ELSEVIER Information Processing Letters 52 (1994) 45–49

Information Processing Letters

A short note on the approximability of the Maximum Leaves Spanning Tree Problem

G. Galbiati [a,*], F. Maffioli [b], A. Morzenti [b]

[a] *Dipartimento di Informatica e Sistemistica, Università degli Studi di Pavia, Italy*
[b] *Dipartimento di Elettronica, Politecnico di Milano, Italy*

Communicated by F.B. Schneider; received 14 February 1994; revised 18 April 1994

Abstract

We prove that the NP-hard problem of finding in an undirected graph G a spanning tree with a maximum number of leaves is MAX-SNP hard. On the basis of a recent result in the theory of approximability of NP-hard optimization problems stating that all problems that are MAX-SNP hard with respect to approximation-preserving reductions do not allow polynomial time approximation schemes, unless P = NP, we conclude that the Maximum Leaves Spanning Tree Problem does not have a polynomial time approximation scheme, unless P = NP, giving therefore a negative answer to this question, which was left open in [6].

Keywords: Analysis of algorithms; Combinatorial problems; Spanning trees

1. Introduction

Since the discovery in the 70's of NP-hard optimization problems which do not allow efficient solution algorithms, unless P = NP, the search for efficient approximation algorithms having different performance guarantees has begun. Different classes of optimization problems having approximation algorithms with similar performance guarantees have been found and classes like FPTAS, PTAS and APX have been defined long ago as the classes consisting of optimization problems having *fully polynomial time approximation schemes, polynomial time approximation schemes and polynomial time k-approximation algorithms* for some constant $k > 1$, respectively [5,7].

Recently two new classes have been defined by Papadimitriou and Yannakakis [8] on the basis of a characterization of languages in NP in terms of logical formulas, namely the classes MAX-SNP and MAX-NP, both contained in APX. In the same paper [8] problems complete for MAX-SNP under a particular type of reduction which preserves approximability within a constant factor, called L-reduction, have been described, so that a complete problem has a polynomial time approximation scheme iff the whole class does.

More recently Arora et al. [1], on the basis of a characterization of NP in terms of multi-prover interactive proofs [2], have proved that no problem complete for MAX-SNP can have a polyno-

* Corresponding author. Present address: 38 Einstein Drive, Princeton, NJ 08540, USA.

mial time approximation scheme, unless P = NP. As a consequence no problem which is MAX-SNP hard with respect to L-reductions can be in PTAS, unless P = NP.

In this note we describe an L-reduction from the Minimum Dominating Set-B Problem, which is MAX-SNP complete [8], to the Maximum Leaves Spanning Tree Problem, thus proving that Maximum Leaves Spanning Tree is MAX-SNP hard.

As a consequence we are able to give a negative answer to the question, left open in [6], of whether a polynomial time approximation scheme for Maximum Leaves Spanning Tree exists.

In [6] the Maximum Leaves Spanning Tree Problem has been extensively studied. Besides applications in circuit layouts and in communications networks other interesting applications are mentioned, like the one that uses a spanning tree with a maximum number of leaves in the alternative solution, described in [9], of the problem of self-stabilizing a set of processors in spite of distributed control [4]. Moreover in [6], a series of approximation algorithms for the Maximum Leaves Spanning Tree Problem is described based on local search, and the first two in this series are shown to have performance ratios of 5 and 3. It was left open whether the series describes a polynomial time approximation scheme.

The consequence of the hardness result shown in this paper is that this series does not form such a scheme because no polynomial time approximation scheme can exist for this problem, unless P = NP.

In conclusion only polynomial time k-approximation algorithms for some constant $k > 1$ can be found for the Maximum Leaves Spanning Tree Problem, like those described in [6], with $k = 5$ and $k = 3$. Improving the value of k is an open research topic.

2. Definitions

In the following we will describe an L-reduction between two NP-hard optimization problems. We give here two relevant definitions.

The first definition is that of an NP Optimization Problem (NPO for short). Initially it was given by Johnson [7]. The following formulation, very similar, is due to Crescenzi and Panconesi [3].

Definition. An *NPO* problem P is a tuple $(\mathcal{I}, Sol, c, opt)$ where:

(1) $\mathcal{I}$ is the set of the instances of P and it is recognizable in polynomial time.

(2) For each instance I, $Sol(I)$ is the set of feasible solutions of I. The set must be recognizable in polynomial time and a polynomial p exists such that $\forall s \in Sol(I)$, $|s| \leqslant p(|I|)$.

(3) For each instance I and each feasible solution $s \in Sol(I)$, $c(I, s)$ is an integer, non negative function, called objective function.

(4) $opt \in \{max, min\}$ tells if problem P is a maximization or a minimization problem.

Solving problem P on a given instance I means finding a feasible solution $s \in Sol(I)$ which maximizes $c(I, s)$ over all feasible solutions $s \in Sol(I)$ if $opt = max$ or minimizes such value if $opt = min$. $Opt(I)$ indicates such optimum value of the objective function.

The second definition describes what a linear reduction is.

Definition. Given two *NPO* problems

$P = (\mathcal{I}_P, Sol_P, c_P, opt_P)$,

$Q = (\mathcal{I}_Q, Sol_Q, c_Q, opt_Q)$,

an *L-reduction* from P to Q is a tuple $(t_1, t_2, \alpha, \beta)$, where t_1 and t_2 are polynomial time computable functions and α and β are

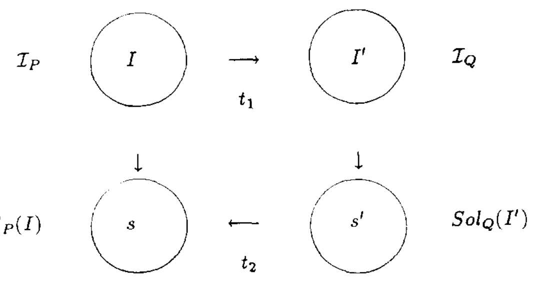

Fig. 1.

positive constants having the following properties (see Fig. 1):

(1) $t_1 : \mathscr{I}_P \to \mathscr{I}_Q$.

(2) $\forall I \in \mathscr{I}_P$ and $\forall s' \in Sol_Q(I')$, with $I' = t_1(I)$, $t_2(I, s') \in Sol_P(I)$.

(3) $\forall I \in \mathscr{I}_P$, $I' = t_1(I)$, $opt_Q(I') \leqslant \alpha opt_P(I)$.

(4) $\forall s' \in Sol_Q(I')$, $I' = t_1(I)$,

$$|opt_P(I) - c_P(I, t_2(I, s'))| \leqslant \beta\,|opt_Q(I') - c_Q(I', s')|.$$

Notice that (1) and (2) define a polynomial time transformation from optimization problem P to optimization problem Q, that is, a fast way to identify from an instance I of P, an instance I' of Q and vice-versa from a solution of I', a solution of I. Instead (3) and (4) characterize the transformation as being an L-reduction.

3. Definitions of the problems

MINIMUM BOUNDED DOMINATING SET (MIN DOMINATING SET-B)

Instance: an undirected graph $G = (V, A)$ with $V = \{v_1, \ldots, v_n\}$ and degrees bounded by a constant B.
Feasible solution: a dominating set W, that is, a set $W \subseteq V$ such that $\forall v \in V - W$ there exists a vertex $v \in W$ such that arc $\{v, w\} \in A$.
Objective function: the cardinality of W.
Opt: *min*.

MAXIMUM LEAVES SPANNING TREE

Instance: an undirected graph $G' = (V', A')$.
Feasible solution: a spanning tree T' of G'.
Objective function: the number of leaves of T'.
Opt: *max*

4. The L-reduction

We now define the tuple $(t_1, t_2, \alpha, \beta)$ which constitutes an L-reduction from MIN DOMINATING SET-B, called problem P, to MAXIMUM LEAVES SPANNING TREE, called problem Q.

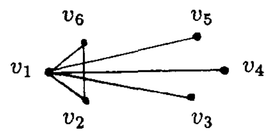

Fig. 2.

The function $t_1 : \mathscr{I}_P \to \mathscr{I}_Q$ is defined as follows. If I is an instance of problem P the instance $I' = t_1(I)$ of problem Q is the graph $G' = (V', A')$, where

$$V' = \{\alpha_0, \alpha_1, 1_1, \ldots, 1_n, 2_1, \ldots, 2_n, 3_1, \ldots, 3_n\},$$
$$A' = \{\{\alpha_0, \alpha_1\}\} \cup \bigcup_{j=1}^{n} \{\alpha_0, 1_j\} \cup \bigcup_{j=1}^{n} \{1_j, 2_j\} \cup \bigcup_{j=1}^{n} \{2_j, 3_j\} \cup \{\{1_h, 2_k\} : \{v_h, v_k\} \in A\}.$$

Figs. 2 and 3 give an example of instances I and $t_1(I)$.

We now define function t_2. If T' is a feasible solution of I' we define $t_2(I, T')$ as the set of vertices $\{v_j$: 1_j is not a leaf in $T'\}$ of G. In order to show that $t_2(I, T')$ is a dominating set of G we use the following helpful construction. We transform tree T' into another spanning tree T'', by connecting to α_0 every vertex 1_j not already adjacent to it and by deleting the unique arc $\{2_i, 1_j\}$ in the resulting cycle. Figs. 4 and 5 illustrate the construction on a spanning tree T' of the graph G' of Fig. 3. Notice that the new tree T'' has the same leaves as T', that is all vertices $\{3_j\}_{j=1,\ldots,n}$, α_1 and some $\{1_j\}$. In fact a leaf 1_j in

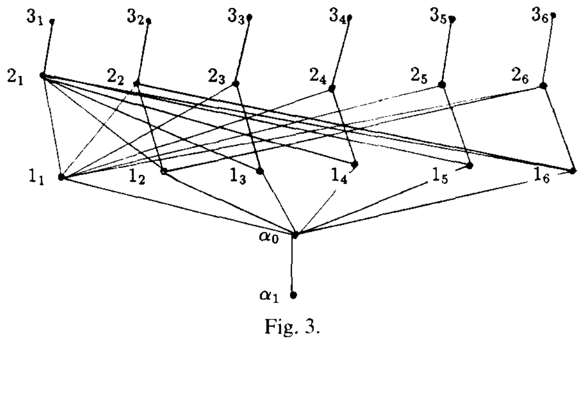

Fig. 3.

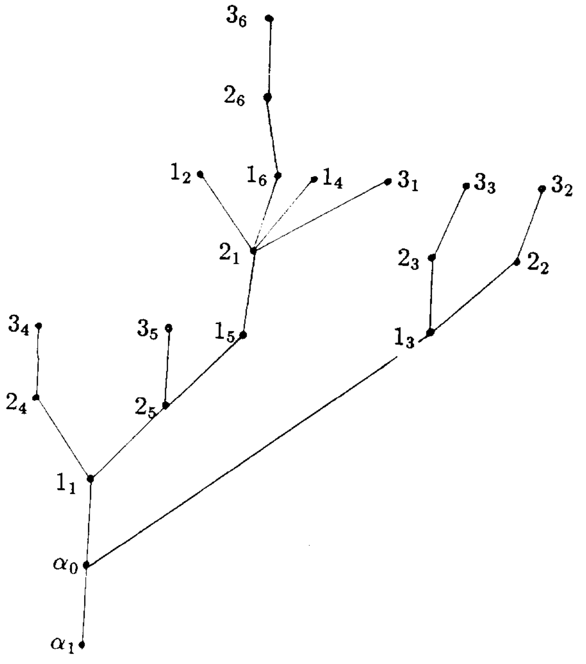

Fig. 4.

T' remains a leaf in T'', an internal node 1_j in T' remains an internal node in T'' and no vertex 2_i, for any i, can be a leaf because of the presence of the adjacent vertex 3_i. Now it is straightforward to see that the set $\{v_j$: 1_j is not a leaf in $T'\}$, invariant under the construction, identify a dominating set of G, and also a partition of the vertices of G into sets made of vertices adjacent to the same vertex of the dominating set.

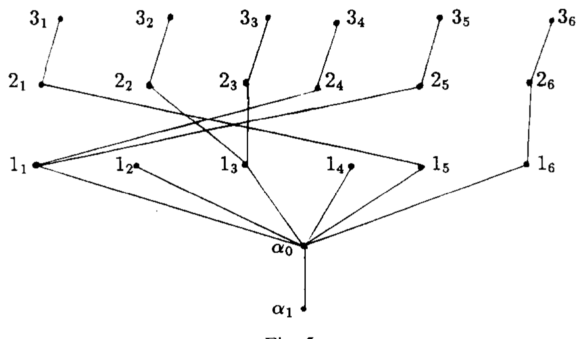

Fig. 5.

Since both t_1 and t_2 can be computed in polynomial time it remains to be proved that the reduction is an L-reduction. We will use the following proposition.

Proposition 1. *If $opt(I) = n_0$ then $opt(I') = 2n - n_0 + 1$.*

Proof. If $opt(I) = n_0$ we construct a spanning tree T' of G' with a number of leaves equal to $2n - n_0 + 1$. Simply we let tree T' consists of all arcs connecting the vertices $\{2_j\}$ to the $\{3_j\}$ and to those vertices $\{1_j\}$ corresponding to the dominating set of cardinality n_0. Then we connect all vertices $\{1_j\}$ to α_0, and α_0 to α_1. Therefore $opt(I') \geqslant 2n - n_0 + 1$. Now if there exists a tree T' with more than $2n - n_0 + 1$ leaves, the tree would have more than $n - n_0$ leaves labelled 1_j and therefore it would identify a dominating set with less than n_0 vertices, contrary to the fact that $opt(I) = n_0$. □

Now we prove the two properties of an L-reduction.

Proposition 2. *There exists $\alpha > 0$ such that $\forall I$, $opt(I') \leqslant \alpha opt(I)$.*

Proof. Using Proposition 1 all we need to prove is that there exists $\alpha > 0$ such that $2n - n_0 + 1 \leqslant \alpha n_0$. This inequality follows if we let $\alpha = 2B + 2$, for instance, and we notice that $n_0(B + 1) \geqslant n$, since graph G is B-bounded. □

Proposition 3. *There exists $\beta > 0$ such that for all spanning trees T' of G'*

$$|opt(I) - c(I, t_2(I, T'))| \leqslant \beta |opt(I') - c(I', T')|.$$

Proof. Let l be the number of leaves of tree T' and let d be the number of vertices $\{1_j\}$ which are not leaves. Using Proposition 1 we can write the preceding inequality as

$$d - n_0 \leqslant \beta(2n - n_0 + 1 - l).$$

Since the leaves of T' are all the $\{3_j\}$, α_1 and some $\{1_j\}$, we have that $l = 2n + 1 - d$ and the conclusion follows with $\beta = 1$. □

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, Proof verification and hardness of approximation problems, in: *Proc. 33rd Ann. IEEE Symp. on Foundations of Computer Science* (1992) 14–23.

[2] S. Arora and S. Safra, Probabilistic checking of proofs: A new characterization of NP, in: *Proc. 33rd Ann. IEEE Symp. on Foundations of Computer Science* (1992) 2–13.

[3] P. Crescenzi and A. Panconesi, Completeness in approximation classes, *Inform. and Comput.* **93** (1991) 241–262.

[4] E.W. Dijkstra, Self-stabilizing systems in spite of distributed control, *Comm. ACM* **17** (1974) 643–644.

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, New York, 1979).

[6] Hsueh-I Lu and R. Ravi, The power of local optimization: Approximation algorithms for Maximum-Leaf Spanning Tree, in: *Proc. Allerton Conf.* (1992) 533–542.

[7] D.S. Johnson, Approximation algorithms for combinatorial problems, *J. Comput. System Sci.* **9** (1974) 256–278.

[8] C.H. Papadimitriou and M. Yannakakis, Optimization, approximation, and complexity classes, *J. Comput. System Sci.* **43** (1991) 425–440.

[9] M. Tchuente, Sur l'auto-stabilisation dans un réseau d'odinateurs, *RAIRO Inform. Théor.* **15** (1981) 47–66.