A simple deterministic distributed MST Algorithm, with Near-Optimal Time and Message Complexities.

Jil Weber
What is a MST?
What is a MST?

MST = minimum spanning tree
How to construct a MST?

At the beginning: all nodes are their own roots
How to construct a MST?

At the beginning: all nodes are their own roots

Fragments
How to construct a MST?
How to construct a MST?
How to construct a MST?
How to construct a MST?
How to construct a MST?
How to construct a MST?
How to construct a MST?
Step 1: Get ID from all neighbours
Step 2: Find blue edge
Step 2: Find blue edge
Step 3: Send Message from root
Step 4: Request message
Step 5.1: $v$ sends also request message
Step 5.1: New root with smaller ID
Step 5.2: v parent of u
Step 5.2: v parent of u
Step 6: root sends ID to all nodes of F5
Time and Message Complexity

→ One phase: Time: $O(n)$  Message: $O(m)$

→ $O(\log(n))$ phases:
   Time: $O(n \cdot \log(n))$
   Message: $O(m \cdot \log(n))$
Preliminaries

• Synchronous CONGEST model
  → send message of size $O(\log(n))$
• Edges have weights and the ID of size $O(\log(n))$
Build base forest

• Construct \((n/k, O(k))\)-MST forest:
The different steps

- $t = \log(k)$ phases
- Phase 0:

The phase $i$ starts: $(n/2^{i-1}, 6 \cdot 2^i)$-MST forest
Phase i:

• For each fragment of diameter at most $2^i$
Phase i:

• For each fragment of diameter at most $2^i$

• Time: $O(2^i)$ and Message: $O(m)$
Phase i:

- Candidate fragment graph:
Phase i:

• Maximal Matching:
Phase i:

• Merge all green fragments:

• We have a \( n/2^i, 6 \cdot 2^{i+1} \) MST forest
Proof

Diameter of each fragment $\leq 6 \cdot 2^{i+1}$:
Algorithm

• Start with \((n/k, O(k))\)- MST forest
  ➔ Base forest with base fragments
• Step 1: Construct a BFS tree with root \(rt\)
  ➔ Time: \(O(D)\) and Message: \(O(m)\)
• \(j\) Phases already calculated
Phase j+1:

- Node $v$ knows:

  Base Fragments: $F_v \in F_0$

Actual Fragment: $F'_v \in F_j$
Phase j+1:

- Step 1: Search minimum weight outgoing edge

- Time: $O(k)$  Message: $O(n)$
Phase j+1:

• Step 2: Send blue edge to the root of the BFS tree

• Time: $O(D + |F_j|)$  
  Message: $O(D \times |F_j|)$
Phase j+1:

• Step 3: Root computes the blue edge e for each actual fragment
Phase j+1:

- Step 4: Inform all roots of the base fragments, which new actual fragment they include

→ Send message \((F_i, F')\)

→ Time: \(O(D + |F_0|)\)
Phase j+1:

• Step 5: All base roots inform their nodes of the new actual fragment

  → Time: $O(1)$  Message: $O(m)$
Convergecast
Send intervals of the base fragment’s root to the root of the tree

→ Time: $O(D + n/k)$
Time Complexity

• Compute blue edge: \( O(k) \)
• Upcast message: \( O(D^+ |F_j|) \)
• Downcast: \( O(D^+ |F_0|) = O(D^+ n/k) \)

\( \rightarrow O(\log(n)) \) phases
\( \rightarrow O( (D + \sqrt{n}) \cdot \log(n) ) \)