Simple Concurrent Labeling Algorithms for Connected Components

Sixue (Cliff) Liu
Princeton University

joint work with

Robert E. Tarjan
Princeton University and Intertrust Technologies

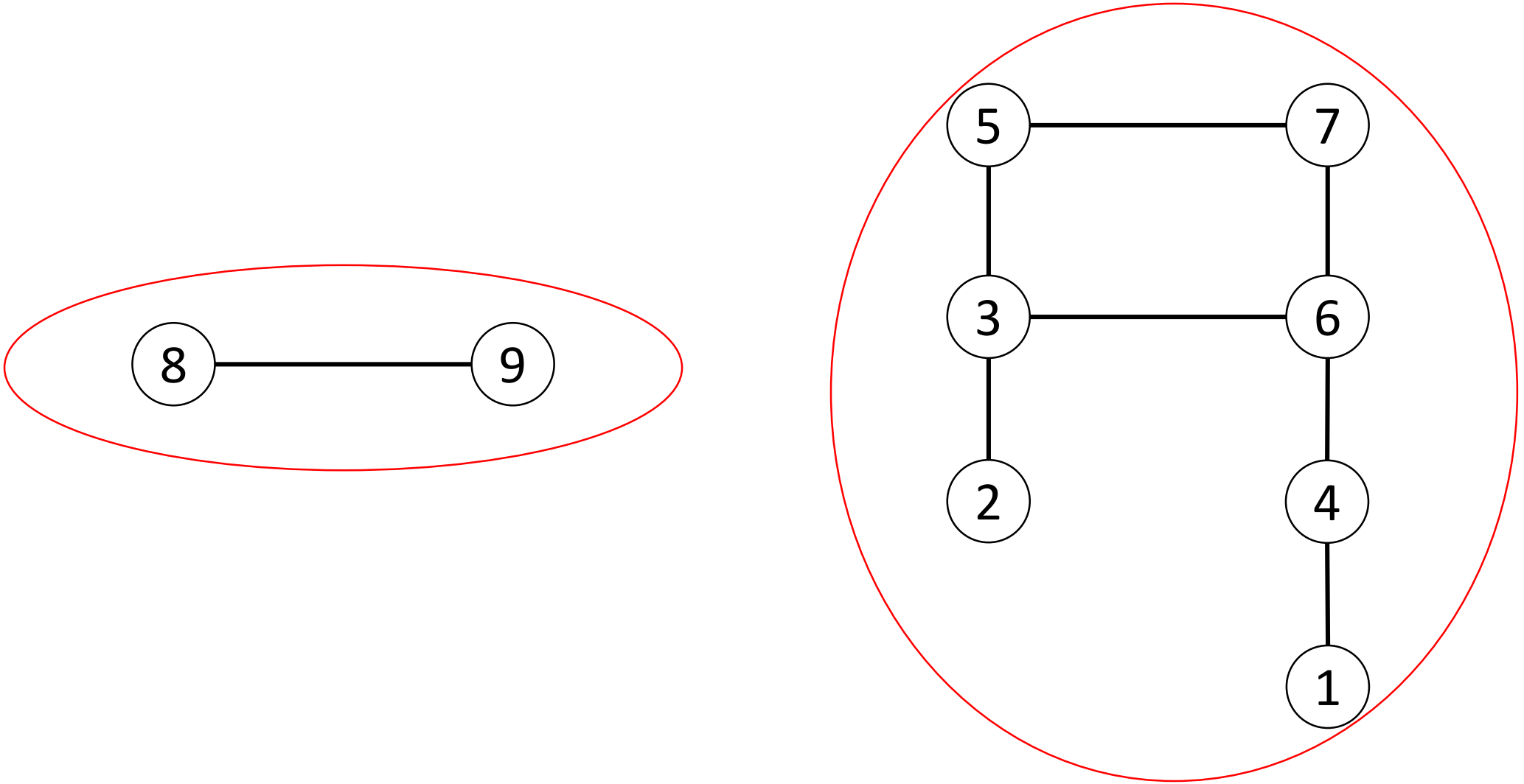
Connected Components

The most basic graph problem?

In an undirected graph, two vertices are connected if there is a path between them. A connected component is a maximal set of pairwise-connected vertices (henceforth just a component).

Problem: Given a graph, compute its components.

Connected Components



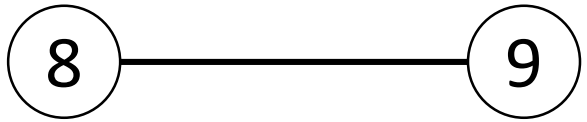
How to represent components?

Label all vertices in each component with a unique vertex in the component: can test if two vertices are in the same component by comparing their labels.

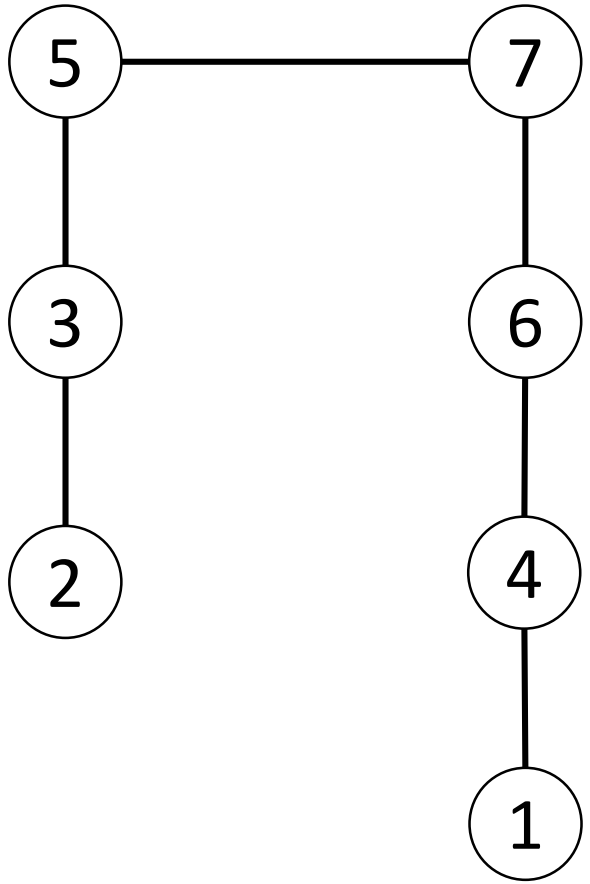
Assume *n* vertices: 1, …, *n*; *m* edges

Minimum labeling: the minimum vertex in the component.

How to represent components?

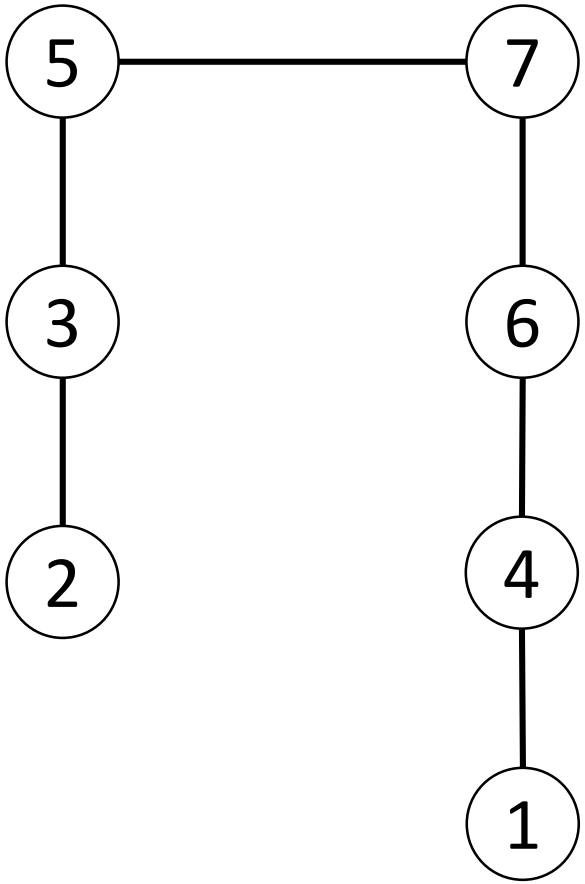


8 8
9 8

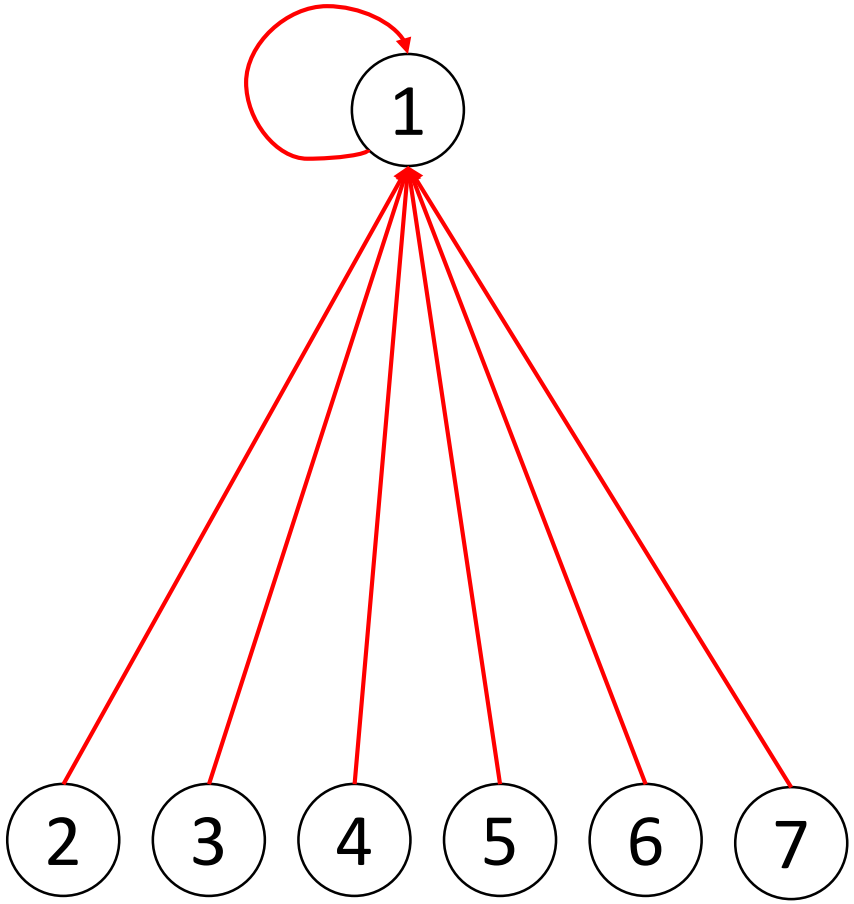


1 1
2 1
3 1
4 1
5 1
6 1
7 1

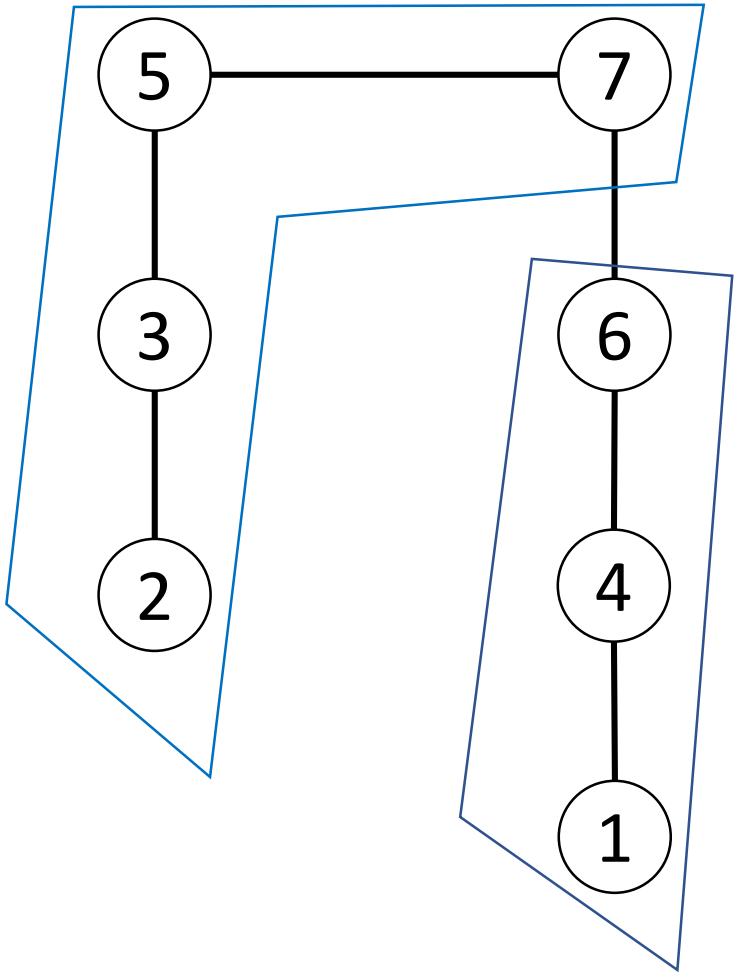
How to represent components?



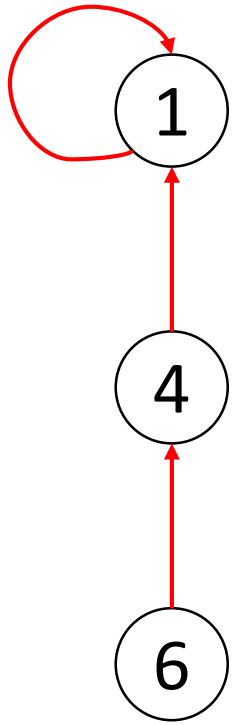
1 1
2 1
3 1
4 1
5 1
6 1
7 1

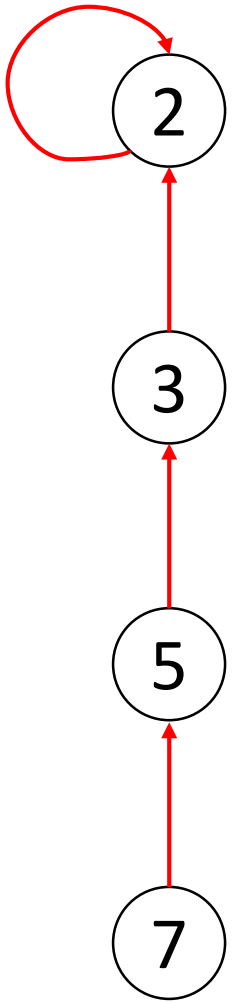


How to represent components: during execution?



1 1
2 2
3 2
4 1
5 3
6 4
7 5





How to represent components?

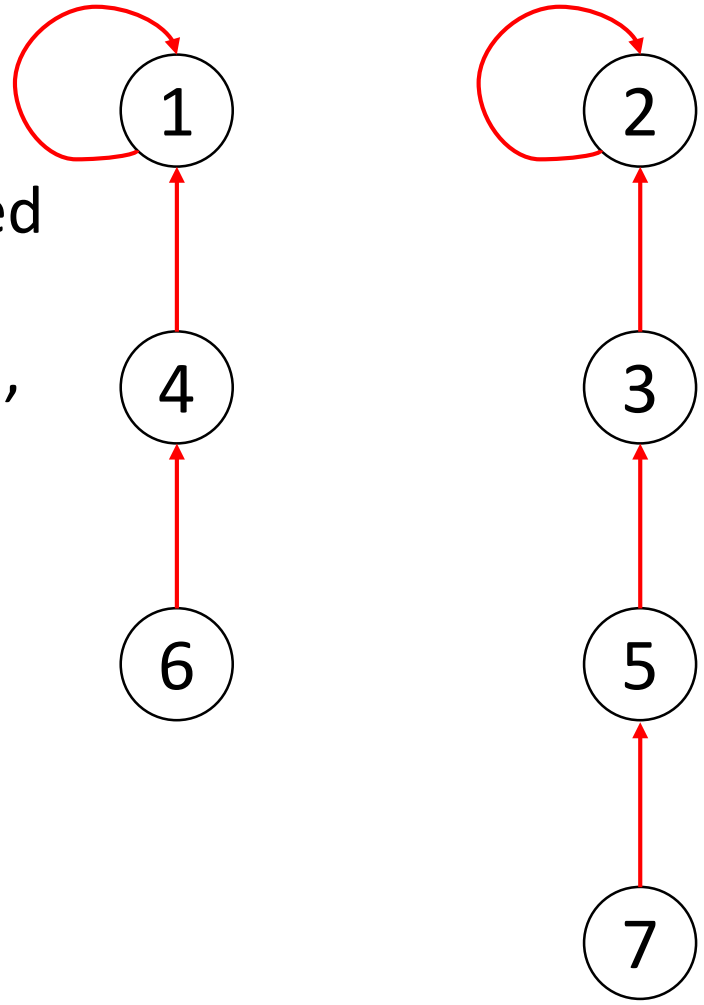
The vertices v and the arcs $(v, v.p)$ define a directed graph (digraph)

If the only cycles are loops (arcs of the form (v, v)), the digraph consists of a set of rooted trees:

v is a root iff $v = v.p$

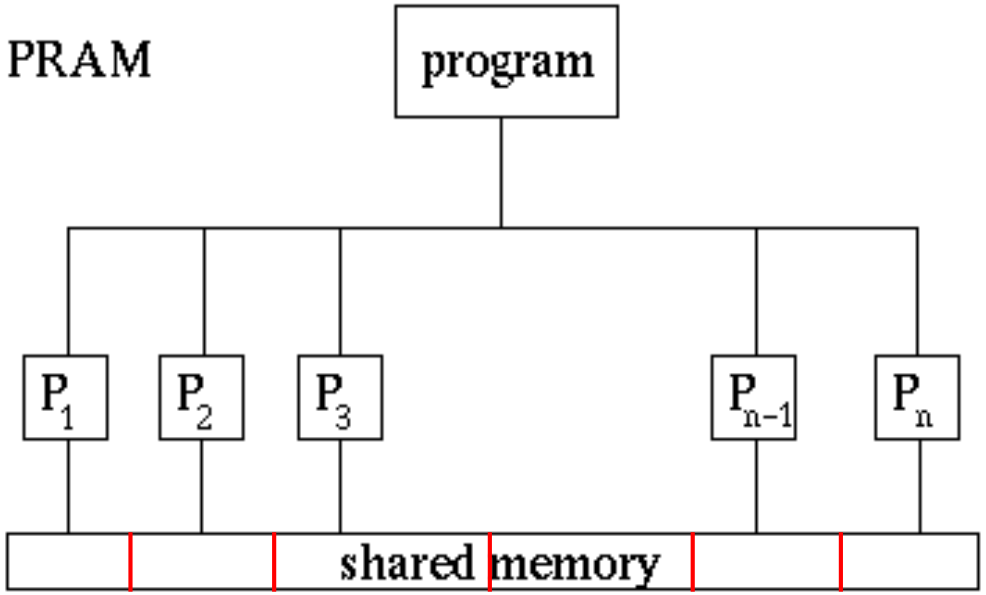
$v.p$ is the parent of v if $v \neq v.p$

If labels never increase, all cycles are loops



PRAM model: O(*m*+*n*) processors

COMBINING CRCW (concurrent read, concurrent write) PRAM in which write conflicts are resolved in favor of the smallest value written



[figure from http://cs.uef.fi/~penttone/parallel/pram.html]

Our results

There are several (or more) buggy analyses of such algorithms in the literature. Our algorithms are both simpler than those in the literature and have correct analyses. Our analytical techniques should also apply to some of the algorithms in the literature.

Our results

Seven extremely simple algorithms:

- Two of them are equivalent and run in $\Theta(\lg^2 n)$ time.
- For another three algorithms, one runs in $O(\lg^2 n)$ time, and two of them also run in $O(d)$ time.
- Another two of them are equivalent and run in $O(\lg n)$ time.

Algorithm R (for root-connect)

for each *v* do *v.p* = *v*;

repeat

{for each (*v, w*) do if *v.p* < *w.p.p* & *w.p.p* = *w.p*

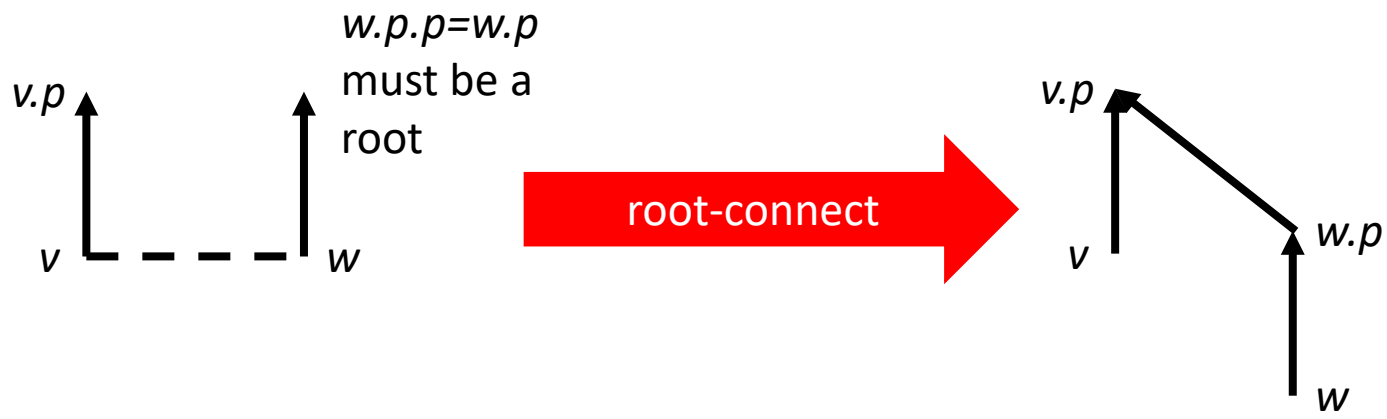
then *w.p.p* = *v.p*;

for each *v* do *v.p* = *v.p.p*}

until no parent changes

Algorithm R (for root-connect)

for each *v* do *v.p* = *v*;
repeat
{for each (*v, w*) do if *v.p* < *w.p.p* & *w.p.p* = *w.p* then *w.p.p* = *v.p*;
for each *v* do *v.p* = *v.p.p*}
until no parent changes



Algorithm R (for root-connect)

for each *v* do *v.p* = *v*;
repeat
{for each (*v, w*) do if *v.p* < *w.p.p* & *w.p.p* = *w.p* then *w.p.p* = *v.p*;
for each *v* do *v.p* = *v.p.p*}
until no parent changes

Surprisingly, R is new (as far as we can tell)

- Shiloach & Vishkin, 1982: Arbitrary CRCW PRAM algorithm, O(lg *n*) steps.
- Awerbuch and Shiloach, 1987: Variant of Shiloach-Vishkin algorithm, simpler, same bounds, simpler analysis
- Reif, 1984: Randomized algorithm, O(lg *n*) steps
- Johnson and Metaxis, 1997: $O(\lg^{3/2} n)$-steps on a CREW PRAM; is monotonic, but does not maintain acyclicity: uses a variant of shortcutting to eliminate any cycles it creates

Our bound

- Algorithm R: $\Theta(\lg n)$ rounds
  - Analysis uses a potential function and a novel multi-round analysis: flat trees can linger for a non-constant number of rounds

Main idea

- Algorithm R runs in O(lg *n*) time.
  - The sum of the heights of certain trees decreases by a constant factor every constant number of rounds.

Analysis of algorithm R

✓After two rounds all trees contain at least two vertices (except in components of one vertex)

- A tree is passive in a round if it does not change during that round, active if it does
- The potential $\Phi(T)$ of an active tree T is its height plus one, plus one more if flat
- The potential of a passive tree is zero

Analysis of algorithm R

- Let T be an active tree at the end of round k
- *The* constituent trees of T at the end of round $j \leq k$ are those at the end of round j whose vertices are in T
- The potential of T in round j is the sum of the potentials of its constituent trees

Lemma: $\Phi(T_{k-1}) \geq \Phi(T)$, and if $k - j \geq 5$, $\Phi(T_j) \geq (4/3)\Phi(T)$

Analysis of algorithm R

- A shortcut reduces the potential by a constant factor
- A calculation gives $\Phi(T_{k-1}) \geq \Phi(T)$, and $\Phi(T_{k-1}) \geq (6/5)\Phi(T)$ if T has height at least 4
- If T has height at most 3 and has at least one active constituent tree of sufficient height, or at least two constituent trees, the lemma holds
- Otherwise T only has one active constituent tree
- After at most four rounds, all constituent trees are combined, and T is passive, a contradiction

Future Work

- Tighten the bounds of our other algorithms.
- Prove the bound $O(\min\{d, \log n\})$ for some of our algorithms or design a simple algorithm to achieve this bound.

A faster algorithm by Andoni et al. runs in time $O(\log d \log\log_{m/n} n)$, but is on the stronger model MPC and it's randomized. [FOCS'18]

✓We already generalized their algorithm to PRAM and achieve the same bound.

- Derandomize this algorithm.

Thanks!