Heavy and leafy trees*

Cristina G. Fernandes[1], Carla N. Lintzmayer[2], Mário César San Felice[3]

[1]Departamento de Ciência da Computação, Universidade de São Paulo, Brazil

[2]Centro de Matemática, Computação e Cognição, Universidade Federal do ABC, Brazil

[3]Departamento de Computação, Universidade Federal de São Carlos, Brazil

cris@ime.usp.br, carla.negri@ufabc.edu.br, felice@ufscar.br

Abstract. *The* MAXIMUM WEIGHTED LEAF TREE *problem consists of, given a connected graph G and a weight function $w\colon V(G) \to \mathbb{Q}^+$, finding a tree in G whose weight on the leaves is maximized. The variant that requires the tree to be spanning is at least as hard to approximate as the maximum independent set problem. If all weights are unitary, it turns into the well-known problem of finding a spanning tree with maximum number of leaves, which is NP-hard, but is in APX. No further innapproximability result is known for* MAXIMUM WEIGHTED LEAF TREE*, and the best approximation is an $\mathrm{O}(\lg n)$-approximation. In this paper, we present an $\mathrm{O}(\lg W)$-approximation where W is the maximum weight divided by the minimum weight that appears on the vertices.*

1. Introduction

The MAXIMUM WEIGHTED LEAF TREE problem is that of, given a connected (undirected) graph G and a weight function $w\colon V(G) \to \mathbb{Q}^+$, finding a tree T in G such that the weight of the leaves, which are vertices of degree 1, of T is maximized. When $w(v) = 1$ for all $v \in V(G)$, one can observe that such an optimal tree can always be extended to be spanning with the same weight. The variant of the problem with uniform weights and that requires the tree to be spanning is known as the MAXIMUM LEAF SPANNING TREE problem, which is NP-hard [Garey and Johnson 1979] and also APX-hard [Galbiati et al. 1994]. The best known result for it is a 2-approximation algorithm proposed by Solis-Oba [Solis-Oba 1998, Solis-Oba et al. 2017]. The directed version was shown to be NP-hard [Alon et al. 2009] and MaxSNP-hard [Schwartges et al. 2011] on Directed Acyclic Graphs (DAGs). For the directed version there is a 92-approximation algorithm [Daligault and Thomassé 2009] and a $\frac{3}{2}$-approximation for DAGs [Fernandes and Lintzmayer 2021].

Interestingly, if one has a vertex-weighted graph and requires a spanning tree whose weight on leaves is maximum, then the problem becomes at least as hard to approximate as the problem of finding a maximum independent set in a graph [Drescher and Vetta 2010], that is, it is not approximable within $n^{1-\varepsilon}$ for any $\varepsilon > 0$, unless P $=$ NP, where n is the number of vertices in the graph. On the other hand, for the MAXIMUM WEIGHTED LEAF TREE problem, there exists an $\mathrm{O}(\lg n)$-approximation for general graphs [Gandhi et al. 2018].

*C. G. Fernandes was partially supported by CNPq (Proc. 308116/2016-0 and 423833/2018-9). C. N. Lintzmayer was partially supported by CNPq (Proc. 428385/2018-4).

The $O(\lg n)$-approximation presented by Gandhi, Hajiaghayi, Kortsarz, and Purohit [Gandhi et al. 2018] for the MAXIMUM WEIGHTED LEAF TREE problem is in fact for directed graphs, but the authors mention that it is applicable for undirected graphs as well. Their approach starts by reducing the problem on general weights to the MAXIMUM {0,1}-WEIGHTED LEAF TREE problem, when the weights are only 0 or 1. Then, they show that any α-approximation algorithm for the problem when the weights are 0 or 1 is an $(\alpha \lg n)$-approximation for general weights.

Lemma 1.1 ([Gandhi et al. 2018]). *If there is an α-approximation for the* MAXIMUM {0,1}-WEIGHTED LEAF TREE *problem, then there exists an $O(\alpha \lg n)$-approximation for the* MAXIMUM WEIGHTED LEAF TREE *problem (with general weights).*

At last, they show that the MAXIMUM {0,1}-WEIGHTED LEAF TREE problem can be reduced to the MAXIMUM LEAF SPANNING TREE (unweighted) preserving the approximation ratio.

Lemma 1.2 ([Gandhi et al. 2018]). *Given a digraph G with $\{0, 1\}$-vertex weights, we can construct an unweighted digraph G' in polynomial time so that G' has a solution for the* MAXIMUM LEAF SPANNING TREE *problem with at least φ leaves if and only if G has a solution for the* MAXIMUM {0,1}-WEIGHTED LEAF TREE *problem of weight at least φ.*

Since both directed and undirected versions of MAXIMUM LEAF SPANNING TREE have constant-factor approximation algorithms, their $O(\lg n)$-approximation result follows.

Another interesting result given by these authors is a relation between MAXIMUM WEIGHTED LEAF TREE and the CONNECTED MAXIMUM CUT problem. In the latter, one is given a graph G and a weight function $w\colon E(G) \to \mathbb{Q}^+$ and wants to find a set $S \subset V(G)$ such that the weight of edges in the cut $(S, V(G) \setminus S)$ is maximized and $G[S]$ is connected. They showed that any α-approximation algorithm for the MAXIMUM WEIGHTED LEAF TREE problem provides a 4α-approximation algorithm for the CONNECTED MAXIMUM CUT problem.

While there is an $O(\lg n)$-approximation for the MAXIMUM WEIGHTED LEAF TREE problem, to the best of our knowledge, there is no result of inapproximability for it in the literature. Thus, one might wonder whether there is a constant-factor approximation for this problem. Notice that this would also imply a constant-factor approximation for the CONNECTED MAXIMUM CUT problem.

We present some preliminary results we obtained on the MAXIMUM WEIGHTED LEAF TREE problem. In Section 2, we show that, by restricting attention to instances where weights are powers of 2, we lose at most a factor of 2 on the approximation ratio. In Section 3, we present an approximation for this restricted problem. Its ratio depends on the maximum exponent of the powers of 2 that appear as weights on the vertices. As a consequence, we derive a $2(2 + \lg W)$-approximation for MAXIMUM WEIGHTED LEAF TREE, where W is the maximum weight of a vertex divided by the minimum weight of a vertex. This becomes a constant approximation for instances in which W is bounded by a constant.

2. General weights

In this section, we show how approximation algorithms for instances with general weights can be obtained through instances whose weights are only powers of 2.

We start with some notation. For a tree T, let $L(T)$ be the set of leaves of T and $w(T)$ denote $w(L(T)) = \sum_{v \in L(T)} w(v)$. We use $\mathrm{opt}(G, w)$ to denote the cost of an optimal solution for the MAXIMUM WEIGHTED LEAF TREE problem, that is, $w(T^*)$, where T^* is an optimal solution for the instance $\langle G, w\rangle$. We also write $\mathrm{opt}(G)$ to denote the number of leaves in an optimal solution for the MAXIMUM LEAF SPANNING TREE problem, which is the amount of leaves in an optimal solution for G.

Let $\langle G, w\rangle$ be an instance of the MAXIMUM WEIGHTED LEAF TREE problem. Let $w_{\min} := \min\{w(v) : v \in V(G)\}$. We create a new function w' by setting $w'(v)$ as the highest power of 2 which is smaller than or equal to $w(v)/w_{\min}$.

Suppose there is an α-approximation algorithm for the MAXIMUM WEIGHTED LEAF TREE problem when the weights are powers of 2 and let T be the solution produced by this algorithm for $\langle G, w'\rangle$. If T^* is an optimal solution for $\langle G, w\rangle$, then note that

$$w(T) \geq w_{\min} w'(T) \geq w_{\min} \frac{\mathrm{opt}(G, w')}{\alpha} \geq w_{\min} \frac{w'(T^*)}{\alpha} > \frac{w(T^*)}{2\alpha} = \frac{\mathrm{opt}(G, w)}{2\alpha}.$$

Thus any α-approximation for instances where the weights are integers and powers of 2 yields a 2α-approximation for general weights.

3. An approximation for weights that are powers of 2

We now restrict our attention to weight functions $w\colon V(G) \to \{1, 2, 4, \ldots, 2^k\}$ for some $k \geq 0$, and show a $(k + 2)$-approximation in this case.

Theorem 3.1. *There is a $(k+2)$-approximation algorithm for the* MAXIMUM WEIGHTED LEAF TREE *problem when the weights of the vertices are in $\{1, 2, 4, \ldots, 2^k\}$, for $k \geq 0$.*

Proof. The result follows by induction on k. If $k = 0$, then return the tree produced by the 2-approximation algorithm for the MAXIMUM LEAF SPANNING TREE problem over G [Solis-Oba 1998, Solis-Oba et al. 2017].

Now let $k \geq 1$ and denote by $\ell_{2^k}(T)$ the amount of leaves of weight 2^k of any tree T. Build $w'\colon V(G) \to \{0, 1\}$ such that $w'(v) = 1$ if $w(v) = 2^k$ and $w'(v) = 0$ otherwise. Let T' be the tree returned by the 2-approximation algorithm for the MAXIMUM {0,1}-WEIGHTED LEAF TREE problem over $\langle G, w'\rangle$ (Lemma 1.2). Build $w''\colon V(G) \to \{1, 2, \ldots, 2^{k-1}\}$ such that $w''(v) = 2^{k-1}$ if $w(v) = 2^k$ and $w''(v) = w(v)$ otherwise. Let T'' be the tree returned by the $(k-1+2)$-approximation algorithm existing by induction. The algorithm returns the tree with cost $\max\{w(T'), w(T'')\}$.

Note that, by definition, $w(T) = w''(T) + 2^{k-1}\ell_{2^k}(T)$. Let T^* be an optimal solution for $\langle G, w\rangle$. Next, we analyze the relations between this term $2^{k-1}\ell_{2^k}(T)$ and a fraction of the cost $w(T^*)$.

If $2^{k-1}\ell_{2^k}(T^*) \geq \frac{w(T^*)}{k+2} = \frac{1}{k+2}\mathrm{opt}(G, w)$, then the tree T' is such that

$$w(T') \;\geq\; 2^k w'(T') \;\geq\; 2^k \frac{\mathrm{opt}(G, w')}{2} \;\geq\; 2^{k-1} w'(T^*) \;=\; 2^{k-1}\ell_{2^k}(T^*) \;\geq\; \frac{1}{k+2}\,\mathrm{opt}(G, w)\,,$$

where we used the fact that $w'(T) = \ell_{2^k}(T)$ for any tree T.

Otherwise, if $2^{k-1}\ell_{2^k}(T^*) < \frac{w(T^*)}{k+2} = \frac{1}{k+2}\mathrm{opt}(G,w)$, then the tree T'' is such that

$$\begin{aligned} w(T'') \;\geq\; w''(T'') \;\geq\; \frac{\mathrm{opt}(G,w'')}{k-1+2} \;\geq\; \frac{w''(T^*)}{k+1} \;=\; \frac{w(T^*) - 2^{k-1}\ell_{2^k}(T^*)}{k+1} \\ >\; \frac{w(T^*) - \frac{w(T^*)}{k+2}}{k+1} \;=\; \frac{(k+2)w(T^*) - w(T^*)}{(k+1)(k+2)} \;=\; \frac{w(T^*)}{k+2} \;=\; \frac{1}{k+2}\,\mathrm{opt}(G,w)\,, \end{aligned}$$

where we used the fact that $w(T) = w''(T) + 2^{k-1}\ell_{2^k}(T)$ for any tree T. □

Corollary 3.2. *There exists a* $2(2+\lg W)$*-approximation algorithm for the* MAXIMUM WEIGHTED LEAF TREE *problem on general weights, where* W *is the maximum weight of a vertex divided by the minimum weight of a vertex in the input.*

Proof. It follows from Theorem 3.1 and from the result on Section 2. □

4. Remarks

The $\mathrm{O}(\lg n)$-approximation from the literature also reduces the problem to MAXIMUM {0,1}-WEIGHTED LEAF TREE instances: it partitions the vertices of the graph by dividing the range of weights into $\mathrm{O}(\log n)$ intervals and, for each interval, it creates an instance with weight 1 only on the vertices with weight in this interval. Our algorithm rounds the weights to powers of two and, for each such power, creates an instance with weight 1 on the vertices with weight greater than or equal to this power. We are analyzing the similarities and differences between the two algorithms, trying to improve on them.

References

Alon, N., Fomin, F., Gutin, G., Krivelevich, M., and Saurabh, S. (2009). Spanning directed trees with many leaves. *SIAM Journal on Discrete Mathematics*, 23(1):466–476.

Daligault, J. and Thomassé, S. (2009). On Finding Directed Trees with Many Leaves. In *International Workshop on Parameterized and Exact Computation*, volume 5917 of *Lecture Notes in Computer Science*, pages 86–97.

Drescher, M. and Vetta, A. (2010). An Approximation Algorithm for the Maximum Leaf Spanning Arborescence Problem. *ACM Transactions on Algorithms*, 6(3).

Fernandes, C. G. and Lintzmayer, C. N. (2021). Leafy spanning arborescences in DAGs. *Discrete Applied Mathematics*. In press. Available also at `https://arxiv.org/abs/2007.07660`.

Galbiati, G., Maffioli, F., and Morzenti, A. (1994). A short note on the approximability of the maximum leaves spanning tree problem. *Information Processing Letters*, 52(1):45–49.

Gandhi, R., Hajiaghayi, M. T., Kortsarz, G., Purohit, M., and Sarpatwar, K. (2018). On maximum leaf trees and connections to connected maximum cut problems. *Information Processing Letters*, 129:31–34.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability*. W.H. Freeman and Co., New York.

Schwartges, N., Spoerhase, J., and Wolff, A. (2011). Approximation algorithms for the Maximum Leaf Spanning Tree Problem on acyclic digraphs. In *9th Workshop on Approximation and Online Algorithms (WAOA)*, volume 7164 of *Lecture Notes in Computer Science*, pages 77–88.

Solis-Oba, R. (1998). 2-approximation algorithm for finding a spanning tree with maximum number of leaves. In *Proceedings of the European Symposium on Algorithms (ESA)*, volume 1461 of *Lecture Notes in Computer Science*, pages 441–452.

Solis-Oba, R., Bonsma, P., and Lowski, S. (2017). A 2-Approximation Algorithm for Finding a Spanning Tree with Maximum Number of Leaves. *Algorithmica*, 77:374–388.