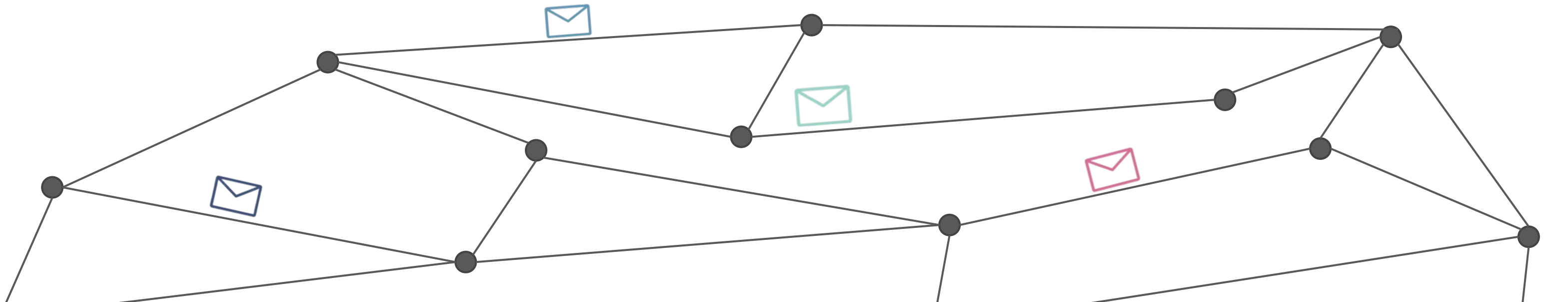
Broadcast in Almost Mixing Time

Anton Paramonov and Roger Wattenhofer

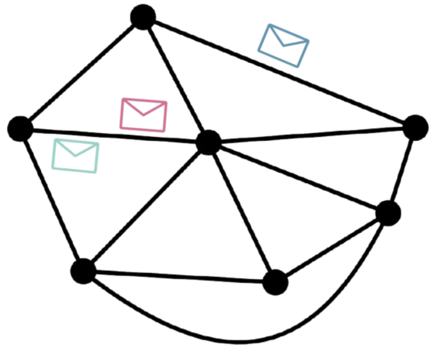
ETH zürich

Model & Problem



CONGEST Model

- Undirected graph with *n* nodes
- Synchronous rounds
- Messages of size *O(log n)*
- Distributed algorithm



Problem: Multi-message broadcast

A dedicated node *s* possesses a set of *k* messages, where each message has a size of $O(\log n)$ bits. The objective is to ensure that every node in the network learns all messages.

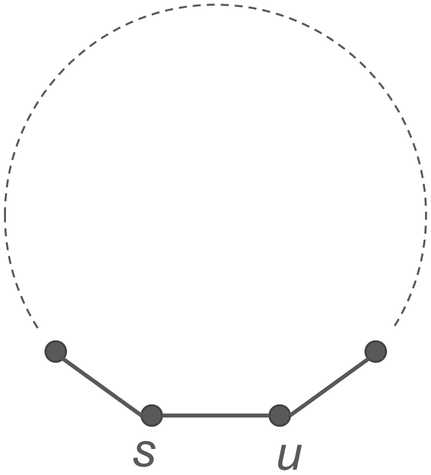
Problem: Multi-message broadcast

A dedicated node s possesses a set of k messages, where each message has a size of $O(\log n)$ bits. The objective is to ensure that every node in the network learns all messages.

Minimize the number of rounds.

Rounds vs Throughput

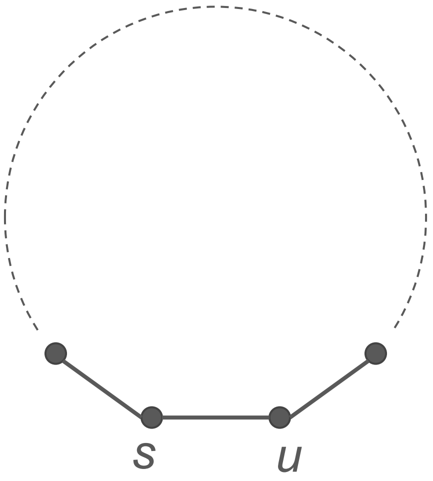
$k = n$



Rounds vs Throughput

k = *n*

Throughput = 2

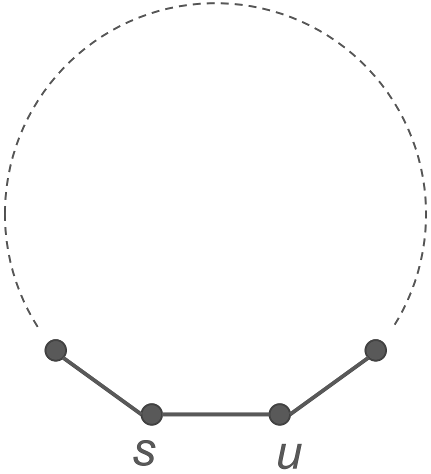


Rounds vs Throughput

$k = n$

Throughput = 2

$\not\Rightarrow$ n/2 rounds!

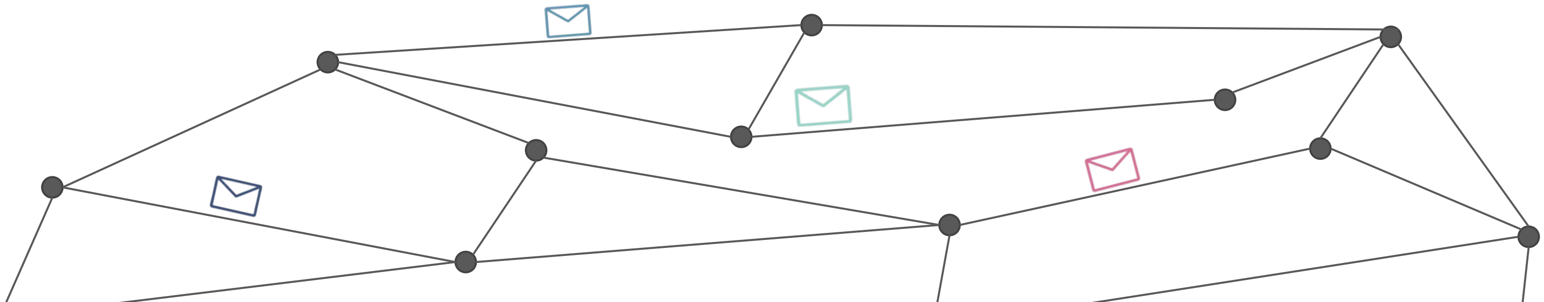


Main result

Main result

There exists a distributed randomized algorithm that for an expander graph G solves a multi-message broadcast in $\tilde{O}(k/\delta(G))$ *rounds with high probability.*

Preliminaries



Random Walk

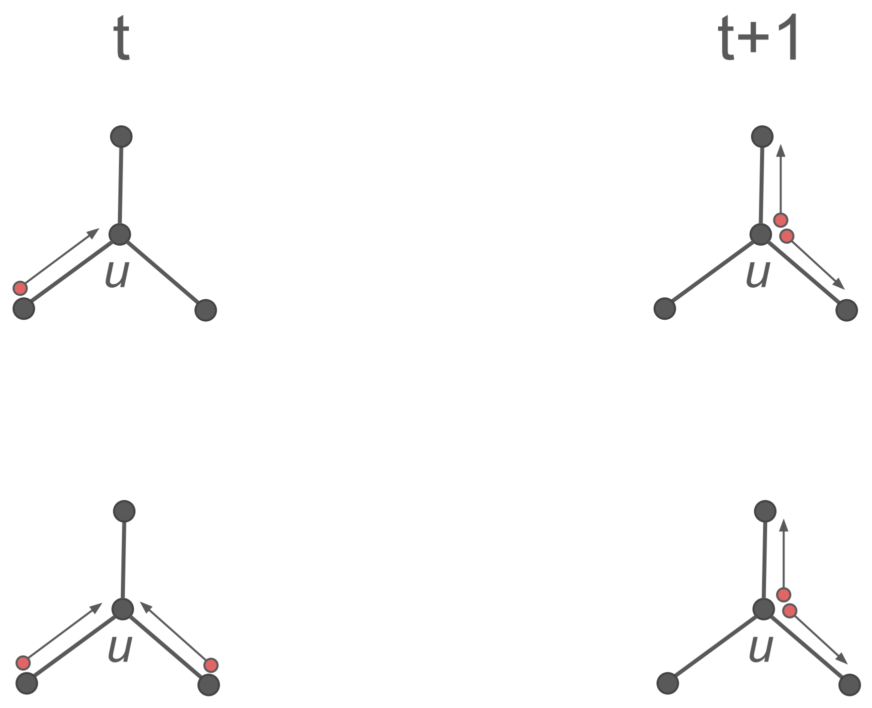
Expanders

Graphs having polylogarithmic mixing time

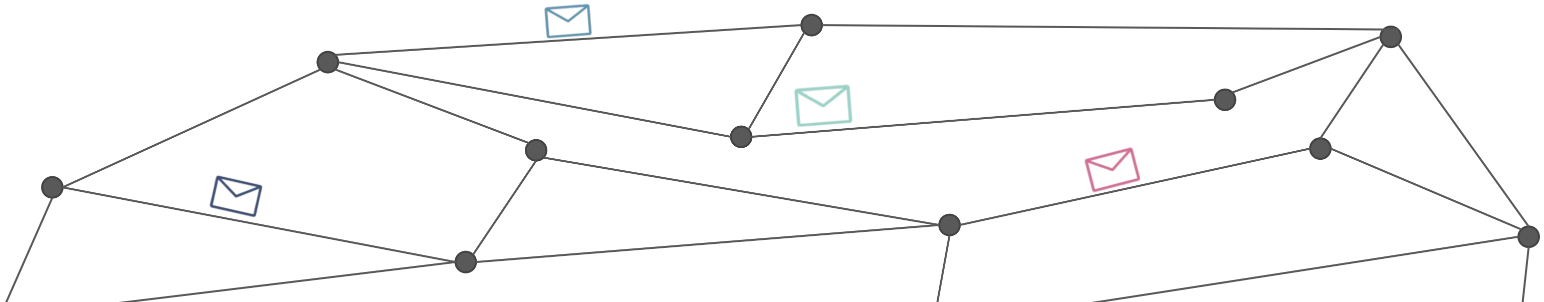
- Erdős–Rényi graph
- Hypercube
- Ramanujan graph
- …

COalescing BRAnching random walk

COalescing BRAnching random walk



Tree packing



Let’s pack!

Downcasting *k’* messages in a tree of depth *D* takes *O(D + k’)* rounds.

Let’s pack!

Downcasting k' messages in a tree of depth D takes $O(D + k')$ rounds.

⇒

Downcasting k messages in T edge-disjoint trees of depth D takes $O(D + k/T)$ rounds!

Tree packing

Nash-Williams:
If minCut is λ, can pack $\lambda/2$ edge-disjoint spanning trees.

Tree packing

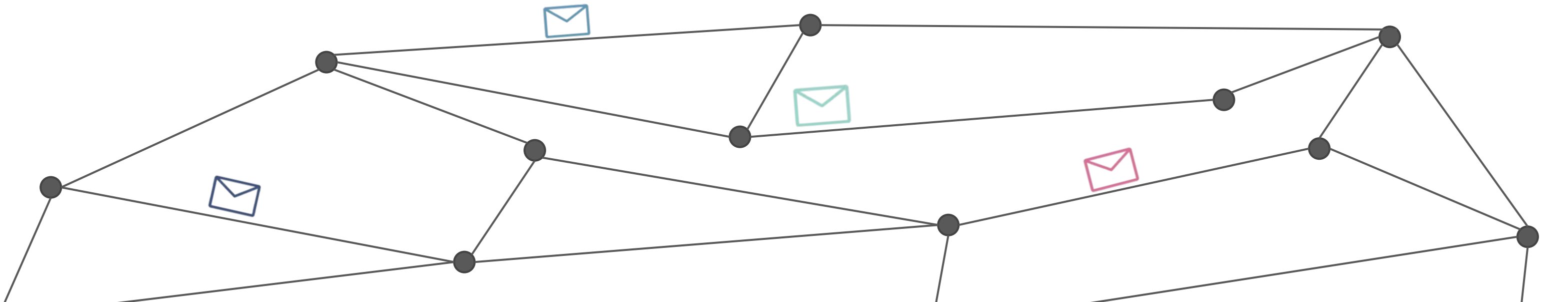
Nash-Williams:
If minCut is λ, can pack $\lambda/2$ edge-disjoint spanning trees.

What about trees of depth <= *D*?

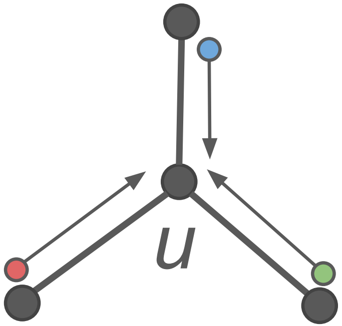
Don’t need them edge-disjoint

Congestion $O(\log n) \Rightarrow O(\log n)$ times slower

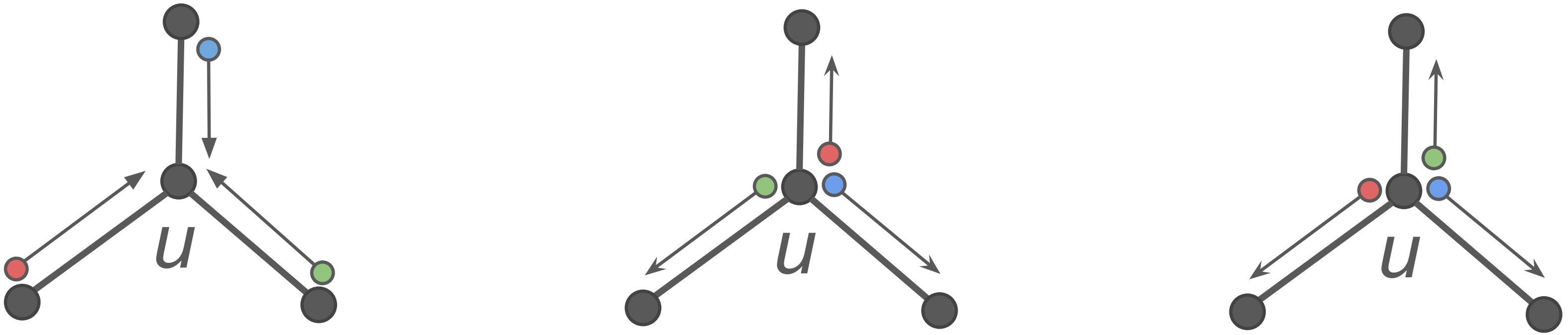
COBRAs for Tree packing



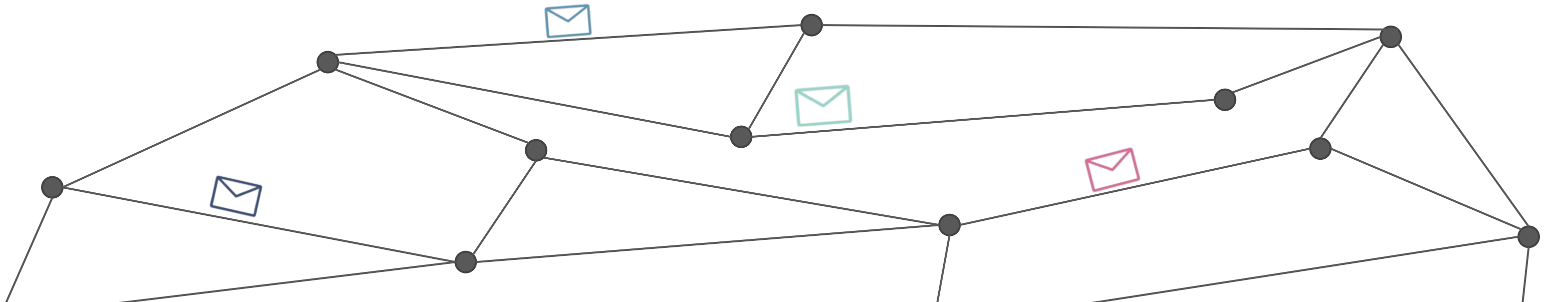
δ COBRAs



δ COBRAs



The Algorithm



Algorithm

1. Launch $\delta(G)$ COBRAs
2. Wait for *O(log n)* rounds
3. Each COBRA results in a spanning subgraph. Carve a spanning tree
4. Distribute and downcast messages

Algorithm

1. Launch δ(G) COBRAs
2. Wait for $\tilde{O}(1)$ rounds
3. Each COBRA results in a spanning subgraph. Carve a spanning tree
4. Distribute and downcast messages

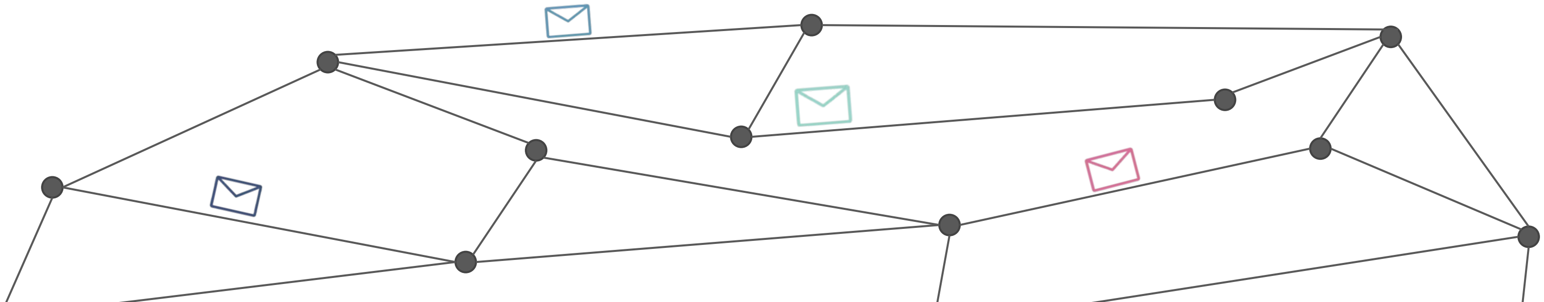
Round complexity:
$\tilde{O}(1) + \tilde{O}(1 + k/\delta)$

Algorithm

1. Launch $\delta(G)$ COBRAs
2. Wait for $\tilde{O}(1)$ rounds
3. Each COBRA results in a spanning subgraph. Carve a spanning tree
4. Distribute and downcast messages

Round complexity:
$\tilde{O}(1) + (\tilde{O}(1 + k/\delta)) \cdot \tilde{O}(1)$

Towards general graphs



Searching for a Lower Bound

$D + k/\lambda$ is not enough!

Searching for a Lower Bound

D + k/λ is not enough!

There exists a graph with $D = O(1)$, $k/\lambda = O(1)$ but OPT = $\Omega(n^{1/2})$

Searching for a Lower Bound

$D + k/\lambda$ is not enough!

There exists a graph with $D = O(1)$, $k/\lambda = O(1)$ but OPT = $\Omega(n^{1/4})$

Thank you!