

CAHIER DU LAMSADE

162

Juin 1999

Master-slave strategy and polynomial approximation

Laurent Alfandari, Vangelis Th. Paschos

Table of Contents

Résumé	ii
Abstract	ii
1 Introduction	1
2 Master-slave tractable problems	2
3 Master-slave tractable network-design problems	5
3.1 Minimum spanning tree of depth 2	5
3.2 Minimum bounded-diameter spanning forest	7
3.3 Edge-covering by trees	9
4 Discussion: introducing a new kind of reduction	11
References	12
A Pseudo-reduction and maximization packing problems	14
A.1 Dual master-slave approximation	14
A.2 Pseudo-reduction and improved approximations for unweighted graph-packing problems	15

Stratégie maître-esclave et approximation polynomiale

Résumé

De nombreux problèmes de minimisation consistant à couvrir les sommets ou les arêtes d'un graphe par des sous-graphes sont classiquement modélisés par un problème de set covering. Si le nombre de sous-graphes est polynomial en n , ces problèmes peuvent alors être approchés à rapport logarithmique par l'algorithme glouton standard pour le set covering. Nous étendons la classe des problèmes approximables par cette approche à des problèmes de couverture et de partitionnement où le nombre de sous-graphes peut être exponentiel en n , en revisitant une technique appelée « maître-esclave » et en l'étendant à des problèmes pondérés. Nous appliquons finalement l'approche maître-esclave à deux problèmes de dimensionnement de réseau et un problème de conception de circuits électroniques afin de produire des résultats positifs d'approximation pour ces problèmes.

Mots-clé : problèmes combinatoires, complexité, algorithme polynomial d'approximation, NP-complétude, couverture d'ensembles, dimensionnement des réseaux, graphe, arbre, forêt.

Master-slave strategy and polynomial approximations

Abstract

A lot of minimization covering problems on graphs consist in covering vertices or edges by subgraphs verifying a certain property. These problems can be seen as particular cases of set-covering. If the number of subgraphs is polynomial in the order n of the input-graph, then these problems can be approximated within logarithmic ratio by the classical greedy set-covering algorithm. We extend the class of problems approximable by this approach to covering problems where the number of candidate subgraphs is exponential in n , by revisiting an old technique called “master-slave” and extending it to weighted master or/and slave problems. Finally, we use the master-slave tool to prove some positive approximation results for two network-design and a VLSI-design graph-problems.

Keywords: combinatorial problems, computational complexity, polynomial-time approximation algorithm, NP-completeness, set-covering, network design, graph, tree, forest.

1 Introduction

Given an **NP**-hard optimization problem Π , one is interested in finding a polynomial time approximation algorithm (PTAA) **A** with a good performance ratio, this ratio being usually defined as the worst, over all instances of a given size, of the values of the fraction “measure of the solution returned by algorithm **A** over the measure of an optimal solution”.

Let us restrict ourselves to **NP**-hard minimization covering graph-problems consisting in covering vertices or edges by subgraphs verifying a certain property. Some of these problems can be approximated by the following thought process: at each step, one tries to cover a maximum number of elements (vertices or edges, depending on the definition of the problem) among the uncovered vertices or edges. The maximization problem solved at each step is called “the slave” which serves “the master”, the (original) minimization problem. The terms “slave” and “master” are due to Simon ([13]) who points out the fact that if the slave-problem is polynomial then the master one is approximable within $O(\log n)$. Then it uses this fact in order to prove that if some master problems are approximate-equivalent, so are the corresponding slave ones (two problems are said approximate-equivalent if they are linked by approximation-preserving reductions, i.e., reductions along which approximation bounds and inapproximability results are transferred from one to the other).

A classical example of master-slave approximation is the one given by Johnson (in [8], algorithm D3 at the end of section 7 devoted to graph-coloring). At each iteration this algorithm computes a maximum independent set of the surviving graph, it colors its vertices by a new color and removes them from the graph. The master problem in this case is the minimum graph-coloring, while the slave one is the maximum independent set (for reasons of economy we do not define here well-known **NP**-complete problems such as graph-coloring, independent set, set-covering, set-packing, dominating set, hitting set, etc.; their definitions are given in [6]). The drawback of D3 is exactly that it is not polynomial, since it uses a maximum independent set computation. In any case, it achieves approximation ratio $O(\ln n)$ at worst.

Here, we first extend the master-slave strategy to **NP**-hard weighted covering and partitioning problems. But our main purpose is to add a contribution towards a systematic classification of **NP**-complete problems with respect to their approximability and, also, to bring to the fore a unified way to obtain approximation results for a certain class of problems for which achievement of logarithmic ratios is not obvious. For this, we define the class **M-ST** of master-slave tractable problems, approximable, in polynomial time, within logarithmic ratio. Informally speaking, given a particular unweighted **NP**-hard graph-covering problem Π , we try to show that Π can be modeled as a set-covering problem, no matter if the size of the obtained set-covering instance is exponential in the size of the generic instance of Π . The set-covering instance obtained includes as ground set the set of objects (vertices or edges) to be covered, and as set-system all the subgraphs able to be included into a feasible solution. Via this modeling, Π can be solved by the following kind of greedy iterative procedure **G**: at each iteration try to cover the maximum number of the uncovered objects. If one can model the problem at hand in terms of set-covering, and if one can prove that covering the maximum number of uncovered objects can be performed in polynomial time (even if the explicit construction of the set-covering instance is exponential), then one has proved that Π belongs to the class **M-ST**. Since **G** is a kind of “simulation” of the greedy set-covering algorithm, one can conclude that Π , as well as every **M-ST** problem, is approximable within approximation ratio equal to the one of the greedy set-covering algorithm; this ratio is $O(\log n)$ ([14]). In other words, instead of developing and analyzing a proper approximation algorithm for every graph-covering problem, we rather try to prove that it satisfies the conditions of inclusion in **M-ST** (let us note that proof of the inclusion of a problem in class **M-ST** is not trivial at all). Then, approximability of Π within logarithmic ratio ensues immediately. Using this

method, we study three natural network-design problems and we prove them **M-ST**. Once again, we note that achievement of logarithmic approximation ratios for them is, to our knowledge, new and seems non-trivial. At the end of the paper, we try to inscribe the master-slave game into the formal framework of a new kind of reduction, by means of which we show that, in some way, set covering is complete for the class **M-ST**. Finally, let us note that inclusion in class **M-ST** is interesting, not only for proving logarithmic ratios, but also for proving lower approximability-bounds (inapproximability results). Plainly, as it is proved in [1], two of the problems studied are not approximable within $(1 - \epsilon) \ln n$, $\forall \epsilon > 0$. Inclusion of them in **M-ST** constitutes a proof that master-slave approximation is the best one can do in order to approximately solve these problems.

In what follows, given an instance I of an **NP**-hard problem Π and a PTAA A for Π , we denote by $OPT(I)$ the optimal value of I , by $A(I)$ the value of the solution of I provided by A , and we say that algorithm A approximates Π within ratio ρ if $A(I)/OPT(I) \leq \rho$, for every instance I of Π .

2 Master-slave tractable problems

We denote by $\mathcal{S}_{\mathcal{P}}$ the set of subgraphs of a graph $G = (V, E)$ satisfying a property or a predicate $\mathcal{P}$, by $V(G)$ (resp., $E(G)$) the vertex-set (resp., edge-set) of G , by n its order ($|V|$) and define the following class.

Definition 1. Consider an **NP**-hard minimization graph-problem Π and a property $\mathcal{P}$; suppose that (i) a feasible solution of Π is an element of $2^{\mathcal{S}_{\mathcal{P}}}$ and (ii) a cost function $\bar{c}$, computable in polynomial time, is associated with $S \in \mathcal{S}_{\mathcal{P}}$ such that the cost of a solution $\mathcal{S}'$ of Π is $c(\mathcal{S}') = \sum_{S \in \mathcal{S}'} \bar{c}(S)$. Then Π is *master-slave tractable* (**M-ST**) iff it satisfies the following conditions:

[M-ST-1] a solution $\mathcal{S}'$ of Π is

- either a $\mathcal{P}$ -cover (i.e., a subset $\mathcal{S}' \subset \mathcal{S}_{\mathcal{P}}$ such that $\cup_{S \in \mathcal{S}'} V(S) = V$),
- or a $\mathcal{P}$ -partition (i.e., a subset $\mathcal{S}' \subset \mathcal{S}_{\mathcal{P}}$ such that $\cup_{S \in \mathcal{S}'} V(S) = V$ and for $S, S' \in \mathcal{S}'$, $V(S) \cap V(S') = \emptyset$), and every $\mathcal{P}$ -cover of G can be polynomially transformed into a $\mathcal{P}$ -partition of at most the same cost;

[M-ST-2] given a binary vector $\vec{u} \in \{0, 1\}^n$ associated with V , the (slave) problem of finding the subgraph $S^* = \operatorname{argmax}_{S \in \mathcal{S}_{\mathcal{P}}} \{\sum_{v \in V(S)} u[v]/\bar{c}(S)\}$ is in **P**.

For the case of edge-cover or partition, we simply replace v by e and V by E . ■

The above definition generalizes the classical master-slave method ([13]) where, $\forall S \in \mathcal{S}_{\mathcal{P}}$, $\bar{c}(S) = 1$, i.e., where $c(\mathcal{S}') = |\mathcal{S}'|$. Note that in the case of partitioning, if property $\mathcal{P}$ is hereditary (i.e., every subgraph of S satisfies $\mathcal{P}$ whenever S satisfies $\mathcal{P}$), then condition [M-ST-1] of definition 1 is always satisfied. However, any partitioning problem does not satisfy condition [M-ST-1]; for instance, from a set-cover one cannot systematically obtain a set-partition of the same weight, or cardinality.

We now show that **M-ST** problems are approximable within logarithmic ratio by the following “simulation” of the classical greedy weighted set-covering (WSC) algorithm.

BEGIN /*MSGREEDY*/

- (1) FOR $v \in V$ DO $u[v] \leftarrow 1$ OD
- (2) $\mathcal{S}' \leftarrow \emptyset$;
- (3) WHILE $\exists v \in V : u[v] = 1$ DO

```

(4)       $S^* \leftarrow \operatorname{argmax}_{S \in \mathcal{S}_{\mathcal{P}}} \{\sum_{v \in V(S)} u[v] / \bar{c}(S)\};$ 
(5)       $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{S^*\};$ 
(6)      FOR  $v \in V(S^*)$  DO  $u[v] \leftarrow 0$  OD
(7)      OD
(8)      OUTPUT  $\mathcal{S}'$ ; (*OUTPUT  $h(\mathcal{S}')$ ;*)
END. /*MSGREEDY*/

```

In the above algorithm, $u[v]$ indicates whether vertex v is uncovered ($u[v] = 1$) or not ($u[v] = 0$) at the current step of the **WHILE** loop. Function h at line (8) polynomially transforms a $\mathcal{P}$ -cover into a $\mathcal{P}$ -partition (when dealing with a partitioning-problem). For edge-covering or partitioning problems, simply replace v (resp., V) by e (resp., E).

Theorem 1. *If Π is **M-ST**, then **MSGREEDY** polynomially approximates Π within $\min\{1 + \ln \Delta_{\Pi}, \ln n - \ln \ln n + 0.78\}$ if the costs of every subgraph satisfying $\mathcal{P}$ are all identical, and within $1 + \ln \Delta_{\Pi}$ otherwise, with $\Delta_{\Pi} = \max_{S \in \mathcal{S}_{\mathcal{P}}} \{|S|\}$.*

Proof. We prove the theorem for vertex-covering and partitioning problems, the proof being quite similar in case of edge-covering or partitioning. Let us transform an instance $G = (V, E)$ of Π into an instance $\varphi(G) = (C, \bar{\mathcal{S}})$ of WSC in the following way. Let $C = V$ be the ground element set in $\varphi(G)$, and for every subgraph $S \in \mathcal{S}_{\mathcal{P}}$, add a set $\bar{S} = V(S)$ with weight $w(\bar{S}) = \bar{c}(S)$ in $\bar{\mathcal{S}}$ (note that the number of sets $|\bar{\mathcal{S}}|$ can be exponential in n). Under this transformation, there is a 1-1 correspondence between the solutions of Π on G and the solutions of WSC on $\varphi(G)$ constructed as above, i.e., $\{S_1, S_2, \dots, S_t\}$ is a $\mathcal{P}$ -cover for G iff $\{\bar{S}_1, \bar{S}_2, \dots, \bar{S}_t\}$ is a set-cover for $\varphi(G)$ and, moreover the cost of the $\mathcal{P}$ -cover of G is the same as the total weight of the set-cover of $\varphi(G)$. Hence, $\operatorname{OPT}(G) = \operatorname{OPT}(\varphi(G))$. Now, the greedy algorithm for the WSC-instance $(C, \bar{\mathcal{S}})$ can be re-written in the following way.

```

BEGIN /*WSCGREEDY*/
  FOR  $c \in C$  DO  $u[c] \leftarrow 1$  OD
   $\bar{\mathcal{S}}' \leftarrow \emptyset$ ;
  WHILE  $\exists c \in C : u[c] = 1$  DO
     $\bar{S}^* \leftarrow \operatorname{argmax}_{\bar{S} \in \bar{\mathcal{S}}} \{\sum_{c \in \bar{S}} u[c] / w(\bar{S})\};$ 
     $\bar{\mathcal{S}}' \leftarrow \bar{\mathcal{S}}' \cup \{\bar{S}^*\};$ 
    FOR  $c \in \bar{S}^* \cap C$  DO  $u[c] \leftarrow 0$  OD
  OD
  OUTPUT  $\bar{\mathcal{S}}'$ ;
END. /*WSCGREEDY*/

```

Since subgraph $S \in \mathcal{S}_{\mathcal{P}}$ (resp., vertex-set V) in G corresponds to set $\bar{S}$ (resp., ground set C) in $\varphi(G)$, then solution $\mathcal{S}' = \{S_1, S_2, \dots, S_t\}$ computed by algorithm **MSGREEDY** for G corresponds to the cover $\bar{\mathcal{S}}' = \{\bar{S}_1, \bar{S}_2, \dots, \bar{S}_t\}$ for $\varphi(G)$, and $c(\mathcal{S}') = w(\bar{\mathcal{S}}')$. Hence, $c(\mathcal{S}') / \operatorname{OPT}(G) = w(\bar{\mathcal{S}}') / \operatorname{OPT}(\varphi(G))$.

If a feasible solution for Π is not a $\mathcal{P}$ -cover but a $\mathcal{P}$ -partition, then by condition [M-ST-1] $\mathcal{P}$ -cover $\mathcal{S}'$ can be transformed in polynomial time into a $\mathcal{P}$ -partition $\mathcal{S}''$ satisfying $c(\mathcal{S}'') \leq c(\mathcal{S}')$.

The approximation ratio of **WSCGREEDY** is bounded above by $1 + \ln \Delta_{\text{SC}}$, where $\Delta_{\text{SC}} = \max_{\bar{S} \in \bar{\mathcal{S}}} \{|\bar{S}|\}$ in the weighted case ([4]), and by $\ln |C| - \ln \ln |C| + 0.78$ in the unweighted case ([14]). Since in $\varphi(G)$, $\Delta_{\text{SC}} = \Delta_{\Pi}$ and $|C| = n$, the proof is completed. ■

Let us remark here that all problems linked to set-covering by approximation-preserving reductions as, for example, the dominating set, the hitting set, a version of the coloring problem where the input graph has stability number bounded above by a fixed positive constant, etc., are **M-ST** ones. Algorithm D3 of [8] mentioned above is indeed a simulation of **WSCGREEDY** for

coloring, but coloring is (unfortunately) not **M-ST** since the (independent set) slave problem is not in **P**.

Simon, in [13], mentions that if the (unweighted) slave-problem is not solvable in polynomial time, but is approximable within a factor $\rho \leq 1$, then the (unweighted) master can be polynomially approximated with a ratio $(1/\rho) \ln n$. Of course ρ can depend on an instance-parameter. Note that a “naïve” application of this result to graph-coloring, where at each round one colors a maximal independent set instead of a maximum one, does not improve the best approximation ratio for coloring. Plainly, the best ratio – $(\log n)^2/n$ ([3]) – for the maximum independent set implies a ratio $n/\log n$ for coloring. This ratio, obtained by Johnson ([9]) in 1974, has been repeatedly improved since then. We can extend Simon’s result to the weighted case where the costs of the subgraphs in $\mathcal{S}_P$ are not constrained to be identical. Let us first transform definition 1 to include a larger class of minimization problems.

Definition 2. A minimization graph-problem Π is *extended master-slave tractable* (**EM-ST**) iff it satisfies condition [M-ST-1] of definition 1 and if, given a binary vector $\vec{u} \in \{0, 1\}^n$ associated with V , the problem of finding the subgraph $S^* = \max_{S \in \mathcal{S}_P} \{\sum_{v \in V(S)} u[v]/\bar{c}(S)\}$ is polynomially approximable within ratio $\rho \leq 1$. ■

Theorem 2. An **EM-ST** problem Π is polynomially approximable within $(1/\rho)(1 + \ln \Delta_\Pi)$.

Proof. Consider the transformation of an instance $G = (V, E)$ of Π into a set-covering instance $(C, \bar{\mathcal{S}})$ as shown in the proof of theorem 1; now, denote by WSCGREEDY^o the version of WSCGREEDY where, at each iteration, instead of the set maximizing the ratio $\sum_{c \in \bar{\mathcal{S}}} u[c]/w(\bar{\mathcal{S}})$, a set, the associated ratio of which is at least ρ times the maximum one ($\rho \leq 1$) is chosen, and revisit the analysis of WSCGREEDY presented in [4]. Suppose, without loss of generality, that the solution computed by WSCGREEDY^o is, after r iterations, the set $\{\bar{\mathcal{S}}_1, \bar{\mathcal{S}}_2, \dots, \bar{\mathcal{S}}_r\}$. Now let $u_j^r = \sum_{c \in \bar{\mathcal{S}}_j} u[c]$ at step r of WSCGREEDY^o , $m = |\bar{\mathcal{S}}|$ and $w_j = w(\bar{\mathcal{S}}_j)$; let $x^* = (x_j^*)_{j=1, \dots, n}$ denote the incidence vector of an optimal cover and let s_j be the largest superscript r such that $u_j^r > 0$. The same analysis as Chvatal’s leads to the following assertion:

$$\text{WSCGREEDY}^o(C, \bar{\mathcal{S}}) \leq \sum_{j=1}^m \left(\sum_{r=1}^{s_j} (u_j^r - u_j^{r+1}) \left(\frac{w_r}{u_j^r} \right) \right) x_j^*.$$

Now, set $\bar{\mathcal{S}}_r$ selected at step r satisfies $u_j^r/w_r \geq \rho(u_j^r/w_j)$, $\forall j$, then

$$\text{WSCGREEDY}^o(C, \bar{\mathcal{S}}) \leq \frac{1}{\rho} \sum_{j=1}^m \left(\sum_{r=1}^{s_j} \frac{u_j^r - u_j^{r+1}}{u_j^r} \right) w_j x_j^*.$$

In [4] it is shown that $\sum_{r=1}^{s_j} (u_j^r - u_j^{r+1})/u_j^r \leq H(|\bar{\mathcal{S}}_j|)$, where $H(k) = \sum_{i=1}^k (1/i)$. Consequently,

$$\text{WSCGREEDY}^o(C, \bar{\mathcal{S}}) \leq \frac{1}{\rho} \sum_{j=1}^m H(|\bar{\mathcal{S}}_j|) w_j x_j^* \leq \frac{1}{\rho} H(\Delta_{\text{SC}}) \text{OPT}(C, \bar{\mathcal{S}}).$$

As $H(\Delta_{\text{SC}}) \approx 1 + \ln \Delta_{\text{SC}}$, we get the ratio of the claim for WSC and transfer this ratio to problem Π thanks to the 1-1 correspondence between solutions of Π and solutions of $(C, \bar{\mathcal{S}})$, as shown in the proof of theorem 1. ■

In the next section, we use the result of theorem 1 to study the approximation of some network-design problems consisting of covering or partitionning vertices or edges by subgraphs, the number of candidate subgraphs being exponential in the size of the input-graph.

3 Master-slave tractable network-design problems

3.1 Minimum spanning tree of depth 2

Consider the following communication problem. Given a set V of cities and an extra city r , find a subset $V' \subset V$ (where one wishes to construct relay stations), such that every city in V' is connected to r and every city in $V \setminus V'$ is connected to a city of V' , and such that the total length of connections is minimum. Formally, the problem is the following.

Minimum-weight rooted spanning tree of depth 2 (RST2).

Given a complete graph $G = (V \cup \{r\}, E)$ with positive integer distances $\vec{d}$ on its edges, find a tree T spanning $V \cup \{r\}$, minimizing quantity $\sum_{e \in E(T)} d(e)$, and such that, for any vertex $v \in V$, the number of edges in T in the path connecting v to r is at most 2.

Recall that a star $S_{v,X}$ is a tree spanning $v \cup X$ such that all vertices of X have degree 1 in the tree. Let v be the “center” of the star and call the quantity $|X|$ the “degree” of the star. We claim that RST2 is equivalent to the following problem.

Minimum-weight spanning star-forest.

Given a complete graph $K_{|V|}$, a cost-vector $\vec{d}$ on edges and a weight vector $\vec{w}$ on vertices, find a spanning forest of stars, i.e., a collection of stars $F = \{S_{v_1, X_1}, S_{v_2, X_2}, \dots, S_{v_t, X_t}\}$ partitioning V and minimizing $c(F) = \sum_{i=1}^t [w(v_i) + \sum_{x \in X_i} d(v_i, x)]$.

In fact, given $G = K_{|V \cup \{r\}|}$, one can consider the (complete) subgraph of G induced by V and set $w(v) = d(vr)$, $v \in V$. In the sequel, when speaking for RST2, we will refer to the latter problem. The proof of the **NP**-completeness of RST2 as well as further positive and negative approximation results about it can be found in [2]. Finally, Note that the total number of stars in a complete graph of order n is $n2^{n-1}$, exponential in n .

Theorem 3. *RST2 is M-ST; consequently, it is approximable within $(1 + \ln n)$.*

Proof. We first prove condition [M-ST-1]. Consider a star-cover $F = \{S_{v_j, X_j} : j = 1, \dots, |F|\}$ and the following procedure.

```

BEGIN /*POSTPROCESS(F)*/
  let  $V_c = \{v_j : j = 1, \dots, |F|\}$ ;
  FOR  $j \leftarrow 1$  TO  $|F|$  DO
    consider  $S_{v_j, X_j}$ ;
    IF  $\exists x \in X_j, (x \in V_c) \text{ OR } (x \in X_i, i < j)$  THEN  $S_{v_j, X_j} \leftarrow S_{v_j, (X_j \setminus \{x\})}$  FI
    IF  $\exists i < j, v_i = v_j$  THEN  $F \leftarrow (F \setminus \{S_{v_i, X_i}, S_{v_j, X_j}\}) \cup \{S_{v_i, (X_i \cup X_j)}\}$  FI
  OD
  OUTPUT  $F$ ;
END. /*POSTPROCESS(F)*/

```

The above procedure is a kind of post-processing transforming a star-cover to a star-partition of at most the same cost. Star-adjustment operations performed here consist of edge-removals or star-groupings which do not increase the cost of the final solution obtained. So, the second part of condition [M-ST-1] in definition 1 is satisfied.

Let us denote by $\bar{c}(S_{v,X}) = w(v) + \sum_{x \in X} d(v, x)$ the cost function associated with star $S_{v,X}$. We now prove that, given a binary vector $\vec{u}$ on $|V|$ components, the slave problem of finding star

$$S_{v^*, X^*} = \operatorname{argmax}_{S_{v,X}} \left\{ \frac{u[v] + \sum_{x \in X} u[x]}{\bar{c}(S_{v,X})} \right\} = \operatorname{argmin}_{S_{v,X}} \left\{ \frac{w(v) + \sum_{x \in X} d(v, x)}{u[v] + \sum_{x \in X} u[x]} \right\}$$

is polynomial (condition [M-ST-2]). Let us denote by $R(S_{v,X}; \vec{u})$ the last of the above ratios.

Observe first that if a vertex $x \in X^*$ satisfies $u[x] = 0$, then removing x from X^* does not change the value of $R(S_{v^*,X^*}; \vec{u})$; so, suppose without loss of generality, that $u[x] = 1, \forall x \in X^*$. Also suppose that the vertices in V are labelled by $v_i, i = 1, \dots, n$, and consider the following procedure finding, given $\vec{u}$, star $S_{v^*,X^*} = \operatorname{argmin}_{S_{v,X}} \{R(S_{v,X}; \vec{u})\}$.

```

BEGIN /*SLAVE_RST2( $\vec{u}$ )*/
  min_ratio  $\leftarrow \infty$ ;
  FOR i  $\leftarrow 1$  TO n DO
     $V_i \leftarrow \{v \in V \setminus \{v_i\} : u[v] = 1\}$ ;
    FOR j  $\leftarrow |V_i|$  DOWNT0 1 DO
      let  $X_j = \{v'_1, v'_2, \dots, v'_j\}$  be the list of vertices of  $V_i$ 
        sorted in such a way that  $d(v_i, v'_1) \leq d(v_i, v'_2) \leq \dots \leq d(v_i, v'_j)$ ;
    OD
    j  $\leftarrow 0$ ;
     $X_0 \leftarrow \emptyset$ ;
    WHILE  $R(S_{v_i, X_{j+1}}; \vec{u}) < R(S_{v_i, X_j}; \vec{u})$  AND  $j < |V_i|$  DO j  $\leftarrow j + 1$  OD
    IF  $R(S_{v_i, X_j}; \vec{u}) < \text{min\_ratio}$  THEN
      min_ratio  $\leftarrow R(S_{v_i, X_j}; \vec{u})$ ;
      SOL  $\leftarrow S_{v_i, X_j}$ ;
    FI
  OD
  OUTPUT SOL;
END; /*SLAVE_RST2( $\vec{u}$ )*/

```

In order to prove that procedure **SLAVE_RST2** correctly finds in polynomial time the star minimizing ratio R over all stars of G , we first prove that, for a fixed v_i , it correctly finds in polynomial time the star minimizing R over all stars with center v_i . Then the correctness-result follows immediately since all vertices are examined as possible star-centers. For reasons of simplicity, we restrict ourselves to the case where $u[v_i] = 1$, the proof for the case $u[v_i] = 0$ being completely analogous. Consider first the problem of determining, for a fixed $|X| = j, j = 0, 1, \dots, |V_i|$, the quantity $\min_X \{(w(v_i) + \sum_{x \in X} d(v_i, x)) / (|X| + 1)\}$. Since the list $(v'_1, v'_2, \dots, v'_{|V_i|})$ is sorted in increasing order with respect to the distances of its elements from v_i , we get (setting $v'_0 = v_i$):

$$\begin{aligned}
\min_{|X|=j} \{R(S_{v_i, X}; \vec{u})\} &= \frac{w(v_i) + \sum_{k=0}^j d(v_i, v'_k)}{j+1} \\
\min_X \left\{ \frac{w(v_i) + \sum_{x \in X} d(v_i, x)}{(|X| + 1)} \right\} &= \min_{j=0, \dots, |V_i|} \left\{ \frac{w(v_i) + \sum_{k=0}^j d(v_i, v'_k)}{j+1} \right\}.
\end{aligned}$$

Let $U_j = R(S_{v_i, X_j}; \vec{u}) = (w(v_i) + S_j) / (j+1)$ with $S_j = \sum_{k=1}^j d(v_i, v'_k)$. Procedure **SLAVE_RST2** increments j while U_j decreases in j and stops as soon as U_j increases. In order to complete the proof, we now prove that when U_j starts increasing, it will never decrease again, i.e., if $U_{j+1} - U_j \geq 0$, then $U_{j+2} - U_{j+1} \geq 0$. Let us study the sign of $U_{j+1} - U_j$.

$$\begin{aligned}
U_{j+1} - U_j &= \frac{w(v_i) + S_j + d(v_i, v'_{j+1})}{j+2} - \frac{w(v_i) + S_j}{j+1} \\
&= \frac{1}{(j+1)(j+2)} \left[(j+1) [w(v_i) + S_j + d(v_i, v'_{j+1})] - (j+2)(w(v_i) + S_j) \right]
\end{aligned}$$

$$= \frac{1}{(j+1)(j+2)} [(j+1)d(v_i, v'_{j+1}) - w(v_i) - S_j] = \frac{1}{j+2} [d(v_i, v'_{j+1}) - U_j].$$

Suppose now that $U_{j+1} - U_j \geq 0$; we shall show that $U_{j+2} - U_{j+1} \geq 0$.

$$\begin{aligned} U_{j+2} - U_{j+1} &= \frac{1}{j+3} (d(v_i, v'_{j+2}) - U_{j+1}) = \frac{1}{j+3} \left(d(v_i, v'_{j+2}) - \frac{(j+1)U_j + d(v_i, v'_{j+1})}{j+2} \right) \\ &\geq \frac{1}{j+3} \left(d(v_i, v'_{j+1}) - \frac{(j+1)U_j + d(v_i, v'_{j+1})}{j+2} \right) \\ &= \left(\frac{1}{j+3} \right) \left[\frac{j+1}{j+2} (d(v_i, v'_{j+1}) - U_j) \right] = \left(\frac{j+1}{j+3} \right) (U_{j+1} - U_j) \geq 0 \end{aligned}$$

and the correctness of **SLAVE_RST2** follows; since it runs in $O(n^2)$, condition [M-ST-2] is satisfied. Consider now the following algorithm for **RST2**.

```
BEGIN /*MASTER_RST2*/
  F ← ∅;
  FOR v ∈ V DO u[v] ← 1;
  WHILE ∃v ∈ V such that u[v] = 1 DO
    Sv*,x* ← SLAVE_RST2( $\vec{u}$ );
    F ← F ∪ {Sv*,x*};
    FOR v ∈ Sv*,x* DO u[v] ← 0 OD;
  OD
  OUTPUT POSTPROCESS(F);
END. /*MASTER_RST2*/
```

Obviously, the complexity of algorithm **MASTER_RST2** is $O(n^3)$. In all, we have proved that definition 1 applies for **RST2**; furthermore, $\Delta_{\text{RST2}} \leq n$ and the result of the theorem follows. ■

The result of theorem 3 can be slightly modified to apply even in the restricted case of **RST2** where the degree of a star in any solution has to be bounded by a positive integer constant k . It suffices to add, in the **WHILE** loop of procedure **SLAVE_RST2**, an exit-criterion if the number j of uncovered (i.e., having $\vec{u}$ -values equal to 1) neighbours of v_i becomes greater than k and the following corollary holds.

Corollary 1. *There exists a PTAA with ratio $1 + \ln(k+1)$ for the restriction of **RST2** where the degree of a star in any solution has to be at most k .*

Since a tree of depth 2 is of diameter 4, application of algorithm **MASTER_RST2** considering every vertex of V as a potential root leads to the following corollary.

Corollary 2. *The 4-diameter minimum spanning tree problem is approximable within ratio $1 + \ln n$.*

3.2 Minimum bounded-diameter spanning forest

In this section we prove that the minimum k -diameter spanning forest problem, called k -DSF in what follows, is **M-ST** and, since it is an unweighted problem, it admits a PTAA achieving ratio $\ln n - \ln \ln n + 0.78$.

Minimum k -diameter spanning forest.

Given a graph $G = (V, E)$, find a minimum-cardinality partition of V into trees of diameter at most k , (the diameter of a tree T is the maximum number of edges in a path between any pair of vertices in T).

This problem is **NP**-hard (reduction from minimum dominating set for $k = 2$, [6]). The rest of the section is devoted to the proof of the following theorem.

Theorem 4. *k -DSF is **M-ST** and, consequently, approximable within $\ln n - \ln \ln n + 0.78$.*

Proof. Let us first prove satisfaction of condition [M-ST-1]; consider the following procedure, receiving as inputs a set $V_0 \subset V$ and a positive integer r .

```

BEGIN /*SPANF( $V_0, r$ )*/
   $F \leftarrow \emptyset$ ;
   $V^c \leftarrow V_0$ ;
  FOR  $i \leftarrow 1$  TO  $r$  DO
     $V_i \leftarrow \Gamma(V_{i-1}) \setminus V^c$ ;
    FOR  $v \in V_i$  DO
      let  $v'$  be a vertex of  $\Gamma(v) \cap V_{i-1}$ ;
      add  $vv'$  in  $F$ ;
    OD
     $V^c \leftarrow V^c \cup V_i$ ;
  OD
  OUTPUT  $F$ ;
END; /*SPANF( $V_0, r$ )*/

```

In the output F of procedure **SPANF**, every vertex of V_i is linked to a unique vertex of V_0 by a unique path of length i . Consequently, F is a forest of size $|V_0|$ and of depth r . Moreover, this forest is *maximal*, in the sense that it contains *all* vertices of V linked to some vertex of V_0 by a path of at most r edges. Now, let $F = \{T_1, \dots, T_t\}$ be a cover of V by trees of diameter at most k (cover in the sense that a vertex of V can belong to more than one tree of F).

- If $k = 2r$, let r_i be a root of T_i (i.e., a vertex of $V(T_i)$ such that any other vertex in T_i is linked to r_i by a path of at most r edges) and consider $V_0 = \{r_i : i = 1, \dots, t\}$. For this case, procedure **SPANF**(V_0, r) polynomially transforms tree-cover F into a tree-partition of at most the same cardinality, satisfying condition [M-ST-1].
- For the case where $k = 2r + 1$, a tree of diameter at most k can be seen as composed of two rooted trees of depth at most r having their roots connected by an edge which we will call “edge-root” in the sequel; let $r_i r'_i$ be an “edge-root” of T_i (i.e., an edge of $E(T_i)$ such that any other vertex in T_i is connected either to r_i , or to r'_i by a path of at most r edges) and consider $E_0 = \{r_i r'_i : i = 1, \dots, t\}$; moreover, let V_0 be the set of vertices endpoints of an edge in E_0 . A modified version of **SPANF** can be used here. It consists of first calling **SPANF**(V_0, r) to normally produce a set of trees of depth r , next of finding in E_0 a maximal matching $M \subset E_0$ and, finally, of unifying trees having edges of M as edge roots; let us denote by **MSPANF**(E_0, r) this modified version of **SPANF** and by F' the forest computed. Since every vertex of V not in V_0 is within distance at most r from V_0 , F' is indeed a partition of V into trees of diameter at most $k = 2r + 1$, containing $|M|$ trees having edges of M as edge roots, and $|V'|$ other trees ($V' \subset V_0$ denoting the set of vertices of V_0 unsaturated by M). Now, since M is maximal for E_0 , every vertex of V' is connected by an edge (in E_0) to a vertex of $V_0 \setminus V'$. Hence, $|E_0| \geq |M| + |V'|$, i.e., $|F| \geq |F'|$ and condition [M-ST-1] is satisfied.

Consider now the following algorithm for k -DSF. Instructions between parentheses refer to the case $k = 2r + 1$; in this case, remark that, when called from a single edge e , procedure **MSPANF**(e, r) constructs a maximal tree of depth r and edge-root e , i.e., a tree containing all vertices of V lying within distance at most r from an endpoint of e .

```

BEGIN /*MASTER_k-DSF*/
(1)   $r \leftarrow \lceil k/2 \rceil$ ;
(2)   $F \leftarrow \emptyset$ ;
(3)   $V_0 \leftarrow \emptyset$ ;
(4)  FOR  $v \in V$  DO  $u[v] \leftarrow 1$  OD
(5)  WHILE  $\exists v \in V, u[v] = 1$  DO
(6)       $\max \leftarrow 0$ ;
(7)      mark all vertices of  $V$  (* all edges of  $E$  *) as unvisited;
(8)      WHILE  $\exists v_0 \in V$  (*  $\exists v_0 v'_0 \in E$  *) unvisited DO
(9)           $T \leftarrow \text{SPANF}(\{v_0\}, r)$  (*  $\text{MSPANF}(\{v_0 v'_0\}, r)$  *);
(10)         IF  $\sum_{v \in T} u[v] > \max$  THEN
(11)              $T^* \leftarrow T$ ;
(12)              $\max \leftarrow \sum_{v \in T} u[v]$ ;
(13)         FI
(14)         mark  $v_0$  (* $v_0 v'_0$  *) as visited;
(15)     OD
(16)      $F \leftarrow F \cup \{T^*\}$ ;
(17)      $V_0 \leftarrow V_0 \cup \{v_0\}$ ; (*  $E_0 \cup \{v_0 v'_0\}$  *)
(18)     FOR  $v \in V(T^*)$  DO  $u[v] \leftarrow 0$  OD
(19) OD
(20) OUTPUT  $\text{SPANF}(V_0, r)$ ; (*  $\text{MSPANF}(E_0, r)$  *)
END.

```

We shall finally prove that lines (6) to (15) of algorithm **MASTER_k-DSF** correctly solve the slave problem, hence satisfying condition [M-ST-2], i.e., that tree T^* at line (16) maximizes quantity $\sum_{v \in V(T)} u[v]$, where T ranges over all trees of diameter at most k . We shall call T_{opt} an optimal tree. Since T_{opt} has diameter at most k , then if k is even (resp., odd) there exists a vertex v_{opt} (resp., an edge e_{opt}) such that every vertex of T_{opt} is linked to v_{opt} (resp., to one of the endpoints of e_{opt}) by a path of length at most $r = \lceil k/2 \rceil$. On the other hand, procedure **SPANF** (resp., **MSPANF**), called by algorithm **MASTER_k-DSF** from every vertex v_0 (resp., edge $v_0 v'_0$) of G produces a maximal tree of root v_0 (resp., edge-root $v_0 v'_0$) and depth r . Let $\hat{T}$ be the tree computed by **SPANF** (resp., **MSPANF**), when called from v_{opt} (resp., e_{opt}). As $\hat{T}$ is maximal, it contains all the vertices linked to v_{opt} (resp., e_{opt}) by paths of length at most r ; consequently, $V(T_{opt}) \subset V(\hat{T})$ and $\sum_{v \in V(T_{opt})} u[v] \leq \sum_{v \in V(\hat{T})} u[v] \leq \sum_{v \in V(T^*)} u[v]$; hence, tree T^* is also optimal.

Obviously, the above algorithm is polynomial; moreover, $\Delta_{\text{kDSF}} \leq n$ and this completes the proof of theorem 4. ■

3.3 Edge-covering by trees

The last problem studied in this paper, denoted by ECT, is an edge-covering problem frequently met in VLSI-design whenever one tries to minimize the cost of the circuit connexions ([11]).

Edge-covering by weighted trees.

Given a graph $G = (V, E)$ and positive integer valuations $\vec{d}$ on its edges, find a collection of trees $F = \{T_1, \dots, T_t\}$ satisfying $\cup_{i=1, \dots, t} \{E(T_i)\} = E$ and minimizing $c(F) = \sum_{1 \leq i \leq t} \bar{c}(T_i)$, where $\bar{c}(T) = \max_{e \in E(T)} \{d(e)\}$.

To our knowledge, the complexity of ECT is still open. Therefore, since no exact polynomial algorithm is known, we devise a PTAA achieving approximation ratio $O(\ln n)$.

Theorem 5. *ECT is M-ST. Consequently, it is approximable by a PTAA achieving approximation ratio bounded above by $1 + \ln n$.*

Proof. Condition [M-ST-1] of definition 1 is satisfied since every feasible solution of ECT is a cover of E by trees (the cost of which is the sum of the costs of the solution-trees).

We shall now show that condition [M-ST-2] is also satisfied. Given a 0-1 vector $\vec{u}$ associated with E , we show that solution of the slave-problem, i.e., computation of

$$T^* = \operatorname{argmax}_T \left\{ R(T; \vec{u}) = \frac{\sum_{e \in E(T)} u[e]}{\bar{c}(T)} \right\}$$

can be performed in polynomial time.

Let $E_\delta = \{e \in E : d(e) \leq \delta\}$, and $G_\delta = (V, E_\delta)$. Moreover, we denote by $d(G)$ the quantity $\max_{e \in E} \{d(e)\}$, and by $\underline{d}(G)$, the quantity $\min_{e \in E} \{d(e)\}$. Consider now the following procedure optimally solving the slave problem.

```

BEGIN /*SLAVE_ECT( $\vec{u}$ )*/
(1)  max  $\leftarrow 0$ ;
(2)   $\delta \leftarrow d(G)$ ;
(3)  WHILE  $\delta \geq \underline{d}(G)$  DO
(4)     $\operatorname{argmax}_T \{\sum_{e \in E(T)} u[e]\} = T^* \leftarrow \text{KRUSKAL}(G_\delta)$ ;
(5)     $\delta \leftarrow \bar{c}(T^*) - 1$ ;
(6)    IF  $R(T^*; \vec{u}) > \text{max}$  THEN
(7)      max  $\leftarrow R(T^*; \vec{u})$ ;
(8)      retain  $T^*$ ;
(9)    FI
(10) OD
(11) output  $T^*$ ;
END; /*SLAVE_ECT( $\vec{u}$ )*/

```

Algorithm KRUSKAL in line (4) is a modified version of the classical minimum weight spanning tree algorithm ([10]) where, instead of a minimum-distance edge, a maximum-distance one is selected at each step in graph G_δ . Moreover, let us note that KRUSKAL(G_δ) treats $u[e]$'s (and not $d[e]$'s) as edge-distances ($e \in E_\delta$). Since KRUSKAL runs in time $O(|E| \log |E|)$, and the WHILE loop will be executed at worst $|E|$ times, then the whole procedure is polynomial.

We now prove that procedure SLAVE_ECT correctly solves the slave problem for ECT, i.e., the output-tree T^* satisfies $T^* = \operatorname{argmax}_T R(T; \vec{u})$. Suppose that the WHILE loop of the above procedure is executed p times and denote, for $i = 1, \dots, p$, by δ_i the value of δ considered in line (3) and by T_i^* , the tree T^* computed in line (4), at the i th execution of the WHILE loop. We have, for $i = 1, \dots, p$, $\bar{c}(T_i^*) = \delta_{i+1} + 1$. Moreover, set $\delta_{p+1} = \delta_p - 1$, and observe that $\delta_p = \underline{d}(G)$ and $\delta_1 = d(G)$. Consider now, at some execution i of the WHILE loop, $1 \leq i \leq p-1$, a tree T such that $\delta_{i+1} + 1 \leq \bar{c}(T) \leq \delta_i$. We have $\sum_{e \in E(T)} u[e] \leq \sum_{e \in E(T_i^*)} u[e]$, otherwise T would have been selected instead of T_i^* by KRUSKAL. Since $\bar{c}(T) \geq \bar{c}(T_i^*)$, one immediately gets $R(T_i^*; \vec{u}) \geq R(T; \vec{u})$. So, $T_i^* = \operatorname{argmax}_{\delta_{i+1}+1 \leq \bar{c}(T) \leq \delta_i} \{R(T; \vec{u})\}$ and for the final tree T^* obtained at line (11), the following holds:

$$T^* = \operatorname{argmax}_{i=1, \dots, p} \{R(T_i^*; \vec{u})\} = \operatorname{argmax}_{i=1, \dots, p} \left\{ \max_{\delta_{i+1}+1 \leq \bar{c}(T) \leq \delta_i} R(T; \vec{u}) \right\} = \operatorname{argmax}_{\delta_p \leq \bar{c}(T) \leq \delta_1} \{R(T; \vec{u})\}.$$

So, since $\delta_p = \underline{d}(G)$ and $\delta_1 = d(G)$, the optimality of procedure SLAVE_ETC for the slave problem is concluded.

In order to complete the proof of theorem 5, it suffices to consider the following algorithm for ECT.

```

BEGIN /*MASTER_ECT*/
   $\mathcal{C} \leftarrow \emptyset$ ;
  FOR  $e \in E$  DO  $u[e] \leftarrow 1$  OD
  WHILE  $\exists e \in E$  with  $u[e] = 1$  DO
     $T^* \leftarrow \text{SLAVE\_ECT}(\vec{u})$ ;
     $\mathcal{C} \leftarrow \mathcal{C} \cup \{T^*\}$ ;
    FOR  $e \in E(T^*)$  DO  $u[e] \leftarrow 0$  OD
  OD
  OUTPUT  $\mathcal{C}$ ;
END. /*MASTER_ECT*/

```

Let us note that, if all edge-distances are identical, ECT consists in finding the minimum-cardinality edge-cover by trees. In this case, procedure SLAVE_ECT is nothing else but a single call of KRUSKAL(G) (line (4)). ■

Before closing this section, let us recall our introductory remark that the first two problems are provably inapproximable within $(1 - \epsilon) \ln n$ ([1]), and thus lie among the hardest problems of the class **APX**($\log n$). Consequently, the ratio attained here is asymptotically optimal.

4 Discussion: introducing a new kind of reduction

A usual way for transferring (positive or negative) approximation results from a problem to another is by approximation-preserving reductions (A -reductions [12], P -reductions [5], continuous reductions [13], etc.). In these reductions the general underlying idea is that, given two problems Π and Π' (let us suppose wlog that these problems are minimization ones), one can polynomially transform a generic instance I of Π into an instance I' of Π' ; next, on the hypothesis that a ρ -approximation PTAA A exists for Π' , one shows how a Π' -solution S' for I' can be polynomially transformed to a Π -solution S for I such that if $|S'|/\text{OPT}(I') \leq \rho$ then $|S|/\text{OPT}(I) \leq c(\rho)$, for some $c : [1; \infty[\rightarrow [1; \infty[$, where $|S'| = A(I')$ and $|S|$ denotes the size (cardinality or value) of solution S . Usually people call approximation-preserving the reductions where $c(\rho) = \rho$.

Intuitively, theorem 1 (as well as the results of section 3) describes, in some sense, a kind of “reduction” from an **M-ST** problem to WSC. Of course, since this “reduction” is not polynomial (the number of sets is in all the cases studied exponential in n , the order of the graph), it is not approximation-preserving in the common sense. However, we feel that there still exist strong approximation links between these problems and the greedy algorithm for set covering, and that these links can be written in a formal way. We thus introduce the following kind of reduction.

Definition 3. Let Π and Π' be two minimization problems, and $A_{\Pi'}$ be a PTAA for Π' . A *pseudo-reduction* from Π to $(\Pi', A_{\Pi'})$ is a triple (φ, ψ, A_{Π}) such that

- [1] φ transforms an instance I of Π into an instance $I' = \varphi(I)$ of Π' ;
- [2] ψ transforms a Π' -solution S' for I' into a Π -solution $S = \psi(S')$ for I such that $\exists c : [1; \infty[\rightarrow [1; \infty[$, $|S'|/\text{OPT}(I') \leq \rho \implies |S|/\text{OPT}(I) \leq c(\rho)$;
- [3] A_{Π} is a PTAA for Π and, if S and S' denote the solutions returned by A_{Π} and $A_{\Pi'}$ on I and I' , respectively, then $S = \psi(S')$.

We will denote by $\Pi \hookrightarrow (\Pi', A_{\Pi'})$ the fact that Π is pseudo-reducible to $(\Pi', A_{\Pi'})$ with $c(\rho) = \rho$. ■

Pseudo-reduction differs from classical reductions by the fact that functions φ and ψ are not constrained to be computable in time polynomial in $|I|$, and that, given a PTAA $\mathbf{A}_{\Pi'}$ for Π' , we are only interested, in solutions computed by algorithm $\mathbf{A}_{\Pi'}$ in I' . Another meaningful difference is also that solution $S = \varphi(S')$ for I is not constructed polynomially from S' (I' is not, in general, constructible in polynomial time), but it is constructed directly on I by a kind of “simulation” of algorithm $\mathbf{A}_{\Pi'}$. Since $\mathbf{A}_{\Pi'}(I')/\text{OPT}(I') \leq \rho \implies \mathbf{A}_{\Pi}(I)/\text{OPT}(I) \leq c(\rho)$, if $c(\rho) = \rho$, then the pseudo-reduction can be considered as approximation-preserving. We thus have the following proposition.

Proposition 1. *If $\Pi \hookrightarrow (\Pi', \mathbf{A}_{\Pi'})$ and $\mathbf{A}_{\Pi'}$ approximately solves (in polynomial time) Π' within ratio ρ , then Π is polynomially approximable within ρ .*

We have seen in the proof of theorem 1 that, for covering (resp., partitioning) **M-ST** problems, triple $(\varphi, \psi, \text{MSGREEDY})$ (resp., $(\varphi, \psi \circ h, \text{MSGREEDY})$) satisfies definition 3. We can thus recover a concept of completeness to our pseudo-reduction.

Proposition 2. *WSC is M-ST-hard, in the sense that $\forall \Pi \in \mathbf{M-ST}, \Pi \hookrightarrow (\text{WSC}, \text{WSCGREEDY})$.*

From propositions 1 and 2 we find again our former result, namely every **M-ST** problem is approximable within logarithmic ratio.

References

- [1] L. Alfandari. Approximation de problèmes de couverture et de partitionnement de graphes. PhD Thesis. LAMSADE, Université Paris-Dauphine. In preparation.
- [2] L. Alfandari and V. Th. Paschos. Approximating the minimum-weight rooted spanning tree with radius less than two. Manuscript, 1997.
- [3] B. B. Boppana and M. M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. *BIT*, 32(2):180–196, 1992.
- [4] V. Chvátal. A greedy-heuristic for the set covering problem. *Math. Oper. Res.*, 4:233–235, 1979.
- [5] P. Crescenzi and A. Panconesi. Completeness in approximation classes. In *Proc. Fundamentals of Computation Theory*, number 380 in Lecture Notes in Computer Science, pages 116–126. Springer Verlag, 1989.
- [6] M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.
- [7] D. S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Appl. Math.*, 6:243–254, 1983.
- [8] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9:256–278, 1974.
- [9] D. S. Johnson. Worst-case behavior of graph-coloring algorithms. In *Proc. South-Eastern Conference on Combinatorics, Graph Theory and Computing*, pages 513–528, 1974.
- [10] J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.*, 7(1):48–50, 1956.

- [11] M. Minoux. Some combinatorial optimization problems from VLSI-design. Research Seminar “Optimization”. Université Paris I, 1991.
- [12] A. Panconesi and D. Ranjan. Quantifiers and approximation. In *Proc. STOC’90*, pages 446–456, 1990.
- [13] H. U. Simon. On approximate solutions for combinatorial optimization problems. *SIAM J. Disc. Math.*, 3(2):294–310, 1990.
- [14] P. Slavík. A tight analysis of the greedy algorithm for set cover. In *Proc. STOC’96*, pages 435–441, 1996.
- [15] G. Yu and O. Goldsmidt. On locally optimal independent sets and vertex covers. In *Proc. International Federation for Information Processing*, pages 533–542, 1993.

Appendix

A Pseudo-reduction and maximization packing problems

A.1 Dual master-slave approximation

Let us now consider the dual master-slave game where one tries to approximately solve a maximization **NP**-hard problem by iteratively solving a minimization slave problem. A typical instantiation of this game is the greedy maximum independent set algorithm where one iteratively introduces in the solution the minimum-degree vertex and removes from the input-graph the selected vertex and its neighbours.

In the sequel, we will try to answer to the following question: “can one use pseudo-reductions to produce approximation results for graph-problems where the objective is to find a maximum packing of the input-graph into subgraphs satisfying given properties?”. For this, we will link the approximation of this type of problems to the approximation of another classical **NP**-hard optimization problem, the maximum set-packing (SP).

Definition 4. Consider an **NP**-hard minimization graph-problem Π and a property $\mathcal{P}$; suppose that (i) a feasible solution of Π is a subset of $2^{\mathcal{S}_P}$ and (ii) a cost function $\bar{c}$, computable in polynomial time, is associated with $S \in \mathcal{S}_P$ such that the cost of a solution $\mathcal{S}'$ of Π is $c(\mathcal{S}') = \sum_{S \in \mathcal{S}'} \bar{c}(S)$. Then $\mathcal{P}$ is *dual master-slave tractable* (**DM-ST**) iff it satisfies the following conditions:

- [DM-ST-1] a solution $\mathcal{S}'$ of Π is a $\mathcal{P}$ -packing (i.e., a subset $\mathcal{S}' \subset \mathcal{S}_P$ such that for $S, S' \in \mathcal{S}'$, $V(S) \cap V(S') = \emptyset$);
- [DM-ST-2] given a binary vector $\vec{u} \in \{0, 1\}^n$ associated with V , the (slave) problem of finding the subgraph $S^* = \operatorname{argmin}_{S \in \mathcal{S}_P} \{\sum_{v \in V(S)} u[v] / (\min_{v \in V(S)} \{u[v]\} \bar{c}(S))\}$ is in **P**.

For the case of edge-packing, we simply replace v by e and V by E . ■

Consider now the following algorithm for weighted SP (WSP) operating on a SP-instance $(\bar{\mathcal{S}}, \cup_{\bar{S}_i \in \bar{\mathcal{S}}} \bar{S}_i)$ where weights $w(\bar{S}_i)$ are associated with every set $\bar{S}_i \in \bar{\mathcal{S}}$.

```

BEGIN /*WSPGREEDY*/
   $\bar{\mathcal{S}}' \leftarrow \emptyset$ ;
  REPEAT
     $\bar{S}_j \leftarrow \operatorname{argmin}_{\bar{S}_i \in \bar{\mathcal{S}}} \{|\bar{S}_i| / w(\bar{S}_i)\}$ ;
     $\bar{\mathcal{S}}' \leftarrow \bar{\mathcal{S}}' \cup \{\bar{S}_j\}$ ;
     $\bar{\mathcal{S}} \leftarrow \bar{\mathcal{S}} \setminus (\{\bar{S}_j\} \cup \{\bar{S}_i : \bar{S}_i \cap \bar{S}_j \neq \emptyset\})$ ;
  UNTIL  $\bar{\mathcal{S}} = \emptyset$ ;
  OUTPUT  $\bar{\mathcal{S}}'$ ;
END. /*WSPGREEDY*/

```

The above algorithm achieves worst case ratio $1/\Delta_{\text{SP}}$ for WSP, where $\Delta_{\text{SP}} = \max_{\bar{S}_i \in \bar{\mathcal{S}}} \{|\bar{S}_i|\}$ ([7]).

Theorem 6. If Π is a **DM-ST** problem, then $\Pi \hookrightarrow (\text{WSP}, \text{WSPGREEDY})$; consequently, Π is polynomially approximable within $1/\Delta_{\Pi}$.

Proof. Consider a vertex-packing **DM-ST** problem Π (case of edge-packing problems is completely analogous) and the following transformation φ of an instance G of Π into an instance $\varphi(G)$ of SP: for every subgraph $S \in \mathcal{S}_P$, we add a set $\bar{S} = V(S)$ with weight $w(\bar{S}) = \bar{c}(S)$ in $\mathcal{S}$ (note that the number of sets can be exponential in n). There exists a 1-1 correspondence between the solutions of Π on G and the solutions of SP on $\varphi(G)$ constructed as above, i.e., $\mathcal{S}' = \{S_1, S_2, \dots, S_t\}$ is a feasible graph-packing for G , iff $\varphi(\mathcal{S}') = \{\bar{S}_1, \bar{S}_2, \dots, \bar{S}_t\}$ is a feasible set-packing in $\varphi(G)$ of the same value. Hence, $\text{OPT}(G) = \text{OPT}(\varphi(G))$.

Let us now consider the following “simulation” of WSPGREEDY.

```

BEGIN /*DMSGREEDY*/
(1)  FOR  $v \in V$  DO  $u[v] \leftarrow 1$  OD
(2)   $\mathcal{S}' \leftarrow \emptyset$ ;
(3)  STOP  $\leftarrow$  FALSE;
(4)  WHILE  $\neg$ STOP DO
(5)       $S^* \leftarrow \operatorname{argmin}_{S \in \mathcal{S}_P} \{ \sum_{v \in V(S)} u[v] / (\min_{v \in V(S)} \{ u[v] \} \bar{c}(S)) \}$ ;
(6)      IF  $\sum_{v \in V(S^*)} u[v] / (\min_{v \in V(S^*)} \{ u[v] \} \bar{c}(S^*)) = \infty$  THEN
(7)          STOP  $\leftarrow$  TRUE;
(8)          GOTO line (12);
(9)      FI
(10)      $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{S^*\}$ ;
(11)     FOR  $v \in V(S^*)$  DO  $u[v] \leftarrow 0$  OD
(12) OD
(13) OUTPUT  $\mathcal{S}'$ ;
END. /*DMSGREEDY*/

```

It suffices now to remark that, in the SP-instance $\varphi(G)$, thanks to the IF-block (lines (6) to (9)), a set intersecting another set already introduced in $\mathcal{S}'$ will never be selected to be put in $\mathcal{S}'$. Moreover, at line (5), the set minimizing the ratio “cardinality over cost” is selected. Consequently, algorithm **DMSGREEDY** works in $\varphi(G)$ exactly as algorithm **WSPGREEDY** works in a generic SP-instance. This completes the proof of the theorem. ■

In the same way as in section 2, one can define the class of **EDM-ST** problems where condition [DM-ST-2] of definition 4 is relaxed by allowing ρ -approximated computation ($\rho \geq 1$) of the quantity $\operatorname{argmin}_{S \in \mathcal{S}_P} \{ \sum_{v \in V(S)} u[v] / (\min_{v \in V(S)} \{ u[v] \} \bar{c}(S)) \}$ (in polynomial time). In this case, with arguments similar (though much easier) to the ones of section 2, the following theorem can be proved.

Theorem 7. *An EDM-ST problem Π is polynomially approximable within $1/(\rho\Delta_\Pi)$.*

A.2 Pseudo-reduction and improved approximations for unweighted graph-packing problems

Consider a set packing $\bar{\mathcal{S}}' \subseteq \bar{\mathcal{S}}$. A natural (and polynomial) way to improve it is to perform *2-improvements* of $\bar{\mathcal{S}}'$, i.e., to search for triples $(\bar{S}_i, \bar{S}_j, \bar{S}_k)$ such that $\bar{S}_i \in \bar{\mathcal{S}}'$, $\{\bar{S}_j, \bar{S}_k\} \in \bar{\mathcal{S}} \setminus \bar{\mathcal{S}}'$, $\bar{S}_j \cap \bar{S}_k = \emptyset$, $\{\bar{S}_j\} \cap (\bar{\mathcal{S}}' \setminus \{\bar{S}_i\}) = \emptyset$, $\{\bar{S}_k\} \cap (\bar{\mathcal{S}}' \setminus \{\bar{S}_i\}) = \emptyset$. In this case, $(\bar{\mathcal{S}}' \setminus \{\bar{S}_i\}) \cup (\{\bar{S}_j, \bar{S}_k\})$ is a set-packing of cardinality $|\bar{\mathcal{S}}'| + 1$.

The following algorithm, relying on 2-improvements of an initial SP-solution, approximately solves SP.

```

BEGIN /*2_IMPSP*/
(1)  compute a maximal set-packing  $\bar{\mathcal{S}}'$ ;
(2)  WHILE there exists 2-improvement  $(\bar{S}_i, \bar{S}_j, \bar{S}_k)$  DO
(3)       $\bar{\mathcal{S}}' \leftarrow (\bar{\mathcal{S}}' \setminus \{\bar{S}_i\}) \cup \{\bar{S}_j, \bar{S}_k\}$ ;
(4)      make  $\bar{\mathcal{S}}'$  maximal for the inclusion;
(5)  OD
(6)  OUTPUT  $\bar{\mathcal{S}}'$ ;
END. /*2_IMPSP*/

```

Note that lines (1) and (4) can be very easily computed by a simple greedy algorithm iteratively selecting a set and removing the ones having non-empty intersections with it.

Algorithm 2_IMPSP is a simplified version of the one of Yu and Goldschmidt ([15]) proposed for solving IS in k -claw-free graphs (i.e., graphs containing no independent set of k vertices,

all adjacent to a common vertex). As it is proved there, when running in k -claw-free graphs it guarantees, in time $O(n^3)$, independent sets, the sizes of which are at least $2/k$ times the size of the maximum ones. On the other hand, for SP-instances where the cardinality of the maximum-size set is Δ_{SP} , their intersection graphs are $\Delta_{\text{SP}} + 1$ -claw-free (the intersection graph of a SP-instance $\bar{\mathcal{S}} = \{\bar{S}_1, \dots, \bar{S}_n\}$ is a graph $G_{\bar{\mathcal{S}}} = (V, E)$ where $V = \bar{\mathcal{S}}$ and $E = \{v_i v_j : \{\bar{S}_i, \bar{S}_j\} \subseteq \bar{\mathcal{S}}, \bar{S}_i \cap \bar{S}_j \neq \emptyset\}$.) Since an independent set V' of $G_{\bar{\mathcal{S}}}$ becomes an equal-size set packing of $\bar{\mathcal{S}}$, and vice-versa, via the replacement of the vertices of V' by the same-index sets of $\bar{\mathcal{S}}$, *algorithm 2_IMPSP achieves, in $O(n^3)$, approximation ratio $2/(\Delta_{\text{SP}} + 1)$ for SP.*

The notion of 2-improvement can be extended as follows in order to fit with general packing graph-problems.

Definition 5. Let $\mathcal{S}$ be a maximal $\mathcal{P}$ -vertex-packing in a graph G and $V(\mathcal{S}) = \cup_{S \in \mathcal{S}} V(S)$. A *2_packing-improvement* of $\mathcal{S}$ in G is a triple (S_i, S_j, S_k) such that $S_i \in \mathcal{S}$, $\{S_j, S_k\} \in \mathcal{S}_{\mathcal{P}}$, $V(S_j) \subset [V \setminus V(\mathcal{S})] \cup V(S_i)$, $V(S_k) \subset [V \setminus V(\mathcal{S})] \cup V(S_i)$, $V(S_j) \cap V(S_k) = \emptyset$. For the case of $\mathcal{P}$ -edge_packing, it suffices to replace $V(\cdot)$ by $E(\cdot)$. ■

Obviously, the set $(\mathcal{S} \setminus \{S_i\}) \cup \{S_j, S_k\}$ is a $\mathcal{P}$ -packing of size $|\mathcal{S}| + 1$.

Let us now consider **NP**-hard maximization graph-problems Π satisfying the following conditions:

1. there exists a property $\mathcal{P}$ such that any feasible solution of Π in G is a $\mathcal{P}$ -packing;
2. the measure of each feasible solution is its cardinality;
3. the problems of (i) finding a subgraph of G satisfying property $\mathcal{P}$ (if any) and (ii) finding two vertex-disjoint subgraphs of G satisfying property $\mathcal{P}$ (if such a pair exists), are both in **P**.

Also, consider the following algorithm 2_IMPGP for Π .

```

BEGIN /*2_IMPGP*/
  compute a maximal  $\mathcal{P}$ -packing  $\mathcal{S}'$  in  $G$ ;
  WHILE there exists a 2_packing-improvement  $(S_i, S_j, S_k)$  DO
     $\mathcal{S}' \leftarrow (\mathcal{S}' \setminus \{S_i\}) \cup \{S_j, S_k\}$ ;
    make  $\mathcal{S}'$  maximal for the inclusion;
  OD
  OUTPUT  $\mathcal{S}'$ ;
END. /*2_IMPGP*/

```

With arguments very similar to the ones used in the proofs of theorems 1 and 6 and considering algorithm 2_IMPGP as a kind of simulation of 2_IMPSP for Π , then the following theorem can be proved.

Theorem 8. *If Π satisfies conditions 1, 2 and 3, then $\Pi \hookrightarrow (\text{SP}, 2_ \text{IMPSP})$ and is approximable within ratio $2/(\Delta_{\Pi} + 1)$.*