Connectivity and MST

last time: memory per machine $M = n^{1+\varepsilon}$

today: $M = n^{\alpha}$ $\alpha \in (0,1)$

$M = \tilde{O}(n)$

$|V| = n$ $|E| = m$

Each node v maintain label $\ell(v)$

$L_v \subseteq V$: set of vertices labeled v

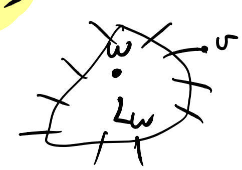
— connected component containing v

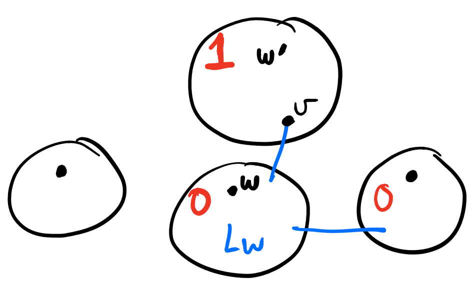
$\Gamma(v)$: neighbors of v

$S \subseteq V$, $\Gamma(S) = \bigcup_{v \in S} \Gamma(v)$

$\Gamma'(v) \triangleq \Gamma(L_v)$

1. Every node $u \in V$ active with label $\ell(u) = u$
2. For $i = 1, 2, 3 \ldots O(\log n)$ do
   (a) Call each active node a leader w. prob. $1/2$
   (b) For every active non-leader w
   find $w^* = \min \{ \ell(v) \mid v \in \Gamma'(w), \ell(v) \text{ leader} \}$
   (c) If w^* not empty, mark w passive
   relabel each node with label w by w^*





Note: Label is not necessarily min. node in component!

Lemma: At any point in algorithm, $\forall u \in V$ there is a path from u to $l(u)$

Pf: By induction on # rounds.

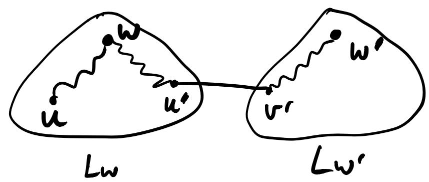
True initially :

Suppose this is true at beginning of round

If $l(u)$ does not change, $\exists$ path from u to $l(u)$

Suppose $l(u) = w$ initially

relabeled to $l(u) = w'$



Corollary: If at any point, 2 nodes s & t have same label then $\exists$ path from s to t in G

Lemma: Every connected component of G has unique label after $O(\log n)$ rounds w.h.p.

Pf Will show that within each connected component, #labels decreases by constant factor in expectation in every round, till every vertex has same label

labels = # connected comp = # active nodes

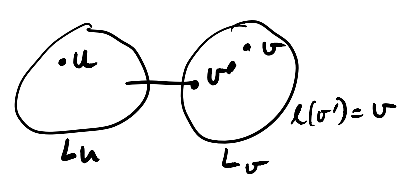
Fix active node u

If component containing u has more than one active node, then

$\exists\ v' \in \Gamma'(u)$ with label different from u

$l(v') = v$

with prob. $\frac{1}{4}$, active node v is selected as leader, u is a non-leader



$\Rightarrow u$ will be relabeled and become passive.

$$\text{Prob.}\left[\text{active node survives a round} \,\middle|\, \exists \text{ more than one label in conn. component}\right] \leq \frac{3}{4}$$

How to implement in MPC?

memory $M = n^{\alpha}$ $\quad \alpha < 1$

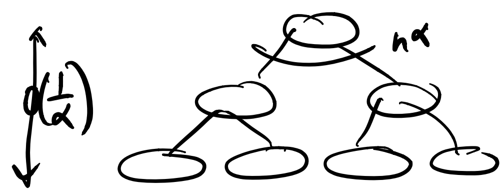
$n^{1-\alpha}$ machines for vertex status : label
active / passive
leader / non-leader

For each active non-leader node w
find $w^* = \min \{ \ell(v) \mid v \in \Gamma'(w),\ \ell(v) \text{ leader} \}$

For each edge $\{u,v\}$, if $\ell(u) \neq \ell(v)$, $\ell(u)$ non-leader
$\ell(v)$ leader

$\ell(v)$ is potential label for $\ell(u)$

For each active non-leader node w
first compute # candidate labels (w. duplicates)



then compute minimum over set of candidate labels.

$O(\frac{1}{\alpha})$ rounds

Broadcast new labels to all nodes in $O(\frac{1}{\alpha})$ rounds

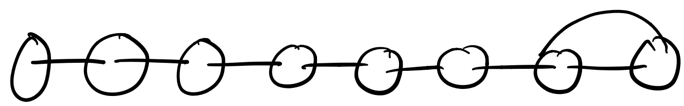
Overall: $O(\frac{\log n}{\alpha})$ rounds in MPC model w. memory n^{α}

MST : Boruvka's algorithm ['26]

Maintain connected components

Repeat $O(\log n)$ times :

Choose cheapest edge out of each current component
& merge components



Problem: merging could take large # rounds!

Fix: use idea of random leaders.

Mark each component leader w. prob $\frac{1}{2}$

Each non-leader chooses cheapest outgoing edges only if it goes to leader component

Cannot have long chain of merges.

Open Problem:

Conjecture: Connectivity requires $\Omega(\log n)$ rounds with memory n^{α} $\alpha < 1$

Hard Case? Distinguish between cycle on n nodes vs. 2 cycles on $\frac{n}{2}$ nodes

$O(1)$ rounds with $\tilde{O}(n)$ memory

Claim: Sorting in $O(\frac{1}{\alpha})$ rounds with n^{α} memory

Idea: Choose n^{α} pivots at random
Sort pivots on one machine
divide into subproblems of size $\sim n^{1-\alpha}$ each
recurse on subproblems in parallel
$O(\frac{1}{\alpha})$ rounds

Sort edges in increasing order
$w(e_1) \leq w(e_2) \ldots \leq w(e_m)$

Kruskal: examines edges in this order

Observation: Edge e_i is in MST iff its endpoints not in the same connected component in graph with $\{e_1 \ldots e_{i-1}\}$

Group edges into chunks of n edges
Each chunk fits on one machine
Process chunks simultaneously

For $i = \{1 \ldots \frac{m}{n}\}$

$E_i = \{e_{(i-1)n+1} \ldots e_{in}\}$ ith chunk

$E_i' = \bigcup_{j=1}^{i} E_i$ union of E_1 to E_i

F_i' : forest of connected components from E_i'

F_0' : all vertices isolated

Any edge $\{u, v\} \in E_i$ is in MST iff
components of u & v are different in

$$F_{i-1}' \cup \{\text{edges preceding } \{u,v\} \text{ in } E_i\}$$

Note: Single machine can hold E_i & F_{i-1}' and make decisions for chunk E_i

Need: Algo to compute connected components in $O(1)$ rounds.

We already have such an algo!
via l_0-sampling sketches
Can be implemented in $O(1)$ rounds of MPC
with $n \cdot \text{polylog}(n)$ memory — enough to store sketches
of all vertices and run connectivity algo on single machine