moments of size distribution, with Laplace transform:

\[ L(s) = \kappa \sum e^{-s_i s} \]

Let \( \ell_t(s) \) be \( L(\kappa s) \) after \( t \)-th round. Then \( 1 - \ell_t(\frac{1}{2}) \) is rate of clustering.

\[ \ell_0(s) = e^{-s} \]

\[ \ell_{t+1}(s) = \frac{1}{1 + \ell_t(\frac{1}{2})} \left( \ell_t(s) \cdot \left( \frac{1}{1 + \ell_t(\frac{1}{2})} \right)^2 - \ell_t(s) \cdot \left( \frac{1}{1 + \ell_t(\frac{1}{2})} \right)^2 \cdot \frac{1}{2} + s \cdot \left( \frac{1}{1 + \ell_t(\frac{1}{2})} \right)^2 \right) \]
**Challenge**
Evolution of susceptibility depends on more than its previous value.

**Approach (Schramm)**
Track all moments of size distribution, with *Laplace transform*:

\[
L(s) = \frac{1}{\kappa} \sum e^{-c_i s}.
\]
Challenge
Evolution of susceptibility depends on more than its previous value.

Approach (Schramm)
Track all moments of size distribution, with Laplace transform:

\[ L(s) = \frac{1}{\kappa} \sum e^{-c_i s}. \]

Let \( \ell_t(s) \) be \( L(\kappa s) \) after \( t \)-th round. Then \( 1 - \ell_t(\frac{1}{2}) \) is rate of clustering.
Challenge

Evolution of susceptibility depends on more than its previous value.

Approach (Schramm)

Track all moments of size distribution, with Laplace transform:

\[ L(s) = \frac{1}{\kappa} \sum e^{-c_is}. \]

Let \( \ell_t(s) \) be \( L(\kappa s) \) after \( t \)-th round. Then \( 1 - \ell_t(\frac{1}{2}) \) is rate of clustering.

\[
\ell_0(s) = e^{-s}
\]

\[
\ell_{t+1}(s) = \ldots \text{depends only on 3 evaluations of } \ell_t(\cdot) \ldots
\]
**Challenge**

Evolution of susceptibility depends on more than its previous value.

**Approach (Schramm)**

Track *all* moments of size distribution, with *Laplace transform*:

\[
L(s) = \frac{1}{\kappa} \sum c_i e^{-c_i s}. \]

Let \( \ell_t(s) \) be \( L(\kappa s) \) after \( t \)-th round. Then \( 1 - \ell_t(\frac{1}{2}) \) is rate of clustering.

\[
\ell_0(s) = e^{-s}
\]

\[
\ell_{t+1}(s) = \frac{1}{1 + \ell_t(\frac{1}{2})} \left[ \ell_t \left( s \cdot \frac{1 + \ell_t(\frac{1}{2})}{2} \right)^2 - \ell_t \left( s \cdot \frac{1 + \ell_t(\frac{1}{2})}{2} \right) \ell_t \left( \frac{1}{2} + s \cdot \frac{1 + \ell_t(\frac{1}{2})}{2} \right) + \ell_t \left( \frac{1}{2} + s \cdot \frac{1 + \ell_t(\frac{1}{2})}{2} \right) + \ell_t \left( \frac{1}{2} \right) \ell_t \left( s \cdot \frac{1 + \ell_t(\frac{1}{2})}{2} \right) \right]
\]
Re-interpretation

Show by induction: the functions $\ell_t(\cdot)$ are convex combinations of negative exponentials.
Convexity

Re-interpretation

- Show by induction: the functions $\ell_t(\cdot)$ are convex combinations of negative exponentials.
- Rewrite the recursion for $\ell_{t+1}(\cdot)$ as a weighted arithmetic mean of two evaluations of $\ell_t(\cdot)$. 

\[ \ell_{t+1}(\cdot) \]

\[ \ell_t(\cdot) \]

\[ \ell_t\left(\frac{1}{2}\right) \]

\[ \ell_t(s) \]

\[ 1 \]

\[ s \]
Show by induction: the functions $\ell_t(\cdot)$ are convex combinations of negative exponentials.

Rewrite the recursion for $\ell_{t+1}(\cdot)$ as a weighted arithmetic mean of two evaluations of $\ell_t(\cdot)$.

Therefore, $\ell_t(\frac{1}{2})$ always rises by some tangible amount.
Main contributions.

- The simplest parallelization of the centralized clustering protocol is not optimal.
- The next-simplest, where accepters choose their *smallest* incoming request, achieves optimal performance.
- We demonstrate the usefulness of: susceptibility, Laplace transform.
Main contributions.

- The simplest parallelization of the centralized clustering protocol is not optimal.
- The next-simplest, where accepters choose their *smallest* incoming request, achieves optimal performance.
- We demonstrate the usefulness of: susceptibility, Laplace transform.

**Question**

What is the true behavior of the original protocol?
Main contributions.

- The simplest parallelization of the centralized clustering protocol is not optimal.
- The next-simplest, where accepters choose their \textit{smallest} incoming request, achieves optimal performance.
- We demonstrate the usefulness of: susceptibility, Laplace transform.

**Question**

What is the true behavior of the original protocol?

**Empirical results**

For $n = 10^6 \approx 2^{20}$, original protocol takes 135 rounds, while size-biased protocol takes 75 rounds.
Main contributions.

- The simplest parallelization of the centralized clustering protocol is not optimal.
- The next-simplest, where accepters choose their smallest incoming request, achieves optimal performance.
- We demonstrate the usefulness of: susceptibility, Laplace transform, and theoreticians!

**Question**

What is the true behavior of the original protocol?

**Empirical results**

For $n = 10^6 \approx 2^{20}$, original protocol takes 135 rounds, while size-biased protocol takes 75 rounds.