Theoretical Computer Science 250 (2001) 179–200
www.elsevier.com/locate/tcs

# Generalized submodular cover problems and applications

Judit Bar-Ilan[a,1], Guy Kortsarz[b,1], David Peleg[c,*,2]

[a] *School of Library, Archive and Information Studies, The Hebrew University, Jerusalem 91904, Israel*
[b] *Department of Computer Science, The Open University of Israel, Ramat Aviv, Israel*
[c] *Department of Applied Mathematics and Computer Science, The Weizmann Institute of Science, Rehovot 76100, Israel*

Received February 1996; revised October 1998
Communicated by G. Ausiello

**Abstract**

The greedy approach has been successfully applied in the past to produce logarithmic ratio approximations to NP-hard problems under certain conditions. The problems for which these conditions hold are known as *submodular cover* problems. The current paper[3] extends the applicability of the greedy approach to wider classes of problems. The usefulness of our extensions is illustrated by giving new approximate solutions for two different types of problems. The first problem is that of finding the spanning tree of minimum weight among those whose diameter is bounded by $D$. A logarithmic ratio approximation algorithm is given for the cases of $D = 4$ and 5. This approximation ratio is also proved to be the best possible, unless P = NP. The second type involves some (known and new) center selection problems, for which new logarithmic ratio approximation algorithms are given. Again, it is shown that the ratio must be at least logarithmic unless P = NP. © 2001 Elsevier Science B.V. All rights reserved.

*Keywords:* NP-hard problems; Greedy approximation; Submodular covers

## 1. Introduction

Greedy approximation algorithms for various NP-hard problems were proposed in [4, 12, 15]. A more general framework for using greedy algorithms for approximation was proposed in [6, 20].

* Corresponding author.
*E-mail addresses*: judit@shum.cc.huji.ac.il (J. Bar-Ilan), peleg@wisdom.weizmann.ac.il (D. Peleg).
[1] Part of this work was carried out while the author was with the Department of Applied Mathematics and Computer Science at the Weizmann Institute of Science.
[2] Supported in part by a Walter and Elise Haas Career Development Award and by a grant from the Israel Science Foundation.
[3] A preliminary version of this paper has appeared as an extended abstract in *Proc. 4th Israel Symp. on the Theory of Computing and Systems*, Jerusalem, Israel, June 1996.

The general approximation method introduced in [20] applies to *submodular-cover* problems. A submodular-cover problem is described in the following way. Let $\mathscr{U}$ be a collection of elements and $f : 2^{\mathscr{U}} \mapsto \mathscr{Z}$, an integral, nondecreasing and submodular function. A (positive integral) *cost* is associated with every element of $\mathscr{U}$, and the optimal feasible solution to the problem is a subset $S^*$ of $\mathscr{U}$ with minimum cost such that $f(S^*) = f(\mathscr{U})$. Intuitively, the entire universe $\mathscr{U}$ is a "cover" and the function $f(S)$ measures the extent of which $S$ is close to being a cover as well. That is, $f$ measures how close $S$ is to covering the same amount $\mathscr{U}$ covers. Now, adding a new element to $S$ increases the amount $S$ covers, i.e., increases $f$. Intuitively, the function $f$ is non-increasing and submodular if there is some independence among the "cover-increment" of elements outside $S$. Namely, for every two elements $u, v \in U \setminus S$, the contribution to $f$ of $u$ and $v$ together is no greater than the sum of their separate contributions. More formally, $f(S \cup \{u, v\}) \leqslant f(S \cup \{u\}) + f(S \cup \{v\})$. See also Definition 2.1 and the discussion thereafter.

It is proved in [20] that for a submodular-cover problem, the greedy approach yields a logarithmic ratio approximation algorithm. In particular, this method deals with the weighted set-cover problem, including variants with load bounds on the vertices.

The current paper extends this method in two ways. Section 3 deals with two-phase applications of Wolsey's method, with two *different* submodular functions used in the two phases. Then, Section 4 studies algorithms that apply the greedy procedure a number of times successively.

Section 5 presents a number of new applications to our techniques. One major class of applications concerns logarithmic ratio approximations for some known and new *multicenter selection and allocation* problems. Three examples are given in the paper for this class of applications. The first problem in that class is the *average cost center* problem, which generalizes (the dual of) the $k$-median problem (cf. [5]). The second example is the *fault tolerant center selection* problem, generalizing a variant studied in [3]. A third example concerns *capacitated facility location* problems, which again generalize facility location problems studied in the literature. In all of these cases, the generalized problems discussed here were not given approximation algorithms in the past, to the best of our knowledge. Moreover, for all of these problems, it is also shown that the result is apparently near-optimal, in the sense that unless $\mathrm{P} = \mathrm{NP}$, the approximation ratio must be logarithmic.

One last application discussed in the paper, taken from a different domain, concerns the problem of finding the spanning tree of minimum weight among those whose diameter is bounded by $D$. A logarithmic-ratio approximation is given for this problem for $D = 4$ and 5. (The case $D = 4$ is given some applications in the area of information retrieval in [1, 2]. There, shallow trees are used to efficiently compress a collection of bits; the shorter the resulting tree, the faster the process of deciphering the message.) Again, it is shown that the approximation ratio for this problem must be $\Omega(\log n)$, unless $\mathrm{P} = \mathrm{NP}$.

This result was recently generalized, using specialized techniques, to give a polynomial time approximation algorithm of ratio $O(\log n)$ for any constant $D$, and an algorithm of ratio $O(n^{\varepsilon})$, for any fixed $0 < \varepsilon < 1$, for general $D$ [13].

## 2. Preliminaries

### 2.1. Framework

We start by introducing some definitions and notations that will be used in what follows. Let $\mathscr{U}$ be a finite set. From now on we only consider integral valued nonnegative functions $f : \mathscr{U} \mapsto \mathscr{Z}$.

**Definition 2.1.** The function $f$ is
1. *nondecreasing* if $f(S) \leqslant f(T)$ for all $S \subseteq T \subseteq \mathscr{U}$,
2. *submodular* if $f(S) + f(T) \geqslant f(S \cup T) + f(S \cap T)$ for all $S, T \subseteq \mathscr{U}$.

**Notation 2.2.** Given a set $\mathscr{U}$, a function $f$, a subset $S \subseteq \mathscr{U}$ and an element $x \in \mathscr{U}$, denote
- $\Delta_f(S, x) = f(S \cup \{x\}) - f(S)$ and
- $\Delta_f(S) = \max_{x \in \mathscr{U}}\{\Delta_f(S, x)\}$.

Using this notation, one can express the above two properties in a different but equivalent way, as follows. Given a set $\mathscr{U}$ and a nonnegative integral function $f : 2^{\mathscr{U}} : \mapsto \mathscr{Z}$, we say that $f$ obeys the *improvement independence* (II) axiom if the following holds:

(II) (*Improvement independence*). For every pair of subsets $S, T \subseteq \mathscr{U}$, $S \subseteq T$, $\sum_{u \in T \setminus S} \Delta_f(S, u) \geqslant f(T) - f(S)$.

Clearly, if $f$ is nondecreasing, then restricting Axiom (II) to $S, T$ such that $S \subseteq T$ causes no loss of generality. The following theorem and its proof can be found, e.g., in [17].

**Theorem 2.3** (cf. Nemhauser and Wolsey [17]). *A function $f$ is submodular and nondecreasing if and only if it obeys Axiom* (II).

The next theorem can also be found in [17]. For this theorem, denote $\mathscr{U} = \{u_1, \ldots, u_n\}$.

**Theorem 2.4** (cf. Nemhauser and Wolsey [17]). *Let $f$ and $f'$ be two submodular functions on $\mathscr{U}$, and let $r_j, 1 \leqslant j \leqslant n$, be nonnegative integers, $c$ a nonnegative real and $k$ a real number. Then the following are also submodular on $\mathscr{U}$*:
- $f_1(S) = f(S) + f'(S)$,
- $f_2(S) = c \cdot f(S)$,
- $f_3(S) = \min\{f(S), k\}$,
- $f_4(S) = \sum_{u_i \in S} r_i$.

We remark that this theorem is only used in this paper with integral $c$ and $k$. Let us describe more formally the framework of submodular-cover problems. The input to a submodular-cover problem consists of a finite set $\mathscr{U}=\{u_1,\dots,u_n\}$ and a nondecreasing submodular function $f:2^{\mathscr{U}}\mapsto\mathscr{Z}$. There is a nonnegative *cost* $c_i$ associated with each element $u_i\in\mathscr{U}$. We define the cost of a subset $S\subseteq\mathscr{U}$ as $c(S)=\sum_{u_i\in S}c_i$. The subsets $S$ s.t. $f(S)=f(\mathscr{U})$ are referred to as the *feasible sets*, or feasible solutions. An optimal solution to a submodular-cover problem P is a feasible set $S^*$ such that $c(S^*)$ is minimum.

The idea behind the greedy algorithm for submodular-cover problems is to start from some initial given set (usually, the empty set) and gradually add elements until attaining feasibility. The element picked in each iteration is the "locally optimal" one in terms of its gain-to-weight ratio, namely, the element maximizing $\Delta_f(S,u_i)/c_i$.

The following theorem is proved in [20].

**Theorem 2.5** (Wolsey [20]). *Let* P *be a submodular-cover problem, and let $S^*$ be an optimal solution for* P. *Then the greedy algorithm produces a feasible set $S$ with approximation ratio $c(S)/c(S^*)\leqslant\ln(\Delta_f(\emptyset))+1$.*

## 2.2. The family $ILP^f$ of programs

We next describe a family of linear minimization problems called $ILP^f$. These are problems of the following form:

$$\begin{aligned}\min\quad & \sum_{i=1}^{n}c_ix_i\\ \text{s.t.}\quad & A\bar{z}\geqslant\bar{b},\ \text{where } \bar{z}=\bar{x};\bar{y},\ \bar{x}=(x_1,\dots,x_n),\ \bar{y}=(y_1,\dots,y_q),\\ & x_i\in\{0,1\},\ 0\leqslant y_j\leqslant\gamma_j\quad\text{for } 1\leqslant i\leqslant n, 1\leqslant j\leqslant q,\end{aligned}$$

where $\bar{b}$ consists of $m$ integers (which w.l.o.g. may be assumed to be nonnegative), $c_i>0$ for every $i$, the coefficients of the $x_i$ variables in the $m\times(n+q)$ matrix $A$ are positive, and the $m\times q$ submatrix $A_q$ of the last $q$ columns in $A$ is a "flow matrix" (also called "incidence matrix"). Namely, the coefficients of each $y_j$ in $A_q$ are 0, except for one row containing 1 and one row containing $-1$. The $y_j$ variables represent the arcs and the matrix rows represent the vertices. The 1 (resp., $-1$) entry in the column corresponding to $y_j$ represents the end (resp., start) vertex of the arc. The restriction $0\leqslant y_j\leqslant\gamma_j$ is the capacity constraint for the arc corresponding to $y_j$.

Our analysis for the quality of the greedy approximation algorithm on $ILP^f$ directly extends the ideas of [20]. For an instance of the $ILP^f$ problem, define the corresponding submodular function, similar to the one used in [20] for the bounded load set cover. Let $A_i$ be the $i$th row in the matrix $A$. Define a universe $\mathscr{U}=\{u_1,\dots,u_n\}$ of $n$ elements. Assign each element $u_i$ a cost coefficient $c_i$. Since each $x_i$ is a $0-1$ variable, there is a one-to-one correspondence between vectors $\bar{x}$ and subsets $S\subseteq\mathscr{U}$ of $\mathscr{U}$ (i.e., $S$ consists of the elements $u_i\in\mathscr{U}$ for which $x_i=1$). Hereafter, we denote by $\bar{x}^S$ the characteristic incidence vector corresponding to the set $S\subseteq\mathscr{U}$.

Recall that in the formulation of $ILP^f$, $\bar{z}=\bar{x};\bar{y}$, i.e., $\bar{z}$ is the concatenation of the $\bar{x}$ and $\bar{y}$ vectors. For a fixed vector $\bar{x}^S$ corresponding to a subset $S\subseteq\mathscr{U}$, and some assignment $\bar{Y}$ for the variables $\bar{y}$, let $\bar{z}=\bar{x}^S;\bar{Y}$ and

$$f_1(\bar{z})=\sum_i \min\{A_i\cdot\bar{z},b_i\}$$

and denote

$$f_2(S)=\max_{\bar{Y}}\{f_1(\bar{z})\}.$$

It is clear that if a feasible solution for the $ILP^f$ program exists at all, then $f_2(\mathscr{U})=\sum_i b_i$, and therefore, a subset $S$ for which $f_2(S)=f_2(\mathscr{U})$ is a feasible solution to the $ILP^f$ program.

A bound on the approximation ratio of the greedy algorithm on $ILP^f$ follows from Theorem 2.5 since $f_2$ is submodular and nondecreasing. We are not aware of a reference to an explicit proof of submodularity for precisely this function $f_2$ in the literature, although the proof follows a well-understood path. For completeness, we provide a proof of this fact in the appendix.

Letting $B_{\max}=\max\{1,\max_i\{b_i\}\}$, we have the following result.

**Lemma 2.6.** *The $ILP^f$ problem has an* $O(\log(mB_{\max}))$ *approximation algorithm.*

For future reference, recall the relation of the above-defined function $f_2$ to flow theory. Suppose that, given $S$, we plug into the above inequalities the values of $\bar{x}^S$ corresponding to $S$. After rearranging the inequalities, each $b_j$ is replaced with some $b'_j\leqslant b_j$ (this follows since the coefficients of the $x_j$ are nonnegative). This leaves us with a maximum flow instance.

Indeed, construct the following directed flow graph $G(A,S)$. Add a vertex $v_j$ for each row $j$. Add a source $s$ and a sink $t$. If a variable $y_i$ appears with positive sign in row $j$, and with negative sign in row $k$, add an arc emanating from $v_k$ to $v_j$. Each vertex $v_j$ for which $b'_j>0$ is connected with an arc to $t$ of capacity $b'_j$. Also, add an arc from $s$ to each vertex $v_j$ with $b'_j<0$, with capacity $-b'_j$. To compute $f_2(S)$, we simply compute the maximum flow in $G(A,S)$. This maximum flow function sets values for the variables $y_j$, according to the flow on the respective arcs. (A minor subtlety is that a feasible $\bar{y}$ vector does not necessarily correspond to a flow function in $G(A,S)$. Rather, it may correspond to an excessive flow function, where the amount of flow entering each vertex may exceed the amount of flow leaving this vertex. Nevertheless, since in $f_2$ we gain nothing by putting more than $b_i$ units of flow over an arc with capacity $b_i$ entering the sink, an excessive flow can be converted into a legal flow with the same $f_2$ value.)

Finally, we observe that while the logarithmic approximation ratio obtained for $ILP^f$ may seem rather weak, it is asymptotically the best achievable, assuming $P\neq NP$. Moreover, virtually the same happens for all the problems considered throughout the rest of this paper.

Most of these hardness results on approximability easily follow from [18], which proves a hardness result for the *set cover* and *dominating set* problems. The set cover problem can be described as follows. We are given an undirected bipartite graph $G(X,Y,E)$ where the edges cross from $X$ to $Y$. The goal is to choose a minimum subset $W \subseteq X$, such that every vertex in $Y$ has a neighbor in $W$. In the dominating set problem, the input is an undirected graph $G(V,E)$. The goal is to choose the smallest subset $W \subseteq V$ such that every vertex $v \in V \setminus W$ has a neighbor in $W$.

The following theorem is proven in [18], strengthening two similar theorems in [7, 16], which were given under somewhat weaker complexity assumptions.

**Theorem 2.7** (Raz and Safra [18]). *There exists a constant* $c<1$ *such that the set cover and dominating set problems admit no* $c \cdot \ln n$*-ratio approximation*, *unless* $\mathrm{P}=\mathrm{NP}$.

Clearly, our program $ILP^f$ extends the set cover problem. Hence we have the following.

**Corollary 2.8.** *There exists a constant* $c<1$ *such that the* $ILP^f$ *problem admits no* $c \cdot \ln m$*-ratio approximation algorithm*, *unless* $\mathrm{P}=\mathrm{NP}$.

## 3. Two-phase greedy algorithms

This section considers a class of linear minimization problems denoted $ILP^c$. This class is an extension of the class $ILP^f$, which is no longer directly within the above scheme. We provide the first general approximation algorithm for problems in this class, based on two phases of greedy selection, using two different submodular functions.

The class $ILP^c$ consists of programs of the $ILP^f$ form with the additional *flow-cost inequality*

$$\sum_{j=1}^{n} r_j x_j - \sum_{j=1}^{q} l_j y_j + \alpha \geqslant 0 \tag{1}$$

for nonnegative $r_j$ and $l_j$, $1 \leqslant j \leqslant n$ and $1 \leqslant j \leqslant q$. The above inequality has a "flow-cost" interpretation, namely, the coefficient $l_j$ is the cost of the arc corresponding to $y_j$. For fixed $x_i$, this inequality bounds the cost of the flow function.

More precisely, let $(L2)$ denote the instance of the $ILP^c$ program at hand and by $(L1)$ the $ILP^f$ program obtained from $(L2)$ by eliminating the flow-cost inequality. Define the flow graph $G(A,S)$ for $(L1)$ as in the previous section. The flow-cost inequality (1) requires that $\sum l_j y_j$, which is exactly the flow-cost in $G(A,S)$, be bounded by some $\alpha'$. For fixed $S$ (and therefore for fixed incidence vector $\bar{x}^S$), denote by $G(A,S,\bar{l})$ the max-flow min-cost instance corresponding to the $ILP^c$ program. (This is the flow graph $G(A,S)$ extended by adding the flow-cost coefficients $l_j$ on the arcs.)

Let $B_{\max} = \max_j\{b_j\}$ and $L_{\max} = \max_j\{l_j\}$ in the instance. Start with some feasible solution $S_0 \subseteq \mathcal{U}$ for $(L_1)$. Let $\hat{Y}$ be an assignment of values that maximizes the flow in $G(A,S_0)$, namely, $f_1(\bar{x}^{S_0};\hat{Y}) = f_2(S_0)$. Let $Y_{\max} = \max_i\{\hat{Y}_i\}$. Since the maximum possible flow in $G(S,A)$ is bounded by $m \cdot B_{\max}$, we may assume that this assignment $\hat{Y}$ satisfies

$$Y_{\max} \leqslant mB_{\max}. \tag{2}$$

Since $S_0$ is feasible for $(L_1)$, there exists for $S_0$ an assignment $\bar{Y}$ for the $\bar{y}$ variables such that the inequalities $A \cdot (\bar{x}^{S_0};\bar{Y}) \geqslant \bar{b}$ are satisfied. However, it may be impossible to satisfy the flow-cost inequality with $S_0$. We therefore need to extend $S_0$ to a larger set $S \cup S_0$, for which it is possible to satisfy the flow-cost inequality as well.

Define a new universe $\mathcal{U}_0 = \mathcal{U} \backslash S_0$. Now, for every set $S \subseteq \mathcal{U}_0$, let $\bar{z} = \bar{x}^{S \cup S_0};\bar{y}$, and define

$$\begin{aligned} f_3(S) \quad &= \quad \max_{y_j}\{-\textstyle\sum l_j y_j\} \\ &\quad\ \text{s.t.} \quad A\bar{z} \geqslant \bar{b}. \end{aligned}$$

Intuitively, once plugging the $\bar{x}^{S \cup S_0}$ values in $ILP^c$, one gets a max-flow min-cost instance $G(A,S,\bar{l})$. Since $S_0$ is a solution to $(L_1)$, and $S_0 \subseteq S \cup S_0$, we know that there is an assignment $\hat{Y}$ of values for the $\bar{y}$ variables, such that for $\hat{z} = \bar{x}^{S \cup S_0};\hat{Y}$, $A \cdot \hat{z} \geqslant \bar{b}$. In terms of flow, this means that one can choose a flow function on the corresponding flow graph, so as to saturate all the arcs entering the sink. Among all the $\bar{Y}$ vectors maximizing the flow (and therefore, among the flow functions saturating all the arcs entering the sink), we look for the one minimizing the flow-cost. In summary, in $f_3(S)$ we compute a max-flow min-cost function.

Now, further define for any set $S \subseteq \mathcal{U}_0$

$$f_4(S) = \min\left\{\sum_{x_j \in S} r_j x_j + f_3(S), -\alpha\right\}.$$

Note that if there is a feasible solution at all, then $f_4(\mathcal{U}_0) = -\alpha$. Also note, that if $S \subseteq \mathcal{U}_0$ satisfies $f_4(S) = -\alpha$, then $S_0 \cup S$ (with the $\bar{Y}$ vector achieving the minimum for $f_3$) is a feasible solution to $(L_2)$, since the flow-cost inequality is also satisfied.

It follows from the above discussion that it is possible to use the greedy method for approximating $ILP^c$ in two phases, as follows. First, greedily find a feasible solution $S_0$ for $(L1)$, and then extend it in $\mathcal{U}_0$ (using the greedy algorithm again) into a feasible solution for $(L2)$ using the above function $f_4$.

Let us now estimate the resulting approximation ratio. Let $S^*$ be an optimal solution for $(L2)$ and $c(S^*)$ its cost. Since the cost of the optimal solution for the corresponding program $(L1)$ is no greater than $c(S^*)$, the next corollary follows from Lemma 2.6.

**Corollary 3.1.** *The solution $S_0$ provided by the first phase of the Algorithm for $(L1)$ satisfies $c(S_0) \leqslant (\ln(mB_{\max})+1)c(S^*)$.*

In order to show that the greedy algorithm is efficient for the second phase, we must prove that $f_4$ is nondecreasing and submodular. First, note that $f_3$ is nondecreasing since adding more elements to a set $S$, may only decrease the flow-cost (note also that $r_j \geqslant 0$). Also, it follows from Theorem 2.4 that for proving submodularity of $f_4$, it suffices to prove that $f_3(S)$ is submodular. The submodularity of $f_3$ can be proven via standard flow properties. For completeness, the proof is given in the appendix.

Note, that by inequality (2) we may bound $\Delta_{f_4}(\emptyset)$ from above by $qY_{\max}L_{\max} \leqslant qmB_{\max}L_{\max}$. Thus the bound stated next on the ratio for the two-phase greedy algorithm follows from [20] (note that $q = \mathrm{O}(m^2)$).

**Corollary 3.2.** *The ILP$^c$ problem has an* $\mathrm{O}(\log m + \log M)$ *ratio approximation, where $M$ is the maximum integer in the instance.*

## 4. Multiple applications of the greedy procedure

This section introduces an extended class of linear optimization problems, denoted *ILP$^s$*, and shows how to approximate it using multiple application of the greedy procedure combined via a binary elimination procedure.

The programs of *ILP$^s$* are of the following form:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c_i x_i + \sum_{j=1}^{q} l_j y_j \\ \text{s.t.} \quad & A\bar{z} \geqslant \bar{b}, \text{ where } \bar{z} = \bar{x}; \bar{y},\ \bar{x} = (x_1, \ldots, x_n),\ \bar{y} = (y_1, \ldots, y_q), \\ & l_j \geqslant 0,\ x_i \in \{0,1\},\ 0 \leqslant y_j \leqslant \gamma_j \quad \text{for } 1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant q \end{aligned}$$

and where $A$ is a flow matrix as characterized before, and $c_i$, $1 \leqslant i \leqslant n$ and $l_j$, $1 \leqslant j \leqslant q$ are nonnegative.

Note that by imposing a flow-cost bound on $\sum l_j y_j$, one gets the following *ILP$^c$* program:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c_i x_i \\ \text{s.t.} \quad & A\bar{z} \geqslant \bar{b}, \text{ where } \bar{z} = \bar{x}; \bar{y},\ \bar{x} = (x_1, \ldots, x_n),\ \bar{y} = (y_1, \ldots, y_q), \\ & \sum_{j=1}^{q} l_j y_j \leqslant Q, \\ & x_i \in \{0,1\},\ y_j \in \mathscr{Z}^+ \quad \text{for } 1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant q. \end{aligned} \tag{3}$$

It is easy to see that by performing a *sequential* search on the possible values of $Q$, it is possible to obtain a good approximation for *ILP$^s$*. However, this procedure will not be polynomial in the input size. The solution is to use a *binary elimination* procedure, to describe next.

For any solution $S$ for the $ILP^s$ problem denote

$$F_1(S)=\sum_{i=1}^{n} c_i x_i \quad \text{and} \quad F_2(S)=\sum_{j=1}^{q} l_j y_j.$$

Let $F(S)=F_1(S)+F_2(S)$ denote the $ILP^s$ objective function value for $S$. The procedure maintains two bounds $Q_{\text{high}} \geqslant Q_{\text{low}}$ on the value of $\sum_j l_j y_j$, and the search is conducted in the interval $(Q_{\text{low}}, Q_{\text{high}})$. It is possible to start the search with, say, the largest possible interval, namely, $Q_{\max}=q\cdot m\cdot B_{\max}\cdot L_{\max}$ and $Q_{\min}=0$. Throughout the search, we keep track of the best solution $S$ (the solution with minimum $F(S)$) encountered so far, and at the end output the best $S$.

The crucial step in the resulting algorithm, named Algorithm Multi-Phase, is the rule by which we choose between the upper and lower half intervals in the search, as the new flow-cost bound, since it is essential to show that we cannot considerably minimize the objective function, searching in the intervals eliminated by the search.

Let us define a binary search rule as follows.

1. Compute program (3) with $Q_{\text{mid}}$ as the flow-cost bound. Let $S$ be the resulting solution.
2. If $F_1(S) \geqslant Q_{\text{mid}}$ then continue the search in the upper half interval.
3. Else $(F_1(S) < Q_{\text{mid}})$ choose the lower half interval.

Let us now analyze the approximation ratio of Algorithm Multi-Phase. As before, let $S^*$ be the optimal solution for the $ILP^s$ program.

**Lemma 4.1.** *Consider an iteration of Algorithm* Multi-Phase *producing a solution $S$ for $ILP^s$. Assume that $F_1(S) \geqslant Q_{\text{mid}}$ and that $F_2(S^*) \leqslant Q_{\text{mid}}$. Then*

$$F(S)=F_1(S)+F_2(S) \leqslant \mathrm{O}(\log Mm)\cdot F(S^*).$$

**Proof.** Let $\tilde{P}$ denote the $ILP^c$ instance resulting by setting the flow-cost bound $Q$ to be $Q_{\text{mid}}$ in the linear program (3). By the restriction on the flow-cost introduced in this program, it follows that the solution $S$ of the algorithm in the $i$th iteration satisfies

$$F_2(S) \leqslant Q_{\text{mid}}. \tag{4}$$

Let $\tilde{c}$ be the value of the objective function in an optimal $ILP^c$ solution for $\tilde{P}$. It follows from Corollary 3.2 that

$$F_1(S) \leqslant \tilde{c}\cdot \mathrm{O}(\log(Mm)). \tag{5}$$

Since $F_2(S^*) \leqslant Q_{\text{mid}}$ by the assumption of the lemma, it follows that $S^*$ is a feasible solution for $\tilde{P}$ as well, and the value it achieves for the objective function is $F_1(S^*)$, so the optimal solution for $\tilde{P}$ satisfies $\tilde{c} \leqslant F_1(S^*)$. From this and Eq. (5) we have $F_1(S) \leqslant F_1(S^*)\cdot \mathrm{O}(\log(Mm))$. Consequently, we conclude by Eq. (4), the assumption that $F_1(S) \geqslant Q_{\text{mid}} \geqslant F_2(S)$ and the fact that $F_2(S) \geqslant 0$ for every $S$, that

$$F(S^*)\cdot \mathrm{O}(\log(Mm)) \geqslant F_1(S^*)\cdot \mathrm{O}(\log(Mm)) \geqslant F_1(S) \geqslant (F_1(S)+Q_{\text{mid}})/2 \geqslant F(S)/2,$$

and the desired claim follows. □

**Lemma 4.2.** *Consider an iteration of Algorithm* MULTI-PHASE. *Let $S$ be the solution produced by this iteration, and let $Q_{\mathrm{mid}}$ be the bound imposed. Assume that $F_1(S) \leqslant Q_{\mathrm{mid}}$. Further assume that $F_2(S^*) \geqslant Q_{\mathrm{mid}}$. Then $F(S) \leqslant 2 \cdot F(S^*)$.*

**Proof.** By the assumption of the claim, both inequalities $F_1(S) \leqslant Q_{\mathrm{mid}}$ and $F_2(S) \leqslant Q_{\mathrm{mid}}$ hold. The lemma now follows as

$$F(S^*) \geqslant F_2(S^*) \geqslant Q_{\mathrm{mid}} \geqslant (F_1(S) + F_2(S))/2 = F(S)/2.$$

Combining Lemmas 4.1 and 4.2 we conclude that for every solution $S'$ in the range ruled out by the search, $F_2(S')$ is within a factor of $O(\log(Mm))$ from $F(S)$ for the current best $S$. Thus either $F_2(S^*)$ is in an interval that the search ruled out, in which case we are in a "good" situation, or $F_2(S^*)$ equals one of the two (consecutive) numbers $Q_{\mathrm{high}}$, $Q_{\mathrm{low}}$, of the final interval. In this later case, these numbers can be used as cost-bounds directly. In summary, we have the following theorem.

**Theorem 4.3.** *Algorithm* MULTI-PHASE *requires polynomial time, and yields* $O(\log M + \log m)$*-approximation for* $ILP^s$.

## 5. Applications

The final section illustrates some applications of the generalized methods presented in the previous sections. In all the coming problems, matching upper and lower bounds are proved for the approximation ratio, under the assumption that $P \neq NP$. (At times, we need to make the additional assumption that various weights involved in the problem definition are polynomial in the number of inequalities.)

### *5.1. The average cost centers (Center(sum)) problem*

The input to this problem is a weighted complete graph with $V = \{1, \ldots, n\}$, a cost function $c_i$ on the vertices, a weight function $w$ on the arcs, and bounds $U$ and $L_j$ for $1 \leqslant j \leqslant n$. The number $c_i$ represents the cost of establishing and maintaining a center at the site represented by the vertex, and the weights $w_{ij}$ represent the cost of communication between vertices, or their physical distance, etc. Some of the weights may be $\infty$. The requirement is to choose a minimum cost set of centers. Each noncenter vertex has to be served by a "sufficiently close" center, i.e., a center for which the corresponding arc weight is bounded by some weight bound $\rho$. We also need to bound the number of clients served by a single center $j$ by $L_j$, and the total weight of the (chosen) center-clients arcs by $U$.

We define a variable $y_{ij}$ iff $w_{ij} \leqslant \rho$. For every $1 \leqslant i, j \leqslant n$, let $y_{ji} = 1$ if $j$ is the center serving $i$, and let $x_i = 1$ if $i$ is chosen as a center. The appropriate $ILP^c$ problem is

$$\begin{array}{ll} \min & \sum_{i=1}^{n} c_i x_i \\ \text{s.t.} & \sum_{j} y_{ji} \geqslant 1 \quad \text{for } 1 \leqslant i \leqslant n, \end{array}$$

$$L_j x_j - \sum_i y_{ji} \geqslant 0 \quad \text{for } 1 \leqslant j \leqslant n,$$

$$U - \sum_{j,i} y_{ji} \cdot w_{ij} \geqslant 0$$

$$y_{ji},\ x_i \in \{0,1\} \quad \text{for } 1 \leqslant i,j \leqslant n.$$

Indeed, this program fits the *ILP*$^c$ framework. All the coefficients of the $\bar{x}$ variables are positive. Also, except for the flow-cost inequality $U - \sum_{j,i} y_{ji} \cdot w_{ij} \geqslant 0$, each variable $y_{ji}$ appears in one row with the coefficient 1 (in the constraint $\sum_j y_{ji} \geqslant 1$ corresponding to $i$) and in one other row with the coefficient $-1$ (in the constraint $L_j x_j - \sum_i y_{ji} \geqslant 0$ corresponding to $j$).

**Corollary 5.1.** *Let $M$ be the maximum integer appearing in the* CENTER(*sum*) *problem. Then the two-phase algorithm yields an* $O(\log n + \log M)$ *approximation for it.*

Note that this problem generalizes the dominating set problem. Indeed, any instance $G(V,E)$ of the dominating set problem can be transformed into an instance of the CENTER(*sum*) problem as follows. Take the complete graph $G(V, V \times V)$, put costs $c_i = 1$ on the vertices, unit weights on the edges corresponding to $E$, and weight $\infty$ on the "nonedges" $(V \times V) \setminus E$. Finally set $L_i = n$ and $U = \infty$. Clearly, a solution for the resulting CENTER(*sum*) problem is also a solution for the dominating set problem. Hence relying on Theorem 2.7 we have

**Corollary 5.2.** *There exists a constant $c < 1$ such that the* CENTER(*sum*) *problem admits no $c \cdot \ln n$-ratio approximation, unless* P = NP.

**Related problems.** This problem belongs to the class of multicenter (or $k$-center) problems (cf. [11]), but it differs from the original $k$-center problem in a number of ways. First, it measures the total (rather than maximum) distance between clients and their servers, hence in that respect it resembles the *k-median* problem (cf. [5]). Secondly, it is dual to the classical problem, in that instead of assuming a fixed bound on the number (or cost) of the center and optimizing the distances, it assumes a bound on the distances and optimizes the cost. Finally, our variant also allows us to impose a balancing condition on the loads of the centers, similar to the load-balanced version of [3] for the classical $k$-center problem.

A variant of this ILP program, without the flow-cost inequality $U - \sum_{j,i} y_{ji} \cdot w_{ij} \geqslant 0$, has already appeared in [20]. In our extension, the flow-cost inequality bounds the total cost of the center-client arcs.

## *5.2. The fault tolerant center selection problem*

Consider the following fault-tolerant variant of the center problem. Given a directed graph, $G(V,E)$ and a set $S \subseteq V$ of "clients", choose a set of centers of minimum cost, such that every noncenter which is also in $S$, will have at least $k$ arc- (or vertex-) disjoint paths from the center vertices. Also, we want to bound the total cost of the path arcs.

For simplicity, denote $V=\{1,\dots,n\}$. Let us first reformulate the requirement of the problem as follows. Consider a single fixed client $v\in S$. Given the set $C$ of centers, in order to get $k$ arc-disjoint paths from $C$ to $v$ one needs to select a subset $E_v\subseteq E$ of the arcs, such that there are at least $k$ arcs of $E_v$ entering $v$, and also, for every $j\notin C$, the number of arcs of $E_v$ entering $j$ should be at least equal to the number of arcs of $E_v$ leaving $j$. The existence of such $E_v$ is clearly equivalent to the existence of $k$ appropriate disjoint paths from $C$ to $v$. The *ILP*$^c$ program we write for this problem has the following form.

$$\begin{aligned}
\min \quad & \sum_{i=1}^{n} c_i x_i \\
\text{s.t.} \quad & k x_v + \sum_i y_{iv}^v \geqslant k \quad \text{for } v\in S,\\
& n x_j + \sum_i y_{ij}^v - \sum_i y_{ji}^v \geqslant 0 \quad \text{for } 1\leqslant v,\ j\leqslant n,\ v\neq j.\\
& \sum_{ij} w_{ij} y_{ij}^v \leqslant U,\\
& y_{ij}^v,\ x_i\in\{0,1\} \quad \text{for } 1\leqslant i,j\leqslant n.
\end{aligned}$$

The interpretation of this program is the following. The variable $x_v$ is set to 1 if and only if $v$ is chosen as a center. The variables $y_{ij}^v$ correspond to the graph arcs $\langle i,j\rangle$ and to $v$. One way of interpreting this is that there is a copy $G_v$ of the graph for every $v$. The variable $y_{ij}^v$ corresponds to the edge $\langle i,j\rangle$ in the copy $G_v$. For a vertex $v\in S$, the set of variables $y_{ij}^v$ set to 1, should induce in $G$ a subset $E_v$ as explained above, i.e., a subset $E_v$ inducing in $G$ a graph $G(V,E_v)$ in which there are $k$ arc-disjoint vertices from the set of centers to $v$.

Now, the first type of inequality states that $v\in S$ is a center or there are at least $k$ arcs of $E_v$ entering $v$. The second inequality says that the number of arcs of $E_v$ entering a vertex $j$ is at least as large as the number of arcs of $E_v$ leaving $j$. In both types, the first term is introduced in order to take care of vertices belonging to the set of centers (vertices $v\in S$ for which $x_v=1$).

A slight variant of this program will solve also the case where the paths are required to be vertex-disjoint.

It is easy to see that the above program is in *ILP*$^c$ form. Specifically, all the coefficients of the $x$ variables are positive. Also, the matrix restricted to the $\bar{y}$ variables is a flow matrix, since each variable $y_{ij}^v$ indeed appears in one row (the row corresponding to $j$ and $v$) with coefficient $+1$ and in another row (the one corresponding to $i$ and $v$) with coefficient $-1$.

**Corollary 5.3.** *The two-phase algorithm yields an approximation algorithm with ratio* $\mathrm{O}(\ln(kn)+\log(M))$ *for the fault-tolerant center problem.*

It is easily seen that this problem generalizes the set cover problem. Given $G(X, Y, E)$ an instance of the set cover problem, a corresponding instance of the fault-tolerant center selection problem can be constructed as follows. Direct all the edges from $X$ to $Y$. Give all the $Y$ vertices cost $\infty$, and all the vertices in $X$ unit cost. Let $S = Y$ and $k = 1$. Clearly, a solution for the above problem is equivalent to a solution for the set cover problem. Hence relying on Theorem 2.7 again we have:

**Corollary 5.4.** *There exists a constant $c < 1$ such that the fault-tolerant center selection problem admits no $c \cdot \ln$-ratio approximation, unless* P = NP.

**Related problems.** The vertex-disjoint variant of the problem, for the restricted case without edge weights and without the flow-cost constraint $\sum_{ij} w_{ij} y_{ij}^{v} \leqslant U$, was first described in [3], and given a (slightly weaker) $O(k \cdot \log n)$-ratio approximation algorithm. The generalized variant presented here, allowing us to place also a bound on the total cost of the paths used in the solution, has not been studied in the past.

## 5.3. Capacitated facility location problems

This problem deals with a number of facilities that are supposed to serve customers "cheaply". Each customer has a demand $b_i$ (which is a nonnegative integral number) for the product generated by the facilities. The cost of satisfying a unit demand of customer $j$ from a facility at $i$ is $cost_{i,j}$ (the costs may be $\infty$). It may happen that several centers must deal with a single customer in order to satisfy the demand. We denote by $y_{i,j}$ the number of units of the product given by $i$ to $j$. The cost of establishing and maintaining a center at $v_i$ is $c_i$. Given a set $S$ and an assignment function determining $y_{i,j}$ the total cost corresponding to $S$ is

$$c(S) = \sum_{i \in S} c_i + \sum_{i \in S, j \in \bar{S}} y_{i,j} \cdot cost_{i,j}.$$

We also wish to bound the load on each center.

We formulate the above constraints as the following $ILP^s$ program. The variable $x_i$ is 1 if a facility is located at vertex $i$. The load bound on each center $i$ is $L_i$ (which may be $\infty$).

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c_i x_i + \sum_{i,j} y_{ij} \cdot cost_{i,j} && (6) \\ \text{s.t.} \quad & \sum_{i} y_{i,j} \geqslant b_j \quad \text{for } 1 \leqslant j \leqslant n, \\ & L_i \cdot x_i - \sum_{j=1}^{n} y_{i,j} \geqslant 0 \quad \text{for } 1 \leqslant i \leqslant n, \\ & x_i \in \{0,1\}, \ \ y_{ij} \in \mathscr{Z}^{+} \quad \text{for } 1 \leqslant i, j \leqslant n. \end{aligned}$$

We have the following corollary.

**Corollary 5.5.** *The capacitated facility location problem can be approximated with ratio* $O(\log M + \log n)$, *where* $n$ *is the number of vertices and* $M$ *is the largest weight.*

Clearly, this problem too is a generalization of the dominating set problem, and hence we have

**Corollary 5.6.** *There exists a constant* $c<1$ *such that the capacitated facility location problem admits no* $c \cdot \ln n$*-ratio approximation, unless* $P = NP$.

**Related problems.** This problem is a variant of the usual (uncapacitated) facility location problem, in which there are no loads, and $L_i = n$ for all $i$ (see, for example, [9, 17]).

There has been some previous work on approximations for some uncapacitated variants of the problem, which are somewhat simpler to handle. In particular, an $O(\log)$ algorithm is given for the uncapacitated problem in [10]. In addition, considerable amount of research has been devoted to the *metric* case of the problem, in which the weights obey the triangle inequality. A 3.16-ratio approximation algorithm for the uncapacitated metric facility location problem is presented in [19]. For references to previous work on these problems (as well as to some work on the related $k$-median problem) see [14].

To the best of our knowledge, our paper gives the first logarithmic approximation for the *capacitated* version of the problem.

## *5.4. Low-diameter minimum spanning trees*

Our final application for the extended scheme is for the following problem. Given an undirected $n$-vertex graph $G=(V,E)$ with a (nonnegative) weight function $w$ on the edges, and a parameter $D$, we look for the minimum weight spanning tree among the trees with diameter bounded by $D$ (the diameter is measured by the number of edges, i.e., the edges are thought of as having *length* 1). Call a tree with a diameter bounded by $D$ a $D$-diameter tree, and denote this problem by *D-MST*. We consider the cases $D=4$ and 5. In a 4-diameter tree, there must be a vertex rooting a tree of height at most 2. Thus a (rooted) 3-layered spanning tree is required with minimum weight. We provide a logarithmic ratio approximation algorithm for the 4-*MST* and 5-*MST* problems (the 3-*MST* case is trivial). We also give a matching lower bound (up to constants).

First, we use Theorem 2.7 to prove a lower bound for the approximability of the problem.

**Claim 5.7.** *There exists a constant* $c<1$ *such that the* 4-*MST problem has no* $c \cdot \ln n$ *approximation, unless* $P = NP$.

**Proof.** We show that a $\rho$-ratio approximation algorithm for 4-*MST* implies a $\rho$-ratio approximation algorithm for the dominating set problem. Let $G(V,E)$ be an instance of the dominating set problem where $|V| = n$. Add three new vertices, $a, b, c$. Connect $a$ to all the vertices in $G$, with edge weights $n^3$, connect $a$ to $b$ and $b$ to $c$, with edge weights 1. Give all the edges in $G$ weight $n$. Denote the resulting graph by $\tilde{G}$.

We make use of the following two observations.

**Observation 1.** Given a dominating set of size $s^*$ in $G$ we can construct in $\tilde{G}$ a 4-diameter spanning tree with weight $s^* n^3 + (n - s^*)n + 2$.

**Observation 2.** Given a 4-diameter spanning tree for $\tilde{G}$, with weight bounded by $sn^3 + o(n^3)$, where $s \leqslant n - 1$, then $a$ is the root, and the tree has no more than $s + 1$ vertices in layer 1 (where the root is on layer 0).

Observation 2 is justified by noting that the vertex $c$ cannot be the root of the 3 layered spanning tree, and neither do the vertices of $G$. Moreover, if $b$ is the root, we must use all the $n^3$-weight edges connecting $a$ to $G$, making the weight greater than $n^4 > (n-1)n^3 + o(n^3) \geqslant sn^3 + o(n^3)$, assuming $n$ is sufficiently large. Also, if there are more than $s$ vertices in layer 1, other than $b$, then there are at least $s + 1$ vertices of $G$ in layer 1, forcing the weight to be at least $(s+1)n^3 > sn^3 + o(n^3)$ for sufficiently large $n$.

Now, assume that we can approximate the 4-*MST* problem with approximation ratio $\rho$. Denote the size of the minimum dominating set in $G$ by $s^*$. Throughout, we assume that $s^* < (n-1)/\rho$. (If this is not the case, then any dominating set we output is within the desired ratio $\rho$.)

Denote the tree resulting from the assumed approximation by $T$. Denote the minimum weight 4-*diameter* tree in $\tilde{G}$ by $T^*$ . Let $T_1$ be a 4-*diameter* spanning tree with weight bounded by $s^* n^3 + \Theta(n(n - s^*))$ (see Observation 1). Thus,

$$w(T) \leqslant \rho w(T^*) \leqslant \rho w(T_1) = \rho(s^* n^3 + \Theta(n(n - s^*))).$$

It follows from Observation 2 that the set of layer one vertices in $T$ is a dominating set in $G$ of size bounded above by $\rho s^*$, which in turn implies an $\rho$ approximation algorithm for the dominating set problem.

We now match this hardness result (up to constants) by a positive result, showing how to fit the 4-*MST* problem into our *ILP*$^s$ scheme. We first introduce a procedure that allows us to deal only with edge-weights that are polynomial in $n$. Let the edges be sorted by nondecreasing weights, and let $E_i = \{e_1, \ldots, e_i\}$ and $w_i = w(e_i)$. Let $G_i$ be a graph induced by the edges $E_i$. Let $j$ be the first index for which $G_j$ contains a tree of diameter 4. Then the problem admits a solution with cost bounded by $(n-1)w_j$, so clearly all edges heavier than $(n-1)w_j$ are unnecessary. Also, the weight of the edges lighter than $w_j/n$ can be made zero, as in any possible spanning tree their total weight is less than $w_j$, hence ignoring their weight altogether will make no difference so long as we are interested in a logarithmic ratio approximation. Consequently, we note that the weights can now be scaled (by dividing them by $w_j$) so that the maximum weight be bounded by $O(n^2)$. This makes all the numbers in the input polynomial in the number of vertices.

In the approximation we make use of the following procedure Shallow-MST($v_1$) (for $V=\{v_1,\dots,v_n\}$, and weights $w_{ij}\geqslant 0$, where $w_{ii}=0$, for every $i$), which approximates the best solution when $v_1$ is taken as the root the tree.

**Procedure Shallow-MST($v_1$)**

1. Assign the *neighbors* $v_i$ of $v_1$ cost $c_i=w_{1i}$, and the non-neighbors of $v_1$ cost $c_i=\infty$.
2. Write the following program:

$$\begin{aligned}
\min \quad & \sum_{i=2}^{n} c_i x_i + \sum_{i,j} y_{ij} w_{ij} \\
\text{s.t.} \quad & \sum_{i=2}^{n} y_{ij} \geqslant 1 \quad \text{for } 2\leqslant j\leqslant n, \\
& (n-1)\cdot x_i - \sum_{j=2}^{n} y_{i,j} \geqslant 0 \quad \text{for } 2\leqslant i\leqslant n, \\
& x_i, y_{ij} \in \{0,1\} \quad \text{for } 2\leqslant i,\ j\leqslant n.
\end{aligned}$$

   and approximate it using Multi-Phase. (Note that we have a variable $y_{ii}$ for every $i$. Also, if $x_i=1$, we can set $y_{ii}$ to be 1, at no cost.)
3. Place the vertices $v_j$ for which $x_j=1$ in layer 1 of the tree.
4. Place the remaining vertices in layer 2, and connect each of them to a vertex in layer 1 according to the $y_{ij}$ values.

To approximate the 4-*MST* problem, we apply procedure Shallow-MST($v_1$) $n$ times, once for every vertex $v_i$ (serving as $v_1$), and output the best tree.

Note that we can also bound the number of children of any vertex *except the root* by any desired bound $L$ (and thus, we "almost" deal with the *bounded-degree* 4-*MST* problem) and get the same ratio.

It is straightforward to show that the above procedure finds a tree whose total weight is $O(\log n + \log W)$ away from the optimum (where $W$ is the maximum edge weight). Thus, since by the above transformation, the edge-weights are polynomial in the number of vertices, we have:

**Corollary 5.8.** *The best polynomial time approximation algorithm for the* 4-*MST problem has ratio* $\Theta(\log n)$.

A similar approximation algorithm follows for the case $D=5$. In a 5-diameter tree, there are two adjacent centers. These are the two end-vertices $v$ and $w$ of the middle edge $e=(v,w)$ in any (length-5) diameter in the tree. Once we know this edge, we can contract its two vertices into some super-vertex $u$, of degree $deg(v)+deg(w)-2$. We then give each edge $(z,u)$ the minimum of the two weights $w((z,v))$, $w((z,w))$. Once this is done, the problem is transformed into the 4-diameter case with $u$ as the root. It is only needed to find the best 4-diameter tree rooted at $u$, and then de-contract the edge $e$ appropriately. Note that we add $w(e)=w((v,w))$ to the weight of the tree.

Thus, the following simple procedure gives the desired logarithmic ratio approximation. Go over the edges one by one. For each edge $e_i = (v_i, w_i)$ contract the edge $e_i$ and get a super-vertex $u_i$. Approximate the 4-*MST* problem, with $u_i$ as the chosen root, using procedure SHALLOW-MST($u_i$). Let $T_i$ be the resulting tree in the approximation. Compute the sum $S_i = w(T_i) + w(e_i)$. Let $j$ be the index achieving the minimum for this sum. De-contract $e_j$ appropriately, and return the resulting tree.

**Related problems.** The problem discussed here is listed as ND4 in [8]. It has not been given an approximation algorithm before. The current result was recently generalized, using specialized techniques, to larger values of $D$. In particular, it was given a polynomial time approximation algorithm of ratio $O(\log n)$ for any constant $D$, and an algorithm of ratio $O(n^{\varepsilon})$, for any fixed $0<\varepsilon<1$, for general $D$ [13].

## Acknowledgements

We are grateful to the two anonymous referees for their helpful comments.

## Appendix A

In the appendix we prove axiom (II) for $f_2$ of Section 2.2 and for $f_3$ of Section 3.

### *A.1. Proof of Axiom* (II) *for* $f_2$

An assignment $\bar{Y}$ to the $\bar{y}$ variables is said to be *optimal for S* if $f_1(\bar{x}^S; \bar{Y}) = f_2(S)$. For every set $S \subseteq \mathcal{U}$, let $\mathcal{Y}_S$ be the set of optimal assignments for $S$.

**Claim A.1.** *Consider sets* $S, T \subseteq \mathcal{U}$ *such that* $S \subseteq T$. *There exists an assignment* $\bar{Y}^S$ *which is optimal for both S and T, namely, such that* $\bar{Y}^S \in \mathcal{Y}_S \cap \mathcal{Y}_T$.

**Proof.** Given two assignments $\bar{Y}$ and $\bar{Y}'$, let $hit(\bar{Y}, \bar{Y}') = \sum_i |Y_i - Y_i'|$. Let $\bar{Y}^S$ and $\bar{Y}^T$ be two assignments minimizing $hit(\bar{Y}^S, \bar{Y}^T)$, where the minimum is taken over all assignment pairs $(\bar{Y}^S, \bar{Y}^T)$ such that $\bar{Y}^S \in \mathcal{Y}_S$ and $\bar{Y}^T \in \mathcal{Y}_T$. We establish the claim by proving that $\bar{Y}^S$ is optimal also for $T$.

Consider some $1 \leqslant i \leqslant q$, and let $j$ and $k$ be the rows in which $y_i$ appears in positive and negative signs, respectively. Notice that if $Y_i^T > Y_i^S$ then clearly

$$A_j \cdot (x^S; \bar{Y}^S) \geqslant b_j \quad \text{and} \quad A_k \cdot (\bar{x}^S; \bar{Y}^S) \leqslant b_k,$$

since otherwise, increasing $Y_i^S$ by 1 would decrease $hit(\bar{Y}^S, \bar{Y}^T)$ conserving the optimality of $\bar{Y}^S$ (and also, of course, conserving the capacity constraints, since $Y_i^S + 1 \leqslant Y_i^T$). This means, for example, that in the inequality of $A_j$, although $A_j \cdot (\bar{x}^S; \bar{Y}^T) > A_j \cdot (\bar{x}^S; \bar{Y}^S)$, it does not help to increase $f_2(S)$ when switching from $\bar{Y}^S$ to $\bar{Y}^T$, since already $A_j \cdot (\bar{x}^S; \bar{Y}^S) \geqslant b_j$.

Similarly, if $Y_i^T < Y_i^S$ then

$$A_j \cdot (x^S; \bar{Y}^S) \leqslant b_j \quad \text{and} \quad A_k \cdot (\bar{x}^S; \bar{Y}^S) \geqslant b_k.$$

It follows that for every $j$,

$$\min\{A_j \cdot (\bar{x}^S; \bar{Y}^S), b_j\} \geqslant \min\{A_j \cdot (\bar{x}^S; \bar{Y}^T), b_j\},$$

because in any coordinate where $\bar{Y}^S$ and $\bar{Y}^T$ differ, the change in $T$ does not help to increase (and may only decrease) $f_2(S)$. From that, it immediately follows that

$$\min\{A_j \cdot (\bar{x}^T; \bar{Y}^S), b_j\} \geqslant \min\{A_j \cdot (\bar{x}^T; \bar{Y}^T), b_j\},$$

since the left term in the minimum was increased by the same amount on each side. By the definition of $f_2$, it follows that $\bar{Y}^S$ is optimal for $T$, completing the proof. $\square$

Next, we prove axiom (II) for $f_2$, relying on Claim A.1. Indeed, consider two fixed sets $S, T \subseteq \mathscr{U}$ such that $S \subseteq T$. It is necessary to show that

$$\sum_{u \in T \setminus S} \Delta_{f_2}(S, u) \geqslant f_2(T) - f_2(S). \tag{A.1}$$

By Claim A.1, and noting that

$$f_2(S \cup \{u\}) = \max_{\bar{Y}} \{f_1(\bar{x}^{S \cup \{u\}}; \bar{Y})\} \geqslant f_1(\bar{x}^{S \cup \{u\}}; \bar{Y}^S),$$

it follows that for establishing inequality (A.1), it suffices to prove that

$$\sum_{u \in T \setminus S} (f_1(\bar{x}^{S \cup \{u\}}; \bar{Y}^S) - f_1(\bar{x}^S; \bar{Y}^S))$$

or, that the function $g(\bar{x}) = f_1(\bar{x}; \bar{Y}^S)$ is submodular. Partitioning $A$ into sub-matrices $\hat{A}; \check{A}$ where $\hat{A}_{m \times n}$ (respectively, $\check{A}_{m \times q}$) consists of the first $n$ (resp., last $q$) columns of $A$, and letting $\bar{h}$ denote the vector of fixed nonnegative numbers $\bar{h} = \check{A} \cdot \bar{Y}^S$, we get that $g(\bar{x}) = \sum_i \min\{\hat{A}_i \cdot \bar{x} + h_i, b_i\}$, hence its submodularity is immediate from Theorem 2.4, since all the coefficients in $A$ are positive. (See also [6].)

## *A.2. Proof of Axiom* (II) *for* $f_3$

In proving axiom (II) for $f_4$, it is easier to rely on the connection of $f_4$ to flow. We start by giving some standard but necessary decomposition properties of flow.

Consider the *ILP*$^c$ program. Let $S, T \subseteq \mathscr{U}$, $S \subseteq T$. Fix corresponding max-flow min-cost assignments $\bar{Y}^S$ and $\bar{Y}^T$ to the $\bar{y}$ variables for $S$ and $T$. That is, $\bar{Y}^S$ and $\bar{Y}^T$ are max-flow min-cost assignments to the arcs of the corresponding directed flow graphs $G(A, S, \bar{l})$ and $G(A, T, \bar{l})$. Given an arc $e$, denote by $Y^S(e)$ (resp., $Y^T(e)$) the flow in $e$ in the $\bar{Y}^S$ (resp., $\bar{Y}^T$) assignment.

Define the following directed *residual* graph $R$ that relies on the connection between $\bar{Y}^S$ and $\bar{Y}^T$. Let $R(\bar{Y}^T, \bar{Y}^S) = (V, A)$, where $V = \{v_i \mid 1 \leqslant i \leqslant m\}$ corresponds to the matrix rows as described in the flow interpretation of $(L1)$. The graph $R$ contains an arc $e = \langle v_1, v_2 \rangle \in A$, if there exists an arc $e'$ touching both $v_1$ and $v_2$ in the original flow graph $G(A, S)$ (hence in $G(A, T)$ as well) and one of the following two cases holds:

- The arc is $e' = \langle v_1, v_2 \rangle$ and $Y^T(e) > Y^S(e)$, or
- The arc is $e' = \langle v_2, v_1 \rangle$ and $Y^T(e) < Y^S(e)$.

In either case, we associate with every $e$ a label $d_e = |Y^T(e) - Y^S(e)|$. As before, assume without loss of generality that $\bar{Y}^T$ and $\bar{Y}^S$ are two optimal (i.e., max-flow min-cost) assignments minimizing the *sum* of labels in the graph,

$$hit(\bar{Y}^T, \bar{Y}^S) = \sum_e d_e.$$

**Lemma A.2.** *$R(\bar{Y}^T, \bar{Y}^S)$ is acyclic.*

**Proof.** For the sake of contradiction, assume the existence of a cycle

$$(v_0, v_1), (v_1, v_2), \ldots, (v_{p-1}, v_0)$$

in $R(\bar{Y}^T, \bar{Y}^S)$. Let $e_i$ be the *original* arc between $v_i$ and $v_{(i+1) \bmod p}$ in the flow graph. Let us modify $\bar{Y}^T$ as follows. For every $i$, if $Y^T(e_i) > Y^S(e_i)$, reduce the flow through $e$ in $\bar{Y}^T$ by 1, and if the $Y^T(e_i) < Y^S(e_i)$, augment the flow through $e$ in $\bar{Y}^T$ by 1. Call the resulting assignment function $\bar{Y}'$. Note that the flow balance in $\bar{Y}'$ at every vertex in the graph remains unchanged after the above modification. We prove that the flow-cost has not changed too, or, that the net change $\mathscr{D}$ in flow-cost satisfies $\mathscr{D} = 0$. Indeed, $\mathscr{D}$ cannot be negative because of the optimality of $\bar{Y}^T$. Furthermore, assuming $\mathscr{D}$ is positive leads to contradiction, as it implies that $\bar{Y}^S$ was not optimal, since the inverse change along the cycle is available for $\bar{Y}^S$.

It follows that $\bar{Y}'$ is also a max-flow min-cost function for $G(A, S)$. However, $hit(\bar{Y}^T, \bar{Y}') < hit(\bar{Y}^T, \bar{Y}^S)$. This is a contradiction. □

We now define a decomposition procedure for $R(\bar{Y}^T, \bar{Y}^S)$. Iteratively identify a collection $\mathscr{P}$ of paths in the residual graph, each carrying one flow unit and bringing $\bar{Y}^S$ "closer" to $\bar{Y}^T$. Each iteration operates as follows. By Lemma A.2. the residual graph of $\bar{Y}^S$ and $\bar{Y}^T$ is acyclic. Consequently, the $j$th iteration selects a directed path $P_j$ from some source vertex $s_j$ (with in-degree 0) to a sink vertex $t_j$ (with out-degree 0). Thereafter, the flow is changed along $P_j$, and the labels in $R = R(\bar{Y}^T, \bar{Y}^S)$ are changed accordingly (deleting zero-labeled arcs) and $P_j$ is added to $\mathscr{P}$. This process continues until exhausting the arcs of $R$. Denote the set of sources encountered by the decomposition procedure by $V_s = \{v \mid v = s_j \text{ for some } j\}$, and the set of sinks by $V_t$, we have the following claim.

**Claim A.3.** *A vertex $v \in V_s$ is never a sink during any iteration of the decomposition procedure. Likewise, a vertex $v \in V_t$ is never a source in any iteration.*

**Proof.** Assume the contrary. If $v$ became a sink before it was a source, it is impossible that later it will have a nonzero out-degree. Similarly, if $v$ becomes a sink after it becomes a source, it is impossible that later on $v$ will have a nonzero in-degree. □

For $v_i \in V_s$, let $s(v_i)$ denote the number of paths in $\mathscr{P}$ starting at $v_i$. Similarly, for $v_j \in V_t$, let $e(v_j)$ be the number of paths of $\mathscr{P}$ ending at $v_j$. Recall that fixing a specific assignment for $\bar{x}$ in the original instance ($L2$) of the $ILP^c$ problem, can be thought of as resulting in a modification of the vector $\bar{b}$ at the right-hand side. This also results in a modification of the flow graph, specifically, the number of flow units available for the vertices. Let $\bar{b}^S$ (resp., $\bar{b}^T$) be the $\bar{b}$ vector resulting from fixing the assignment $\bar{x}^S$ corresponding to $S$ (resp., $\bar{x}^T$ corresponding to $T$). If $b_i^S < 0$ then in the flow graph of $S$, there is an arc from the source $s$ to $v_i$ with capacity $-b_i^S$. We first prove some properties regarding vertices in $V_s$ and $V_t$.

**Claim A.4.** (1) *For every $v_j \in V_t$, $b_j^S < 0$ and the flow in $S$ along the arc $\langle s, v_j \rangle$ is at least $e(v_j)$.*

(2) *For every $v_i \in V_s$, if $b_i^S < 0$ then the arc $\langle s, v_i \rangle$ is saturated, i.e., it carries $b_i^S$ flow units.*

**Proof.** For proving part 1 of the claim, consider a vertex $v_j \in V_t$. The number of flow units entering $v_j$ in $G(A,T)$ is greater by $e(v_j)$ than in $G(A,S)$. The extra flow units entering $v_j$ must be balanced in $T$. Thus $b_j^S < 0$, and the arc $\langle s, v_j \rangle$ carries at least $e(v_j)$ flow units. In this case, the flow can be balanced by reducing the number of flow units entering $v_j$ from the source. (Recall that we are dealing with flows $S$ and $T$ that saturate all the arcs entering each sink. Thus extra flow units cannot be balanced by increasing the number of flow units entering the sink. With that respect, recall that $b_j^T < b_j^S$.)

For proving part 2 of the claim, assume the contrary holds. Select an arbitrary path in $\mathscr{P}$, leading from $v_i$ to some vertex $v_j \in V_t$, and augment the flow through this path by 1. The balance at $v_i$ can be conserved by adding 1 to the flow entering $v_i$ through $s$. The balance through $v_j$ can also be conserved by decreasing the flow from $s$ to $v_j$ (see part 1). Therefore, there exists a legal assignment $\bar{Y}'$ where $hit(\bar{Y}^T, \bar{Y}') < hit(\bar{Y}^T, \bar{Y}^S)$. Moreover, as in the proof of Claim B.1 the net change in flow-cost is 0. This is contradiction. □

For $v \in V_s$, denote by $Path(v_i)$ the subset of $\mathscr{P}$ consisting of the $s(i)$ paths starting at $v_i$. Denote by $k(v_i)$ the additional flow units that enter $v_i$ in $T$ in comparison to $S$, that is, $k(v_i) = b_i^S - b_i^T$.

**Lemma A.5.** *For every $v_i \in V_s$, the number of paths $s(i)$ in $Path(v_i)$ satisfies $s(i) \leqslant k(v_i)$.*

**Proof.** Assume that we change the $\bar{Y}^S$ assignment to $\bar{Y}^T$ by iteratively augmenting the flow through the paths of $\mathscr{P}$. Note that whenever $v_i$ occurs as a middle vertex of some path during this process, the flow balance at $v_i$ remains unmodified by this augmentation. Furthermore, each of the $s(i)$ times that $v_i$ is a source, either the outgoing flow of $v_i$ is incremented by 1, or the incoming flow is decreased by 1. Thus in $\bar{Y}^T$, $s(i)$ additional flow units emanate from $v_i$. Those flow units cannot be balanced in $G(S,A)$ using the source $s$ (in the case $b_i^S < 0$) as indicated by part 2 of Claim A.4, as the arc $\langle s, v_i \rangle$ is saturated. Nor can they be balanced in $G(A,S)$ by reducing the flow from $v_i$ to the sink, since we are dealing with flows saturating all the arcs entering the sink. It follows that in $T$, additional flow units are available for $v_i$, namely, $k(i) = b_i^S - b_i^T \geqslant s(i)$, as required. □

We now complete the description of the decomposition. Assume w.l.o.g that $T \backslash S = \{x_1, \ldots, x_l\}$. Thus, adding $x_j$ to $S$ results in making $a_{ij}$ additional flow units available for $v_i$, where $a_{ij}$ is the coefficient of $x_j$ on the $i$th row of $A$. Clearly,

$$\sum_{j=1}^{n} a_{ij} = k(i) \tag{A.2}$$

and hence

$$s(i) \leqslant \sum_{j=1}^{n} a_{ij}. \tag{A.3}$$

Define for every $v_i$ a function $f_i : Path(v_i) \mapsto T \backslash S$ that arbitrarily assigns to every $x_j$ no more than $a_{ij}$ paths. (This is possible because of Lemma A.5 and Eq. (A.3).) Denote

$$\mathscr{P}(x_j) = \{ p \in \mathscr{P} \mid f_i(p) = x_j \text{ for some } i \}.$$

Note that if $x_j$ is added to $S$, one possibility for trying to reduce the flow is to make all the changes along the $|\mathscr{P}(x_j)|$ paths of $\mathscr{P}(x_j)$, since for every $i$, $a_{ij}$ additional flow units are available for $v_i$. It follows from part 1 of Claim A.4 that the flow can also be conserved in the respective vertices of $V_t$. Let $\Delta'(x_j)$ denote the total change in $f_4$ caused by these changes. Clearly,

$$\Delta(S, x_j) \geqslant \Delta'(x_j). \tag{A.4}$$

Furthermore,

$$\sum_j \Delta'(x_j) = f_3(T) - f_3(S), \tag{A.5}$$

since after the changes along all the paths of $\mathscr{P}$, the flow function $S$ is converted into the flow function $T$. Thus we conclude from Eqs. (A.4) and (A.5) that

$$\sum_j \Delta(S, x_j) \geqslant f_3(T) - f_3(S),$$

which establishes Axiom (II) for $f_3$. □

## References

[1] A. Bookstein, S.T. Klein, Construction of optimal graphs for bit-vector compression, Proc. 13th ACM SIGIR Conf. 1990, pp. 327–342.
[2] A. Bookstein, S.T. Klein, Compression of correlated bit-vectors, Inform. Systems 16 (1991) 387–400.
[3] J. Bar-Ilan, G. Kortsarz, D. Peleg, How to allocate network centers, J. Algorithms 15 (1993) 385–415.
[4] V. Chvatal, A greedy heuristic for the set-covering problem, Math. Oper. Res. 4 (1979) 233–235.
[5] P. Crescenzi, V. Kann, A compendium of NP optimization problems, Technical report, Royal Institute of Technology, Stockholm, Sweden, 1998. http://www.nada.kth.se/~viggo/problemlist/compendium.html.
[6] G. Dobson, Worst case analysis of greedy heuristics for integer programming with nonnegative data, Math. Oper. Res. 7 (1982) 515–531.
[7] U. Feige, A threshold of ln $n$ for approximating set cover, Proc. 28th ACM Symp. on Theory of Computing, 1996, pp. 314–318.
[8] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, San Fransisco, 1979.
[9] M. Gondran, M. Minoux, Graphs and Algorithms, Wiley, New York, 1984.
[10] D.S. Hochbaum, Heuristics for the fixed cost median problem, Math. Programming 22 (1982) 148–162.
[11] D.S. Hochbaum, D.B. Shmoys, A unified approach to approximation algorithms for bottleneck problems, J. ACM 33(3) (1986) 533–550.
[12] D.S. Johnson, Approximation algorithms for combinatorial problems, J. Comput. System Sci. 9 (1974) 256–278.
[13] G. Kortsarz, D. Peleg, Approximating shallow-light trees, Proc. 8th ACM-SIAM Symp. on Discrete Algorithms, 1997, pp. 173–182.
[14] M.R. Korupolu, C.G. Plaxton, R. Rajaraman, Analysis of local search heuristics for facility location problems, Proc. 10th ACM-SIAM Symp. on Discrete Algorithms, 1998, pp. 1–10.
[15] L. Lovász, On the ratio of integral and fractional covers, Discrete Math. 13 (1975) 383–390.
[16] C. Lund, M. Yannakakis, On the hardness of approximating minimization problems, J. ACM 41 (1994) 960–981.
[17] G.L. Nemhauser, L.A. Wolsey, Integer and Combinatorial Optimization, Wiley, New York, 1988.
[18] R. Raz, S. Safra, A sub constant error probability low degree test, and a sub constant error probability PCP characterization of NP, Proc. 29th ACM Symp. on Theory of Computing, 1997, pp. 475–484. 1997.
[19] D.B. Shmoys, E. Tardos, K. Aardal, Approximation algorithms for facility location problems, Proc. 29th Ann. ACM Symp. on Theory of Computing, 1997, pp. 265–274.
[20] L.A. Wolsey, An analysis of the greedy algorithm for the submodular set covering problem, Combinatorica 2 (1982) 385–393.