ELSEVIER

Information Processing Letters 80 (2001) 171–177

Information Processing Letters

www.elsevier.com/locate/ipl

Approximation algorithms for maximum linear arrangement

Refael Hassin *, Shlomi Rubinstein

Department of Statistics and Operations Research, School of Mathematical Sciences, Tel-Aviv University, Tel-Aviv 69978, Israel

Received 28 February 2000; received in revised form 7 February 2001
Communicated by L. Boasson

Abstract

The GENERALIZED MAXIMUM LINEAR ARRANGEMENT PROBLEM is to compute for a given vector $x \in \mathbb{R}^n$ and an $n \times n$ non-negative symmetric matrix $W = (w_{i,j})$, a permutation π of $\{1, \ldots, n\}$ that maximizes $\sum_{i,j} w_{\pi_i,\pi_j} |x_j - x_i|$. We present a fast $\frac{1}{3}$-approximation algorithm for the problem. We present a randomized approximation algorithm with a better performance guarantee for the special case where $x_i = i$, $i = 1, \ldots, n$. Finally, we introduce a $\frac{1}{2}$-approximation algorithm for MAX k-CUT WITH GIVEN SIZES OF PARTS. This matches the bound obtained by Ageev and Sviridenko, but without using linear programming.

Keywords: Algorithms; Algorithmical approximation

1. Introduction

We define the GENERALIZED LINEAR ARRANGEMENT PROBLEM as the problem of computing for a given vector $x = (x_1 \leqslant \cdots \leqslant x_n) \in \mathbb{R}^n$ of 'points' and an $n \times n$ non-negative symmetric matrix $w = (w_{i,j})$ of 'weights', a permutation π of $\{1, \ldots, n\}$ so that $\sum_{i,j} w_{\pi_i,\pi_j} |x_j - x_i|$ is optimized. In an illustrative example, consider n linearly ordered points in which a set of n machines is to be located, and $w_{i,j}$ is a measure of association of the ith and jth machines. Our interest is in the maximization version, the GENERALIZED MAXIMUM LINEAR ARRANGEMENT PROBLEM (GMLAP), where the goal is to *maximize* $\sum_{i,j} w_{\pi_i,\pi_j} |x_j - x_i|$, and keep the machines far from each other (compare with [10]).

The special (NP-hard) case in which $x_i = i$ is known as the LINEAR ARRANGEMENT PROBLEM and an $O(\log n)$-approximation for the *minimization* version is given in [9] (see also [6]). The minimization version is polynomially solvable if we are allowed to locate several elements (or none) at any point, even when some of the locations are already predetermined [8].

Another interesting (also, NP-hard) special case of the problem is MAX CUT PROBLEM WITH GIVEN SIZES OF PARTS where for some $p \leqslant n/2$, $x_1 = \cdots = x_p = 0$ and $x_{p+1} = \cdots = x_n = 1$. Ageev and Sviridenko [2] applied a novel method of rounding linear programming relaxations and developed a $\frac{1}{2}$-approximation algorithm for this problem. They also obtained a similar result for a more general MAX k-CUT PROBLEM in which integers $p_1, \ldots, p_k$ are given and the goal is to compute a k-cut, that is, a partition $S_1, \ldots, S_k$ of $\{1, \ldots, n\}$ with $|S_i| = p_i$, $i = 1, \ldots, k$,

* Corresponding author.
E-mail addresses: hassin@post.tau.ac.il (R. Hassin), shlomiru@post.tau.ac.il (S. Rubinstein).

0020-0190/01/$ – see front matter
PII: S0020-0190(01)00159-4

which maximizes the weight of pairs whose elements are in different parts of the partition.

The GMLAP is a special case of the MAXIMUM QUADRATIC ASSIGNMENT PROBLEM which also includes as special cases fundamental problems like the MAXIMUM TRAVELING SALESMAN PROBLEM. In this problem two $n \times n$ nonnegative symmetric matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ are given and the objective is to compute a permutation π of $\{1, \ldots, n\}$ so that

$$\sum_{\substack{i,j \in V \\ i \neq j}} a_{\pi(i),\pi(j)} b_{i,j}$$

is maximized. A $\frac{1}{4}$-approximation algorithm for this problem, under the assumption that the values of one of the matrices satisfy the triangle inequality, is given in [4]. Of course, this bound applies also to the GMLAP.

We present a $\frac{1}{3}$-approximation algorithm for the GMLAP. An interesting feature of our algorithm is that it simultaneously approximates the max cut problems with sizes p and $n - p$ of parts for all possible values of p. We present a randomized $\frac{1}{2}$-approximation for the MAXIMUM LINEAR ARRANGEMENT PROBLEM. Finally, we present an alternative $\frac{1}{2}$-approximation for MAX k-CUT PROBLEM WITH GIVEN SIZES OF PARTS. Unlike the algorithm of Ageev and Sviridenko, the latter algorithm does not use linear programming.

We first describe, in Section 2, a generic randomized approximation algorithm for MAX CUT WITH GIVEN SIZES OF PARTS. The analysis of this special case will be used in Section 3 where we obtain our main result on the GMLAP. In Section 4 we present our algorithm for MAXIMUM LINEAR ARRANGEMENT and in Section 5 we treat the MAX k-CUT PROBLEM WITH GIVEN SIZES OF PARTS.

For disjoint $S, T \subset V$ we mean by $\{i, j\} \in (S, T)$ that either $i \in S$ and $j \in T$ or $j \in S$ and $i \in T$. We denote

$$w(S, T) = \sum_{\{i,j\} \in (S,T)} w_{i,j},$$

and

$$w(i, T) = w\big(\{i\}, T\big).$$

We use $w(i, V)$ for $w(i, V \setminus \{i\})$. Finally, we denote by *opt* the optimal solution value in the problem under consideration.

2. MAX CUT WITH GIVEN SIZES OF PARTS

Given an undirected graph $G = (V, E)$ with $|V| = n$ and edge weights $w_{i,j}$, $\{i, j\} \in E$, a *cut* is a partition (S, T) of V, and its weight is $w(S, T)$. The problem is to compute a maximum weight cut such that $|S| = p$. Without loss of generality, we assume that $p \leqslant n/2$.

Theorem 1. *Let $\overline{w}(S, T)$ be the expected weight of the cut returned by Max_Cut (Fig. 1). Then, $\overline{w}(S, T) \geqslant opt/3$.*

Proof. The probability for an edge $\{i, j\} \in (P, V \setminus P)$ to be separated by (S, T) is $\frac{1}{2}$, since it corresponds to the event that i is selected to S. The probability for an edge $\{i, j\} \in P \times P$ to be separated by (S, T) is

$$\frac{p}{2p-1} > \frac{1}{2},$$

since it corresponds to the event that *exactly* one of i and j is selected. Consider an optimal solution and denote by OPT the set of size p in it. Let

$$r = |OPT \cap P|, \quad s = w(OPT \cap P, P \setminus OPT),$$

and

$$t = w\big(OPT \cap P, V \setminus (OPT \cup P)\big).$$

Max_Cut
input
1. A graph $G = (V, E)$, $V = \{1, \ldots, n\}$ with edge weights $w_{i,j}$, $\{i, j\} \in E$.
2. An integer $p \leqslant n/2$.

returns
A cut (S, T) such that $|S| = p$.
begin
$P := \{i_1, \ldots, i_{2p} \in V \mid w(i, V) \geqslant w(j, V) \quad \forall i \in P,\ j \notin P\}$.
Randomly choose p vertices from P to form S.
$T := V \setminus S$.
return (S, T).
end *Max_Cut*

Fig. 1. Algorithm *Max_Cut*.

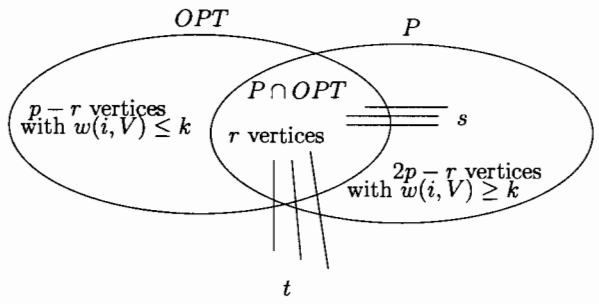

Fig. 2.

Note that by the definition of P, there is some threshold k such that $w(i, V)$ is at least k for $i \in P$ and at most k for $i \in V \setminus P$ (see Fig. 2). Also note that edges with two ends in P are counted twice in $s + \sum_{i \in P \setminus OPT} w(i, V)$ so that

$$\overline{w}(S, T) \geqslant \frac{s + \sum_{i \in P \setminus OPT} w(i, V)}{4} + \frac{t}{2} \geqslant \frac{s + (2p - r)k}{4} + \frac{t}{2} \equiv A,$$

and

$$opt \leqslant (p - r)k + t + s \equiv B.$$

We note that to compute a minimum possible value for the ratio A/B given that it can be made smaller than $\frac{1}{2}$, we can assume without loss of generality that $t = 0$. Let $s = \alpha(2p - r)k$. We now distinguish two cases.

Suppose first that $\alpha \leqslant 1$. We use

$$\overline{w}(S, T) \geqslant k(1 + \alpha)\frac{2p - r}{4}$$

and

$$opt \leqslant k\big((2p - r)(1 + \alpha) - p\big),$$

so that

$$\frac{\overline{w}(S, T)}{opt} \geqslant \frac{1}{4}\frac{(2p - r)(1 + \alpha)}{(2p - r)(1 + \alpha) - p}.$$

This ratio is monotone decreasing in α so that for the worst case we substitute $\alpha = 1$ and obtain

$$\frac{\overline{w}(S, T)}{opt} \geqslant \frac{2p - r}{2(3p - 2r)}.$$

This expression is monotone increasing in r and it is minimized when $r = 0$, in which case we obtain the ratio $\frac{1}{3}$.

begin
Sort V in non-increasing order of $w(i, V)$.
(For simplicity, suppose that
$w(1, V) \geqslant \cdots \geqslant w(n, V)$.)
for $i = 1, \ldots, p$
Assign $2i - 1$ to S, with probability $\frac{1}{2}$.
Assign $2i$ to S otherwise.
end for

Fig. 3. Modified *Max_Cut*.

Suppose now that $\alpha > 1$. We use the inequalities

$$\overline{w}(S, T) \geqslant \frac{s}{2},$$

(since the edges defining s are in $P \times P$) and

$$opt \leqslant (p - r)k + s \leqslant \frac{2p - r}{2}k + s = \frac{s}{2\alpha} + s \leqslant \frac{3}{2}s,$$

so that

$$\frac{\overline{w}(S, T)}{opt} \geqslant \frac{1}{3}. \qquad \square$$

The bound of Theorem 1 is asymptotically achievable: Let G consist of a star with $2p$ vertices, $p - 1$ vertex disjoint edges, and isolated vertices. The optimal solution is to choose the center of the star and one end of each edge to form a set of p vertices. Thus, $opt = 3p - 2$. We could form P from the $2p$ vertices of the star and then $\overline{w}(S, T) = p - \frac{1}{2}$. The corresponding ratio is asymptotically $\frac{1}{3}$.

Algorithm *Max_Cut* is fast and simple, but for our results in the next section we will use a variation of it, which is also easier for derandomization (as we describe in the next section). In this variation we change the main step of the algorithm as described in Fig. 3.

Theorem 2. *Let $\overline{w}(S, T)$ be the expected weight of the partition returned by Max_Cut with the modification given in Fig. 3. Then,*

$$\overline{w}(S, T) \geqslant \frac{opt}{3}.$$

Proof. The proof is as in Theorem 1, with $P = \{1, \ldots, 2p\}$. Note that the probability for an edge in $(P, V \setminus P)$ to be separated by (S, T) is $\frac{1}{2}$, as in Theorem 1. The probability for an edge $\{2i - 1, 2i\}$

for some $i = 1, \ldots, p$, to be separated by (S, T) is 1, and for other edges in $P \times P$ this probability is $\frac{1}{2}$. □

3. Generalized maximum linear arrangement

We start by presenting an alternative way to compute the weight of a solution π to GMLAP (a similar representation is used in [8]): For $p = 1, \ldots, n-1$ let

$$C_p = \sum_{i=1}^{p} \sum_{j=p+1}^{n} w_{\pi_i,\pi_j}.$$

Note that the problem of maximizing C_p over all permutations π of $\{1, \ldots, n\}$ is the max cut problem with sizes of parts p and $n-p$. Now we observe that

$$\sum_{i,j} w_{\pi_i,\pi_j} |x_j - x_i| = \sum_{p=1}^{n-1} C_p |x_{p+1} - x_p|. \tag{1}$$

In other words, the contribution of the interval $[x_p, x_{p+1}]$ to the weight of the solution is $C_p|x_{p+1} - x_p|$. Our algorithm *Randomized_GMLA* (Fig. 4) approximates *simultaneously* all of these cut problems with factor $\frac{1}{3}$ each, and consequently the same bound applies to the GMLAP instance as well. We note that *the permutation defining the approximate solution is independent of the vector* $(x_1, \ldots, x_n)$.

Theorem 3. *Let π be the permutation returned by Randomized_GMLA.*

(1) *Let $S_p = \{\pi_1, \ldots, \pi_p\}$, $T_p = \{\pi_{p+1}, \ldots, \pi_n\}$. Then (S_p, T_p) is a randomized $\frac{1}{3}$-approximation for the max cut problem with sizes of parts p and $n-p$.*

(2) *π is a randomized $\frac{1}{3}$-approximation for the GMLAP.*

Proof. By Theorem 2, for $p = 1, \ldots, n-1$, the value of C_p in the output of the algorithm is a randomized $\frac{1}{3}$-approximation for the respective max cut problem. The proof for the second part of the theorem follows now from (1). □

Derandomizing the algorithm is particularly simple. We apply the 'method of conditional expectations' (see [3]). In the first iteration we assign $\pi_1 := 1$ and $\pi_n := 2$. Consider the ith iteration for $i > 1$. Let S_{i-1} and T_{i-1} be the partial permutation that has already been set in the first $i-1$ iterations. We should set π_i to either $2i-1$ or to $2i$, and π_{n-i+1} to the other value. This is done so that the expected value of the solution is maximized given S_i and T_i and assuming that the following assignments will be done according to *Randomized_GMLA*. We observe that the expected weight gained by the latter assignments is independent of the current decision. Therefore, the current assignment should be done in a way that maximizes the weight of edges connecting the vertices $2i-1$ and $2i$ to the previously assigned vertices. It

Randomized_GMLA
input A non-negative symmetric matrix $W = (w_{i,j},\ i, j = 1, \ldots, n)$.
returns A permutation $\pi_1, \ldots, \pi_n$ of $V = \{1, \ldots, n\}$.
begin
Sort V in non-increasing order of $w(i, V)$.
(For simplicity, suppose that $w(1, V) \geqslant \cdots \geqslant w(n, V)$.)
for $i = 1, \ldots, \lfloor n/2 \rfloor$
  Set $\pi_i := 2i - 1$ and $\pi_{n-i+1} := 2i$ with probability $\frac{1}{2}$.
  Set $\pi_i := 2i$ and $\pi_{n-i+1} := 2i - 1$ otherwise.
  If n is odd, set $\pi_{(n+1)/2} := n$.
end for
return π.
end *Randomized_GMLA*

Fig. 4. Algorithm *Randomized_GMLA*.

is easy to see that under this rule the algorithm sets $\pi_i := 2i - 1$ and $\pi_{n-i+1} := 2i$ if

$$w(2i-1, S_{i-1}) + w(2i, T_{i-1}) \leqslant w(2i, S_{i-1}) + w(2i-1, T_{i-1}).$$

It sets $\pi_i := 2i$ and $\pi_{n-i+1} := 2i - 1$ otherwise. We call the resulting Algorithm *GMLA*.

Theorem 4. *Let* $m = |\{\{i, j\}: w_{i,j} > 0\}|$. *Then, Algorithm GMLA computes a* $\frac{1}{3}$*-approximation for the GMLAP and for* MAX CUT WITH GIVEN SIZES OF PARTS *for every* $p = 1, \ldots, \lfloor n/2 \rfloor$ *in time* $O(m + n \log n)$.

4. MAXIMUM LINEAR ARRANGEMENT

Recall that the MAXIMUM LINEAR ARRANGEMENT PROBLEM is the special case of GMLAP where $x_i = i$, $i = 1, \ldots, n$. There are simple approximation algorithm with constant performance guarantees for this problem. The simplest one relies on the observation that the average value of $|x_j - x_i|$ is $n/3$. Therefore, the expected weight of a random permutation is $(n/3)\sum_{i,j} w_{\pi_i,\pi_j}$ which is at least $\frac{1}{3}opt$.

A better bound can be obtained as follows. Let (S, T) be a partition which maximizes

$$mc = \sum_{\{i,j\} \in (S,T)} w_{i,j}.$$

Let $\pi_1, \ldots, \pi_{|S|}$ be a random permutation of S and let $\pi_{|S|+1}, \ldots, \pi_n$ be a random permutation of T. (Alternatively, order the elements $i \in S$ in decreasing order of $\sum_{j \in T} w_{i,j}$ and order the elements $j \in T$ in increasing order of $\sum_{i \in S} w_{i,j}$.) Let *apx* be the weight of the resulting solution. *opt* is equal to a sum of values of a series of n cuts, each bounded by *mc*. Thus $opt \leqslant n \cdot mc$. The average 'distance' between an element in S and in T is $n/2$. Thus,

$$apx = \frac{n}{2} mc \geqslant \frac{1}{2} opt.$$

We can use the 0.878-approximation algorithm for MAX CUT [7] to obtain a 0.439-approximation for our problem.

We now present a randomized algorithm with a better performance guarantee (see Fig. 5).

Lemma 5. *Consider an optimal solution. The total contribution to the solution's weight which comes from pairs of elements that were placed within distance of at most* $n^{0.8}$ *from each other is less than* $\frac{1}{n^{0.1}} opt$.

Proof. In a random solution, the expected weight between any pair of elements is $n/3$. If the total contribution to the solution's weight which comes from pairs of elements that were placed within distance of at most $n^{0.8}$ from each other is greater than $\frac{1}{n^{0.1}} opt$, then just the expected contribution of these elements to the weight of a random solution would be at least

$$\frac{n/3}{n^{0.8}} \frac{1}{n^{0.1}} opt > opt,$$

a contradiction. □

Max_LA
input A non-negative symmetric matrix $W = (w_{i,j},\ i, j = 1, \ldots, n)$.
returns A permutation π of $V = \{1, \ldots, n\}$.
begin
Partition V into subsets S, T by independently assigning each $i \in V$ with probability $\frac{1}{2}$ to S or T.
Sort S in non-increasing order of $W_i = \sum_{j \in T} w_{i,j}$ and let $\pi_1, \ldots, \pi_{|S|}$ be the corresponding order.
Sort T in non-decreasing order of $W'_i = \sum_{j \in S} w_{i,j}$ and let $\pi_{|S|+1}, \ldots, \pi_n$ be the corresponding order.
return π.
end *Max_LA*

Fig. 5. Algorithm *Max_LA*.

Observation 6. *Let S, T be a random partition of $\{1, \ldots, n\}$, as in Max_LA (Fig. 5). The expected contribution to opt coming from pairs of elements that belong to different parts of the partition is opt/2.*

Observation 7. *The permutation π computed by Max_LA maximizes the contribution to the total weight coming from pairs of elements that belong to different parts of the partition S, T under the constraint that the elements of S are assigned to $1, \ldots, |S|$ and those of T to $|S| + 1, \ldots, n$.*

Lemma 8. *With probability that exponentially approaches 1 as a function of n, it is possible to assign S to $1, \ldots, |S|$ and T to $|S| + 1, \ldots, n$, so that the distance of each element from its relevant end (1 for S or n for T) differs from its distance in opt from the nearest end, by at most $n^{0.6}$.*

Proof. Index the elements according to their distance in *opt* from their nearest end, breaking ties arbitrarily. Assign S to $1, \ldots, |S|$ and T to $n, n-1, \ldots, |S|+1$ in increasing order of the indices. Consider an element with index k. Its distance from the nearest end in *opt* is $\lfloor (k-1)/2 \rfloor$. Its distance under the above assignment depends on the number of lower index elements that were assigned to its part (S or T). Thus, this distance is a random variable with binomial distribution $B(k-1, \frac{1}{2})$. Its mean is $(k-1)/2$ and standard deviation $\frac{1}{2}\sqrt{k-1}$. Therefore, using the Normal approximation of the Binomial distribution, the probability of a deviation of at least $n^{0.6}$ from the mean is of order $e^{-n^{1.2}/k}$ which is less than $e^{-n^{0.2}}$ since $k \leqslant n$. The probability that such a deviation will be obtained for any of the n elements is bounded by $ne^{-n^{0.2}}$. □

Theorem 9. *Algorithm Max_LA (Fig. 5) produces, with probability that approaches 1 as a function of n, a solution of weight $1 - o(1)$ times the contribution to opt which comes from elements that belong to different parts of the partition (S, T).*

Proof. By Lemma 8, the distance between any two points in the solution produced by *Max_LA* is at least their distance in the optimal solution minus $O(n^{0.6})$. By Lemma 5, the relative contribution of the pairs that are closer than $n^{0.8}$ is asymptotically 0. For elements whose distance in *opt* is at least $n^{0.8}$, a relative deviation of $n^{0.6}/n^{0.8}$ is asymptotically zero. Therefore, by Observation 7 and Lemma 8, the algorithm guarantees almost all of the weight that the optimal solution has between S and T. □

Theorem 10. *The expected weight of the solution produced by Max_LA is asymptotically $\frac{1}{2}$opt, with probability that approaches 1 as a function of n.*

Proof. Follows from Theorem 9 and Observation 7. □

5. MAX k-CUT WITH GIVEN SIZES OF PARTS

Given a graph $G = (V, E)$ with edge weights w and integers $p_1, \ldots, p_k$ such that $\sum p_i = n$, the MAX k-CUT WITH GIVEN SIZES OF PARTS is to compute a k-cut, that is, a partition $S_1, \ldots, S_k$ of V, such that $|S_i| = p_i$, $i = 1, \ldots, k$, maximizing the weight of edges whose ends are in different parts of the partition.

A vertex $v \in V$ is said to *cover* the weight of the edges $\{\{u, v\} \in E\}$. A subset $V' \subset V$ covers the weight of the union of edges which have at least one end in it. Bar-Yehuda [5] developed an $O(n^2)$ 2-approximation algorithm for the following problem: Given w, compute a vertex set of minimum size that covers edge weight of size at least w.

One can obtain from this result, in a straightforward way, a solution to the following problem: Given $p \leqslant \frac{1}{2}n$ find a set S' of $2p$ vertices that covers edge weight of at least $w(p)$, where $w(p)$ is the maximum edge weight that can be covered by p vertices. To achieve this goal we apply binary search over $[0, w(E)]$, where $w(E)$ is the total weight of E. For each test value, w, we apply Bar-Yehuda's algorithm and we stop with the highest value for which the algorithm returns a set S' with at most $2p$ vertices. If the set contains less than $2p$ vertices we extend it by adding arbitrary vertices. The complexity of this procedure is $O(n^2 \log w(E))$.

Our algorithm for the case of $k = 2$ (MAX CUT WITH GIVEN SIZES OF PARTS) proceeds as follows: Randomly select p vertices from S' and move them to the other side of the cut. Let the resulting set be S_A. We claim that the expected size of the cut $(S_A, V \setminus S_A)$ is a $\frac{1}{2}$-approximation for the problem. The argument is that the weight of the edges covered

by S' is an upper bound on the optimal solution value, and each of these edges will be in the cut with probability $\frac{1}{2}$. The algorithm can be derandomized by applying the 'method of conditional expectations' (see [3]). However, the rounding method of Ageev and Sviridenko can also be used to obtain a $\frac{1}{2}$-approximation in deterministic linear time, once S' is given [1].

Ageev and Sviridenko [2] also applied their method to obtain a $\frac{1}{2}$-approximation for the MAX k-CUT PROBLEM WITH GIVEN PARTS.

Our algorithm can be modified for the max k-cut problem as well: We first observe that if $p_i \leqslant \frac{1}{2}n$ for every $i = 1, \ldots, k$ then the cut contains more than half of the edges so that a random solution has expected weight of at least half the total weight of the graph. Thus a random solution suffices to obtain a $\frac{1}{2}$-approximation.

Assume now that $p_1 > n/2$. We compute as above sets S' and S_A with $p = n - p_1$. We set $P_1 = V \setminus S_A$ and arbitrarily partition S_A to form $P_2, \ldots, P_k$. The resulting k-cut has the property that the expected weight of edges between P_1 and the other parts is already half the weight of the edges covered by S' which is itself an upper bound on the optimal solution. Thus the k-cut we constructed is a $\frac{1}{2}$-approximation for the problem. Again, the algorithm can be derandomized.

References

[1] A.A. Ageev, Personal communication.

[2] A.A. Ageev, M.I. Sviridenko, Approximation algorithms for maximum coverage and max cut with given sizes of parts, in: Proceedings of IPCO'99, Lecture Notes in Computer Science, Vol. 1610, Springer, Berlin, 1999, pp. 17–30.

[3] N. Alon, J.H. Spencer, The Probabilistic Method, John Wiley and Sons, New York, 1992.

[4] E. Arkin, R. Hassin, M.I. Sviridenko, Approximating the maximum quadratic assignment problem, Inform. Process. Lett. 77 (2001) 13–16. A preliminary version appeared in: Proc. 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA2000), 2000, pp. 889–890.

[5] R. Bar-Yehuda, Using homogeneous weights for approximating the partial cover problem, J. Algorithms 39 (2001) 137–144. A preliminary version appeared in: Proc. 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA99), 1999, pp. 71–75.

[6] G. Even, J. Naor, S. Rao, B. Schieber, Divide-and-conquer approximation algorithms via spreading matrices, J. ACM 47 (2000) 585–616.

[7] M.X. Goemans, D.P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, J. ACM 42 (1995) 1115–1145.

[8] J.-C. Picard, H.D. Ratliff, A cut approach to the rectilinear distance facility location problem, Oper. Res. 26 (1978) 422–433.

[9] S. Rao, A.W. Richa, New approximation techniques for some ordering problems, in: Proc. 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA98), 1998, pp. 211–218.

[10] A. Tamir, Obnoxious facility location on graphs, SIAM J. Discrete Math. 4 (1991) 550–567.