

Faster MST Algorithms

MST problem: Given a weighted graph $G = (V, E, w)$. Let \mathcal{S} be a collection of cycle-free subsets of E closed under inclusion (i.e., of $A \in \mathcal{S}$ and $B \subseteq A$ then also $B \in \mathcal{S}$). Goal is to find the maximal cycle-free subset (i.e., a spanning tree) of largest weight.

\mathcal{S} satisfies the following replacement property:

If $A, B \in \mathcal{S}$ and $|B| = |A| + 1$, then there exists some element $f \in B - A$ such that $A \cup \{f\} \in \mathcal{S}$.

MST can be found by repeatedly applying the red rule.

Red Rule: Pick an arbitrary cycle in the remaining graph and erase the heaviest edge in that cycle.

Forms the basis of the Pipeline Algorithm — $O(n)$ time MST algorithm.

Pipeline Algorithm for MST (Synchronous Model)

0. Build a breadth-first tree B is built on G . Let r be the root of B .

1. Each node v , except the root r , maintains two lists of edges, $Q(v)$ (candidate set) and $U(v)$.

2. A leaf node starts sending edges upwards at pulse 0.

An intermediate node starts sending at the first pulse *after* it has received at least one message from each of its children.

3. Initially, $Q(v)$ contains only the edges adjacent to v , and $U(v)$ is empty.

At each pulse, v sends the minimum-weight edge in $Q(v)$ that does not create a cycle with the edges in $U(v)$ to its parent and moves this edge from $Q(v)$ to $U(v)$.

Any edge that creates a cycle with edges in $U(v)$ is deleted from $Q(v)$.

If $Q(v)$ is empty, v sends a terminate message to its parent. The parent after receiving an edge from a child, adds the edge in its $Q(v)$ list.

4. The root r computes the MST locally from among the edges it hears from its children. The solution is then broadcast over tree B to all nodes.

Analysis

Two observations:

1. The edges reported by each intermediate vertex to its parent in the tree are cycle-free.
2. Every vertex starts sending messages upwards at pulse $L(v) = \text{Depth}(T(v))$.

Consider an intermediate vertex v at height h that has still not terminated its participation in the algorithm, at round t , for some $t \geq h$.

A child is **active** if it has not terminated yet (i.e., it sent an element to v in round $t - 1$).

Lemma

(a) For each child u of v that is still active at round t , $Q(v)$ at the beginning of round t contains at least one edge.

(b) If v sends to its parent an edge of weight w_0 at round t , then all of the elements v was informed at round $t - 1$ by its active children were of weight w_0 or larger.

(c) If v sends to its parent an element of weight w_0 at round t , then any later element it will learn is of weight w_0 or larger.

(d) v sends elements in nondecreasing weight order.

Proof: By induction on the height of the tree.

Base case: trivially holds for leaves.

Consider an intermediate vertex v at height h and assume that the claims hold for each of its children.

Claim (a): Let A_v be the set of m elements sent by v to its parent during the first $t - h$ rounds (i.e., $h \dots, t - 1$).

Consider an active child u of v . Let A_u be the set of elements sent by u to v up to round $t - 1$.

u has transmitted continuously to v , since round $L(u) \leq h - 1$. Hence, $|A_u| \geq m + 1$.

Thus there exists some element $f \in A_u - A_v$ such that $A_v \cup \{f\}$ is cycle-free. This element belongs to $Q(v)$.

Claim (b): Consider any active child u of v . Let f be the element sent by u on round $t - 1$. Let f' be some element sent by u at some round $t' \leq t - 1$ and is still in $Q(v)$ at round t .

$$w(f) \geq w(f') \geq w_0.$$

Claim (c): Follows trivially from claim (b).

Claim (d): Follows from claim (c) and the element selection rule.

Termination Lemma

After a vertex v has terminated its participation in the algorithm, it will learn of no more reportable edges.

Proof: By induction.

Base case: true for leaves.

Induction step: Follows from Claim (a).

Correctness and Analysis

Correctness follows from red rule.

The root starts getting messages at time $Depth(T)$.

The root receives at most $n - 1$ elements from each of its children.

The sends these elements in a fully pipelined fashion.

Time to know all the edges is $O(n - 1 + Depth(T)) = O(n)$.

The root can broadcast the entire set of MST in an additional $O(Depth(T) + n - 1) = O(n)$ time.

The communication complexity is $O(n^2)$ (why?).

$\tilde{O}(D + \sqrt{n})$ time algorithm

1. (Controlled) GHS algorithm: Idea:

Do GHS algorithm till the depth of a fragment reaches \sqrt{n} (called a *terminated* fragment).

A terminated fragment stops merging onto other fragments. However, an active fragment can merge onto a terminated fragment.

At the end of the first part, the size (number of nodes) of a fragment is at least \sqrt{n} , and there are at most \sqrt{n} fragments.

(How can we ensure this in a simple way? Project idea.)

The first part of the algorithm should take $O(\sqrt{n} \log n)$ time.

2. Pipeline algorithm. How to implement this?

Takes $O(D + \sqrt{n})$ time.