Thinning by Random Sampling (1993)

Select half the edges at random.

Build a minimum spanning forest of the sample.

Thin.

How many edges remain?

Karger: $O(n \log n)$ on average

Klein, Tarjan: $< 2n$ on average
Verification:

Given a spanning tree, is it minimum?

Thinning: Given a spanning tree, delete any non-tree edge larger than every edge on tree path joining its ends (red rule).

If all non-tree edges can be thinned, tree is verified.
History of Verification Algorithms

Tarjan, 1979 \(O(m \alpha (m,n))\) time

Komlos, 1984 \(O(m)\) comparisons

Dixon, Rauch, Tarjan, 1992 \(O(m)\) time

King, 1993 \(O(m)\) time (simplified)

All these algorithms will thin.
History

Tarjan (1979) \(O(m \alpha (m,n))\)

path compression

Komlós (1985) \(O(m)\)

Dixon, Rauch, Tarjan (1990) \(O(m)\)

T + K + table look-up

* comparisons only; nonlinear overhead

\[ \alpha (m,n) = \min \{ i \mid A(i,|m/n|) > \log n \} \]

\(A = \text{Ackermann's Function}\)
Minimum Spanning Forest Algorithm

If \(\#\) edges/\(\#\) vertices < 5, then

(Boruvka step) Select the cheapest edge incident to each vertex.

Contract all selected edges.

Recur on contracted graph.

Else

(Sampling and Thinning Step) Sample the edges, each with probability 1/2.

Construct a minimum spanning forest of the sample, recursively.

Thin using this forest.

Recur on Thinned Graph
Bound on Number of Edges Not Thinned

Let $e_1, e_2, ..., e_m$ be the edges, in increasing cost.

Run the following variant of Kruskal's algorithm.

Initialize $F = \emptyset$.

Process the edges in order.

To process $e_i$, flip a coin to see if $e_i$ is in the sample.

If $e_i$ forms a cycle with edges in $F$, discard it as thinned.

Otherwise, if $e_i$ is sampled, add $e_i$ to $F$. (Whether or not $e_i$ is sampled, it is not thinned.)

$F$ is the minimum spanning forest of the sample.
How many edges are not thinned?

The only relevant coin flips are those on unthinned edges, each of which has a chance of $1/2$ of adding an edge to $F$ (a success).

There can be at most $n-1$ successes.

For there to be more than $k$ unthinned edges, the first $k$ relevant coin flips must give at most $n-2$ successes.

The chance of this is at most

$$
\left(\frac{1}{2}\right)^k \sum_{i=0}^{n-2} \binom{k}{i} < \left(\frac{1}{2}\right)^k \sum_{i=0}^n \binom{k}{i}
$$

In particular, the average number of unthinned edges is at most $2n$. 
Analysis

Boruvka step

\[ m < 5n \text{ implies } m' < \frac{9m}{10} \text{ since at least} \]
\[ \frac{n}{2} \text{ edges are contracted} \]
\[ T(m) = O(m) + T\left(\frac{9m}{10}\right) \]

Thinning Step

\[ m > 5n \text{ implies } 2n < \frac{2m}{5} \]
\[ T(m) = O(m) + T\left(\frac{m}{2}\right) + T\left(\frac{2m}{5}\right) \]

where \( T\left(\frac{m}{2}\right) \) and \( T\left(\frac{2m}{5}\right) \) are expected time

\[ T(m) = O(m) \text{ by induction} \]
Preprocessing – Table Lookup

**Idea:** Given enough time (exponential or super-exponential) one can build an optimum algorithm for a given problem in a given computational model, such as a decision tree. (The algorithm itself may be exponential in size.)

This means that sufficiently small (log or log-log size) subproblems can be solved optimally by table lookup using only linear preprocessing time.
Verification

each nontree edge:
cost as large as max on tree path
Overall approach:

Note:

This method can give algorithms optimal to within a constant factor *without* offering a tight estimate of how fast they are.
Further Results

$O(\max(n) \log n) \rightarrow O(\max(n))$ deterministic
chazelle: "soft" heaps

Optimal to within a constant factor

Pettie + Rama Chandran:
chazelle + optimal on small subproblems
Open Problems

Deterministic $O(m)$?

Simpler verification?

Other applications?

directed spanning trees?

shortest paths?