ARCHIVUM MATHEMATICUM (BRNO)
Tomus 40 (2004), 315 – 320

TWO LINEAR TIME ALGORITHMS FOR MST ON MINOR CLOSED GRAPH CLASSES

MARTIN MAREŠ

Abstract. This article presents two simple deterministic algorithms for finding the Minimum Spanning Tree in $O(|V|+|E|)$ time for any non-trivial class of graphs closed on graph minors. This applies in particular to planar graphs and graphs of bounded genus. Both algorithms run on a pointer machine and they require no a priori knowledge of the structure of the class except for its density. Edge weights are only compared.

1. Introduction: The MST problem

The problem of finding a minimum spanning tree of a weighted undirected graph is one of the most well-known algorithmic problems of combinatorial optimization. Since the first solution by Borůvka [1] in 1926 (see [10] for an English translation), a plethora of increasingly more efficient algorithms has been developed (for the full story, see [8] and [4]).

Assuming that edge weights are taken from an arbitrary ordered set (the only operation defined on them is comparison), the current speed record is held by the algorithms of Chazelle [5] and Pettie [11] which achieve time complexity $O(m \cdot \alpha(m,n))$ where n and m are the number of vertices and edges of the graph and $\alpha(m,n)$ is a certain inverse[1] of the Ackermann's function. If the edge weights are integers whose bits can be manipulated in constant time, there exists a MST algorithm by Fredman and Willard [3] running in linear time on a unit-cost RAM. Also, there is a randomized algorithm with expected linear time for the general case due to Karger et al. [6].

Recently, Pettie and Ramachandran [12] have shown an algorithm for the pointer machine with running time bound by the size of the optimum MST decision tree. Since the decision-tree complexity is an obvious lower bound for the algorithmic time complexity of the problem, this algorithm is optimal up to a

2000 *Mathematics Subject Classification*: 05C04.
Key words and phrases: minor closed graph classes, minimum spanning trees.
Partially supported by the Project LN00A056 of the Czech Ministry of Education and by the Forschungsinstitut für Mathematik, ETH Zürich, Switzerland.
Received October 9, 2002.

[1] $\alpha(m,n) = \min\{i \geq 1 : A(i, 4\lceil m/n \rceil) > \log n\}$

multiplicative constant and no random access is needed to achieve optimality. However, the decision-tree complexity of the MST is still unknown and no non-trivial lower bounds are known, hence it is still open whether the MST can be found in linear time or not.

Although the question is still unresolved for general graphs, there are several special cases where linear-time algorithms are known to exist. When the graph is sufficiently dense, meaning that it has at least $n \cdot \log^{(k)} n$ edges[2] for some k, Tarjan's $O(m \cdot \beta(m, n))$ algorithm [13] performs linearly. On the other end of the spectrum, there exist several $O(m + n)$ algorithms for planar graphs (e.g., by Matsui [7]), so the only problematic cases seem to be low density graphs with no special structure which could be taken advantage of.

This article narrows the gap by showing two MST algorithms which run in linear time for any non-trivial class of graphs closed on graph minors. In particular, this includes planar graphs and graphs of bounded genus. We base our time bounds on density of minor closed classes (see for example Nešetřil and De Mendez [9]). Our algorithms do not require any specific knowledge of class structure except for class density needed by the second algorithm. This is a substantial improvement over the previous results for planar graphs which require construction of a planar embedding.

Without loss of generality, we assume that the graph is connected and that all the edge weights are distinct. Also, n and m always denote the number of vertices and edges of the graph under examination.

2. Minor closed classes

For the sake of completeness, we define minor closed graph classes and we also mention the well-known statement about the density of such classes.

Definition 1. Graph H is a *minor* of graph G if it can be obtained from G by a sequence of deletions and contractions of edges and deletions of isolated vertices.

Definition 2. Let $\mathcal{C}$ be a class of graphs. We define its *edge density* $\varrho(\mathcal{C})$ to be the infimum of all ϱ's such that $|E(G)| \leq \varrho \cdot |V(G)|$ holds for any $G \in \mathcal{C}$.

Theorem 1 (see Theorem 6.1 in [9]). *A minor closed class $\mathcal{C}$ has finite edge density iff $\mathcal{C}$ is a non-trivial class, i.e. different from the class of all graphs and the empty class.*

Graphs with finite density not only have a vertex of small degree (as it is well known for planar graphs), but there also has to be a large fraction of such vertices:

Lemma 1 (Density Lemma). *Let $\mathcal{C}$ be a graph class with density ϱ and $G \in \mathcal{C}$ a graph with n vertices. Then at least $n/2$ vertices of G have degree at most 4ϱ.*

Proof. Assume the contrary: let there be at least $n/2$ vertices with degree greater than 4ϱ. Then $\sum_v \deg(v) > n/2 \cdot 4\varrho = 2\varrho n$ which is in contradiction with the number of edges being $\leq \varrho n$. (The proof can also be viewed probabilistically:

[2] $\log^{(k)}$ denotes binary logarithm iterated k times

let X be degree of a vertex of G chosen uniformly at random. Then $\mathbf{E}X \le 2\varrho$, hence by the Markov's inequality $\Pr[X > 4\varrho] < 1/2$, so for at least $n/2$ vertices $\deg(v) \le 4\varrho$.)

For planar graphs, the bound can be easily tightened:

Lemma 2 (Density Lemma for Planar Graphs)**.** *Let G be a planar graph with n vertices. Then at least $n/2$ vertices of v have degree at most 8.*

Proof. It suffices to show that the lemma holds for triangulations (if there are any edges missing, the situation can only get better) with at least 3 vertices. Since G is planar, $\sum_v \deg(v) < 6n$. The numbers $d(v) := \deg(v) - 3$ are non-negative and $\sum_v d(v) < 3n$, hence by the same argument as in the previous proof, for at least $n/2$ vertices v it holds that $d(v) < 6$, hence $\deg(v) \le 8$.

3. A meta-algorithm

Our two algorithms are variations on the original algorithm by Borůvka, but instead of growing a forest of MST subtrees by joining them by edges newly proven to be in the MST, we keep each subtree contracted to a single vertex (this approach has been suggested in a less general setting by Tarjan in [13]). Both algorithms can be considered incarnations of the following "meta-algorithm":

1. Start with the input graph.
2. Find some edges which are part of the MST of the current graph.
3. Contract the graph along these edges.
4. Clean up the graph (to be explained in a moment).
5. Repeat steps 2–4 until there are no edges left.

This procedure is obviously correct (due to it stopping after a finite number of contractions and the well-known fact that given any subset A of MST edges, the rest of the MST are just edges of a MST of the same graph with edges of A contracted). However, we need to decide on how do we choose the edges to be contracted and how to represent the graph in order to perform searching for these edges and all the contractions efficiently.

We would like to make use of the low density of non-trivial minor closed classes of graphs, but unfortunately it is not as simple as it looks, because edge contractions produce loops and parallel edges, leaving us with a multigraph which of course can have a superlinear number of edges. Loops are the easy part: they can be easily detected and removed immediately after the contraction. From a set of parallel edges, we can delete all but the lightest one, but the key problem is to avoid spending a lot of time on detecting them.

To accomplish this, we introduce a *cleanup phase* which is called occassionally during the course of the algorithm and whose purpose is to prune the graph (make it again a simple graph):

4.1. Bucket-sort all graph edges lexicographically, bringing parallel edges together.
4.2. Walk the edge list sequentially, delete the unnecessary parallel edges and recalculate vertex degrees.

4.3. Remove zero degree vertices.

The cleanup takes time $O(m+n)$ where m and n are the number of edges and vertices *before* the cleanup took place, so we need to use this fine spice very carefully.

4. Algorithm 1

We follow the meta-algorithm, using the fact that for each vertex, the lightest incident edge belongs to the MST (it follows from the Tarjan's blue rule [13] applied to a cut formed by edges incident to the vertex). No cycles can arise since the edge weights are distinct. Also, we process contractions due to all vertices of the current graph at once and clean up the graph afterwards. This gives:

Algorithm 1.

1. Start with the input graph.
2. Construct a temporary graph G' on the same set of vertices. For each vertex v, G' contains the lightest edge of G incident to v.
3. Contract the graph along the edges of G': find connected components of G' and renumber every vertex of G to the number of the component they belong to.
4. Clean up the graph (as defined above).
5. Repeat steps 2–4 until there are no edges left.

Lemma 3. *For any non-trivial minor closed class of graphs $\mathcal{C}$, Algorithm 1 finds the MST of any graph in this class in time $O(\varrho(\mathcal{C}) \cdot n)$.*

Proof. Correctness follows from the meta-algorithm (we need the properties of the graph class only for the time bound). Each pass of the algorithm takes time $O(m+n)$ and it reduces n at least by a factor of two. All graphs generated by the algorithm (after cleanup) are minors of the input graph, so they belong to $\mathcal{C}$ as well and, according to Theorem 1, $m \le \varrho n$ holds for all of them. Therefore the total time spent by the algorithm is $O(\varrho n + \varrho n/2 + \varrho n/4 + \dots) = O(\varrho n)$.

5. Algorithm 2

Instead of batching the contractions, we can also perform them greedily on the lightest edges adjacent to low-degree vertices and delay the cleanup until we run out of such vertices. This gives our second algorithm (t is a parameter to be chosen later):

Algorithm 2.

1. Start with the input graph.
2. While there exists a vertex v with $\deg(v) \le 4t$, select the lightest edge e incident to v and contract it (just remove all edges incident to v and add them back to the graph with v renumbered to the other end of e). Delete all loops and isolated vertices that arise. To avoid sequential searching, keep a queue of such low-degree vertices.
3. Clean up the graph (as defined above).

4. Repeat steps 2–3 until there are no edges left.

For minor closed classes of graphs, we set t to the density of the class or, if the exact density is unknown, to any upper bound on the density. We get:

Lemma 4. *For any non-trivial minor closed class of graphs $\mathcal{C}$, Algorithm 2 with $t \geq \varrho(\mathcal{C})$ finds the MST of any graph in this class in time $O(tn)$.*

Proof. Let G_i^k denote the graph we work with at the start of step i in the k-th iteration of the algorithm, let n_i^k and m_i^k be its number of vertices and edges respectively. For each k, G_2^k is a simple graph which is a minor of the input graph, so it belongs to $\mathcal{C}$ and according to Theorem 1, $m_2^k \leq \varrho n_2^k$ where $\varrho = \varrho(\mathcal{C})$. Due to Lemma 1, at least $n_2^k/2$ vertices of G_2^k have $\deg(v) \leq 4\varrho \leq 4t$. Because of this, step 2 runs at least once in each iteration, so the algorithm stops after a finite number of iterations and since it is following the meta-algorithm, it is correct.

All executions of step 2 contribute to the total running time by $O(tn)$ — each contraction takes $O(t)$ and it removes at least one vertex. To conclude the proof of time complexity, it suffices to show that the total time spent by all the cleanups is $O(tn)$ as well.

Let's look at the k-th cleanup: it takes $O(n_3^k + m_3^k)$ time units. However, we know that $n_3^k \leq n_2^k$ and $m_3^k \leq m_2^k \leq tn_2^k$, hence we can bound the time by $O(tn_2^k)$.

Also, G_2^k contains at least $n_2^k/2$ vertices of $\deg(v) \leq 4t$ (let D denote the set of them) while all vertices of G_3^k have $\deg(v) > 4t$. So every $v \in D$ had either to disappear by having been merged to another vertex by a contraction or $\deg(v)$ had to increase by another vertex having been merged to v. Considering that each contraction can affect in this way at most two vertices of D, there must have been at least $n_2^k/4$ contractions and as each of them removes at least one vertex, $n_2^{k+1} \leq 3/4 \cdot n_2^k$.

Thus all the cleanups take time $O(tn_2^1 + tn_2^2 + \dots) = O(tn + t(3/4)n + t(3/4)^2 n + \dots) = O(tn)$.

Remark 1. For planar graphs, we can take advantage of having proven Lemma 2 and improve the performance by a constant factor by setting the parameter t to 2.

6. Conclusions

We have presented algorithms for the minimum spanning tree problem which run in deterministic linear time for any non-trivial class of graphs closed on graph minors. This further reduces the classes of graphs where the complexity of finding the MST is unknown, but it still leaves the general version of the problem open.

Closedness on graph minors seems to be crucial for algorithms of this type, low density per se doesn't suffice: there exists a simple linear time reduction (see [2] for details) of general MST to MST on graphs with maximum degree 3.

References

[1] Borůvka, O., *O jistém problému minimálním (About a Certain Minimal Problem)*, Práce mor. přírodověd. spol. v Brně, III, (1926), 37–58. Czech with German summary.

[2] Frederickson, G. N., *Data structures for on-line updating of minimum spanning trees*, SIAM J. Comput. **14** (1985), 781–798.

[3] Fredman, M. and Willard, D. E., *Trans-dichotomous algorithms for minimum spanning trees and shortest paths*, In Proceedings of FOCS'90 (1990), 719–725.

[4] Graham, R. L. and Hell, P., *On the history of the minimum spanning tree problem*, Ann. Hist. Comput. **7** (1985), 43–57.

[5] Chazelle, B., *A Minimum Spanning Tree Algorithm with Inverse-Ackermann Type Complexity*, J. ACM **47** (2000), 1028–1047.

[6] Karger, D. R., Klein, P. N. and Tarjan, R. E., *Linear expected-time algorithms for connectivity problems*, J. ACM **42** (1995), 321–328.

[7] Matsui, T., *The Minimum Spanning tree Problem on a Planar Graph*, Discrete Appl. Math. **58** (1995), 91–94.

[8] Nešetřil, J., *Some remarks on the history of MST-problem*, Arch. Math. (Brno) **33** (1997), 15–22.

[9] Nešetřil, J. and de Mendez, P. O., *Colorings and Homomorphism of Minor Closed Classes*, To appear in Pollack-Goodman Festschrift, Springer Verlag, 2002.

[10] Nešetřil, J., Milková, E. and Nešetřilová, H., *Otakar Borůvka on Minimum Spanning Tree Problem*, Discrete Math. **233**(1–3) (2001), 3–36.

[11] Pettie, S., *Finding minimum spanning trees in $O(m\alpha(m,n))$ time*, Tech Report TR99-23, Univ. of Texas at Austin, 1999.

[12] Pettie, S. and Ramachandran, V., *An Optimal Minimum Spanning Tree Algorithm*, In Proceedings of ICALP'2000, 49–60, Springer Verlag, 2000.

[13] Tarjan, R. E., *Data structures and network algorithms*, 44 CMBS-NSF Regional Conf. Series in Appl. Math. SIAM, 1983.

Department of Applied Mathematics
and Institute for Theoretical Computer Science (ITI)
Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic
E-mail: `mares@kam.mff.cuni.cz`