Solving Minimum K-Cardinality Cut Problems in Planar Graphs

Maurizio Bruglieri, Francesco Maffioli
Dipartimento di Elettronica e Informazione, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy

Marco Trubian
Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Via Comelico 39, 20100 Milano, Italy

The present work tackles a recent problem in the class of cardinality constrained combinatorial optimization problems for the planar graph case: the minimum *k*-cardinality cut problem. Given an undirected edge-weighted connected graph the min *k*-cardinality cut problem consists in finding a partition of the vertex set *V* in two sets V_1, V_2 such that the number of the edges between V_1 and V_2 is exactly *k* and the sum of the weights of these edges is minimal. Although for general graphs the problem is already strongly $\mathcal{NP}$-hard, we have found a pseudopolynomial algorithm for the planar graph case. This algorithm is based on the fact that the min *k*-cardinality cut problem in the original graph is equivalent to a bi-weighted exact perfect matching problem in a suitable transformation of the geometric dual graph. Because the Lagrangian relaxation of cardinality constraint yields a max cut problem and max cut is polynomially solvable in planar graphs, we also develop a Lagrangian heuristic for the min *k*-cardinality cut in planar graphs. We compare the performance of this heuristic with the performance of a more general heuristic based on a Semidefinite Programming relaxation and on the Goemans and Williamson's random hyperplane technique. NETWORKS, Vol. 48(4), 195–208 2006

Keywords: cut problems; minimum *k*-cardinality cut; Lagrangian relaxation; exact perfect matching; planar graphs; semidefinite programming

1. INTRODUCTION

In recent years a number of articles have been published in which classical combinatorial optimization problems have been modified by imposing an additional cardinality constraint, that is, feasible solutions are constrained to contain a given number k of elements. Applications of cardinality constrained tree problems are in oil-field leasing [13] and facilities layout [15]. Chang et al. [9] deals with portfolio optimization, when the portfolio has to contain a fixed number of assets. A number of other problems, for example, the assignment problem [11], have also been studied under cardinality constraints. A survey on the topic with extensive references is available [14]. A class of combinatorial optimization problems that have applications in a wide variety of areas are cut problems, that is, the problems to find in a given graph a cut of maximal (or minimal) weight. In physics, for example, the maximum cut problem models the problem of finding a ground state of spin glasses having zero magnetization. In VLSI design, it models the problem of minimizing the number of vias (holes on a printed circuit board, or contacts on a chip), see [2]. In numerical analysis it is helpful in finding the L-U factorization of the matrix of a linear system. The minimum cut problem has applications, for example in network reliability theory and in compilers for parallel languages. The minimum cut problem with the addition of a cardinality constraint has been considered, at our knowledge, only in the case of oriented graphs; see for example, the paper by Cong and Y. Ding [10], where an application in the VLSI area is presented. Hence, we set out to investigate cardinality constrained cut problems (k cardinality cut problems) in the case of undirected graphs, because it has not yet been considered. This article contains the results of our research for what concerns connected planar graphs. We start with some basic definitions. Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. In the whole paper we let be $|V| = n$ and $|E| = m$ when it is not differently specified.

Definition 1.

1. *A* cut *is a partition of vertex set V in two sets V_1, V_2 called the* shores *of the cut. A* cut edge set $C := \{\{v_1, v_2\} \in E : v_1 \in V_1, v_2 \in V_2\}$ *is associated with every cut.*

Received November 2004; accepted June 2006
Correspondence to: M. Bruglieri; e-mail: bruglier@elet.polimi.it
DOI 10.1002/net.20129
Published online in Wiley InterScience (www.interscience.wiley.com).

TABLE 1. Complexity results for min k-cardinality cut.

Graph class	Unweighted	Weighted
General	strongly $\mathcal{NP}$-complete	strongly $\mathcal{NP}$-hard
Complete	$\mathcal{P}$	strongly $\mathcal{NP}$-hard
Complete bipartite	$\mathcal{P}$	strongly $\mathcal{NP}$-hard
Tree	$\mathcal{P}$	$\mathcal{P}$

2. *Given $s, t \in V$ an s-t* cut *is a cut (V_1, V_2) such that $s \in V_1$ and $t \in V_2$.*

Because from the cut edge set C one can easily reconstruct the shores V_1, V_2, in the sequel we shall indifferently define cuts either through the shores or through the cut edge set. Let us introduce the notation $\delta(A,B)$ and $\delta(A)$ for $A, B \subset V$ as follows:

$$\delta(A,B) := \{\{v_1, v_2\} \in E : v_1 \in A, v_2 \in B\}$$
$$\delta(A) := \delta(A, \bar{A}),$$

where $\bar{A}$ denotes $V \backslash A$. Let w: $E \to \mathbb{N}$ be a positive integer function on the edge set of graph G. The minimum cut problem (*min cut*) and the maximum cut problem (*max cut*) are the problems to find a cut such that the sum of the weights of the cut edge set C is minimal and maximal, respectively. It will be convenient, to denote the weight of any subset of edges $F \subset E$ by $w(F) := \sum_{e \in F} w(e)$. We can now introduce cardinality constrained cut problems. Let k be a positive integer.

Definition 2. *The minimum k-cardinality cut problem (min k-cardinality cut) is the problem to find a cut such that the cut edge set C has cardinality k and the sum of the weights of the edges belonging to C is minimal.*

In Section 1.1 below we shall present complexity results for a number of important graph classes. The rest of the paper is organized as follows. In Section 2 we present some considerations on the polyhedral structure of the problem. In Section 3 we show that the existence version of the k-cardinality cut is in $\mathcal{P}$ and we use this result, in the following section, to present a DualGreedy approximated Algorithm. In Section 5 we show how the problem can be solved with a pseudopolynomial time algorithm and with a random pseudopolynomial one. In Section 6, we present the Lagrangian relaxation of min k-cardinality cut. We develop an efficient method to solve the Lagrangian dual to find the best Lagrangian lower bound. We show that in correspondence to the optimal Lagrangian multiplier either we obtain an optimal solution of min k-cardinality cut or we obtain two cuts having cardinality k_1 and k_2 with $k_1 < k < k_2$ such that they are optimal solutions of min k_1-cardinality cut and min k_2-cardinality cut, respectively. In the last section we compare through numerical results the performance of Lagrangian relaxation with that of the Semidefinite Programming relaxation presented in [6].

1.1. Complexity of Minimum k-Cardinality Cut Problem

The computational complexity of min k-cardinality cut problems has been investigated for the first time in [6]. For general unweighted graphs the existence version of this problem is already strongly $\mathcal{NP}$-complete, because the largest k value for which min k-cardinality cut is feasible represents the solution of the simple max cut problem, which is strongly $\mathcal{NP}$-complete (see p. 210 of [18]). We notice that without the cardinality constraint the min k-cardinality cut problem reduces to the min cut problem which can be efficiently solved through several algorithms (see, e.g., [19, 27, 28].

Despite this result, there are some special cases for which the min k-cardinality cut problem can be solved in polynomial time. One is the unweighted problem, when G is a complete graph. Then there exists a cut containing k edges if and only if $k = i(n-i)$ for some $i \in \left\{1, \ldots, \lfloor \frac{n}{2} \rfloor\right\}$. Another case is G being a tree. Because every subset of the edge set is a cut, one might simply select the edges with the k smallest weights and define a cut appropriately, to solve even the weighted min k cardinality cut problem. We summarize in Table 1 the complexity results obtained in [6] for min k-cardinality cut, where the column "Unweighted" corresponds to the existence version of the problem.

We notice that for planar graphs the complexity of min k-cardinality cut cannot be determined through the reduction from max cut like in the general case, because for planar graphs the max cut is polynomially solvable ([1]). The aim of the present work is to discuss the problem in the planar graph case. As we will show for planar graphs the complexity is given in Table 2, where $p\mathcal{P}$ denotes the class of decision problems that admit a pseudopolynomial algorithm.

2. SOME CONSIDERATIONS ON THE POLYHEDRAL STRUCTURE OF *k*-CARDINALITY CUT IN PLANAR GRAPHS

Considering the relation between the max cut problem and min k-cardinality cut and the fact that max cut is polynomial for planar graphs, established by Theorem 6.4 of [1], which we report below, it is interesting to consider the polyhedral structure of both problems for planar graphs.

TABLE 2. Complexity results for planar min k-cardinality cut.

Graph class	Unweighted	Weighted
Planar	$\mathcal{P}$	$p\mathcal{P}$

Theorem 1 (Theorem 6.4 of [6]).
Let

$$T(G) := \{x \in \mathbb{R}^{\frac{n(n-1)}{2}} : x_{ij} + x_{il} + x_{jl} \leq 2$$
$$x_{ij} - x_{il} - x_{jl} \leq 0$$
$$-x_{ij} + x_{il} - x_{jl} \leq 0$$
$$-x_{ij} - x_{il} + x_{jl} \leq 0$$
$$\text{for all } 0 \leq i < j < l \leq n\}$$

where

$$x_{ij} = \begin{cases} 1 & \textit{if edge \{i,j\} belongs to the cut,} \\ 0 & \textit{otherwise.} \end{cases}$$

for all $0 \leq i < j \leq n$,

$$\begin{aligned} &\text{maximize} \sum_{e \in E} w(e) x_e \\ &\text{subject to } x \text{ satisfies } T(G) \end{aligned} \tag{1}$$

is the value of a max cut of G if and only if G is not contractible to K_5.

This theorem shows that the max cut problem is solvable in polynomial time for the class of graphs noncontractible to K_5 because it can be formulated as the linear program (1) with $4\binom{n}{3}$ constraints. Finally, because by Kuratowski's theorem (see Theorem 4.5 of [7]), planar graphs are those graphs that are not contractible to K_5 or $K_{3,3}$, the previous result also holds for planar graphs.

Now let $KCUT(G,k)$ denote the convex hull of all incidence vectors of min k-cardinality cut, that is,

$$KCUT(G,k) := \text{conv}\left\{x \in \{0,1\}^{|E|} : x \text{ is the incidence vector of a cut and } \sum_{e \in E} x_e = k\right\}.$$

Therefore,

$$KCUT(G,k) \subset CUT(G) \cap \left\{x \in [0,1]^{|E|} : \sum_{e \in E} x_e = k\right\}. \tag{2}$$

where $CUT(G)$ denotes the cut polytope of G. If the opposite inclusion held, too, we could conclude that in (2) the equality holds and so we may try to exploit Theorem 1 also to solve min k-cardinality cut in polynomial time. But, unfortunately, the opposite inclusion does not hold in (2) as the example below shows. For the graph G drawn in Figure 1, $KCUT(G,3) = \emptyset$ because this graph has only cuts with cardinality 2 or 4. But $CUT(G) \cap \{x \in [0,1]^{|E|} : \sum_{e \in E} x_e = k\} \neq \emptyset$ because, for example, $\tilde{x} = (1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2})$ belongs to this set. Indeed, $\tilde{x} \in CUT(G)$ because $\tilde{x} = \frac{1}{2}x' + \frac{1}{2}x''$ where $x' = (1,0,1,1,0,1)$ and $x'' = (1,1,0,0,0,0)$ are incidence vectors of cuts of G. Moreover, $\sum_{e \in E} \tilde{x}_e = 3$. Examples of grid graphs and triangulations for which the equality does not hold can also be easily constructed.

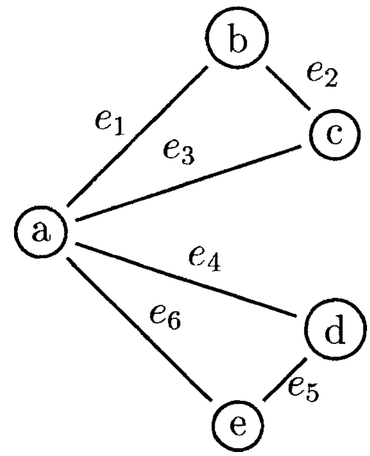

FIG. 1. Planar graph G.

3. THE EXISTENCE VERSION OF PLANAR *k*-CARDINALITY CUT

In this section we face the problem of determining if a given planar graph contains a k-cardinality cut. As a first result we prove the following proposition:

Proposition 1. *Given a planar graph* $G = (V,E)$ *with* $|V| = n$, $|E| = m$, *every k-cardinality cut of G is equivalent to an exact perfect matching of weight k in a suitable planar graph* $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ *with* $|\widetilde{V}| = 12(n-2)$, $|\widetilde{E}| = 15(n-2)$.

Proof. Let us consider a planar embedding of graph G. If the graph owns some faces, also considering the external one, with more than three edges we transform them in triangular faces adding suitable dummy edges. Let $\overline{E}$ the new set of edges. From Euler's formula for planar graphs ($n - m + r = 2$, where r is the number of the faces) one can derive that $|\overline{E}| = 3(n-2)$ and the number of triangular faces is $2(n-2)$. In the graph $\overline{G} = (V, \overline{E})$ we define the edge-weight function w' in the following way:

$$w'(e) = \begin{cases} 1 & \text{if } e \in E, \\ 0 & \text{if } e \in \overline{E} \backslash E, \end{cases}$$

Because the graph $\overline{G}$ is planar it has an associated geometric dual graph $G^* = (V^*, E^*)$ built according to the following rules: associate a vertex $v_i^* \in V^*$ to each face F_i of $\overline{G}$, associate a vertex $v_0^* \in V^*$ to the extern of $\overline{G}$, and associate to each edge $e_i \in \overline{E}$ an edge $e_i^* = \{u_i^*, v_i^*\} \in E^*$ whose vertices correspond to the two faces separated by the edge e_i (see Fig. 2). We define on E^* the weight function $w^*(e^*) = w'(e)$, $\forall e^* \in E^*$, where e is the edge of $\overline{G}$ corresponding to e^*. As a consequence of Theorem 4 and Theorem 5 of [26], every k-cardinality cut of graph G corresponds to an Eulerian subgraph of G^*, that is, a subgraph where each vertex has even degree, having weight k with respect to the weight function w^*. Let $\widetilde{G}$ be the planar graph, with edge-weight function $\tilde{w}$, obtained from G^* in the following way. Replace each vertex $v_i^* \in V^*$ with a cycle C_i having

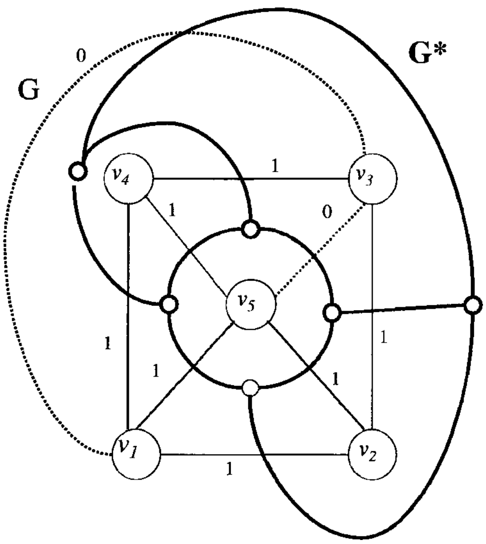

FIG. 2. The planar graph G and its dual G^*.

length (number of vertices and edges) equal to the degree of v_i^* and substitute every edge $e_i^* \in E^*$ with a chain CH_i of three edges. In each of these chains, set the weight of the central edge equal to $w^*(e_i^*)$ and the weights of all other edges equal to 0 (see Fig. 3). Hence, the total number of vertices of the exploded graph is three times the number of triangular faces and twice $|\overline{E}|$, i.e. $|\widetilde{V}| = 12(n-2)$; the total number of edges is three times the number of triangular faces and three times $|\overline{E}|$, that is, $|\widetilde{E}| = 15(n-2)$.

Now we show that the problem of finding an Eulerian subgraph with weight k in G^* is equivalent to find an exact perfect matching with weight k in $\widetilde{G}$. The edges in every perfect matching M of $\widetilde{G}$ can be partitioned into three sets M_1, M_2 and M_3. The set M_1 contains the edges belonging to cycles C_i, M_2 contains the edges that are central in chains CH_i, whereas M_3 contains the edges that are not central in chains CH_i. First, we show that from any Eulerian subgraph of G^* with edge set $\widehat{E}$, once we put in M_2 the central edges of the chains CH_i corresponding to the edges of $\widehat{E}$ and we put in M_3 the not central edges of the chains CH_i corresponding to the edges of $E^* \backslash \widehat{E}$, there exists an unique way to build an exact perfect matching $M = M_1 \cup M_2 \cup M_3$ of $\widetilde{G}$. Vice versa, we show that the edges in E^* corresponding to the edges in M_2 made up an Eulerian subgraph of G^*. We observe that because all the faces of $\overline{G}$ are triangular, every vertex $v_i^* \in V^*$ has a degree of 3; hence, every corresponding cycle C_i has length of exactly 3. Because any vertex in an Eulerian subgraph has a degree of 2, then any Eulerian subgraph of G^* is composed by vertex disjoint cycles. If a vertex $v_i^* \in V^*$ does not belong to an Eulerian subgraph [see Fig. 4(a)], then the vertices in $\widetilde{G}$ belonging to the set S containing the vertices of cycle C_i and the vertices in the chains corresponding to the edges incident to v_i^* are covered by the perfect matching M' [see Fig. 4(b)]. As in G^*, the edges incident to v_i^* does not belong to the Eulerian subgraph, then in $\widetilde{G}$ the central edges in the chains corresponding to the edges incident to v_i^* do not belong to the perfect matching M'. Whereas, if the same vertex v_i^* belongs to an Eulerian subgraph [see Fig. 5(a)], then in $\widetilde{G}$ the vertices of S, a part two of them on the border, are covered by the perfect matching M'' [see Fig. 5(b)]. As in G^*, two of the edges incident to v_i^* belong to the Eulerian subgraph; then in $\widetilde{G}$ the central edges in the corresponding chains belong to the perfect matching M'', while the central edge of the other chain does not. In this second case a similar reasoning on vertices v_j^* and v_l^*, which must have two edges belonging to the Eulerian subgraph, will take into account for the two uncovered vertices on the border. Vice versa, every perfect matching $M = M_1 \cup M_2 \cup M_3$ of $\widetilde{G}$ will cover every subset of vertices as S only in one of the two ways depicted above. As a consequence, the edges in M_2 univocally determine in G^* an Eulerian subgraph. Because the edges in $M_1 \cup M_3$ have weight equal to zero, while the edges in M_2 have weight equal to the corresponding edges in $\widehat{E}$, then the total weight of a perfect matching M is equal to the weight $w(\widehat{E})$, and vice versa. ∎

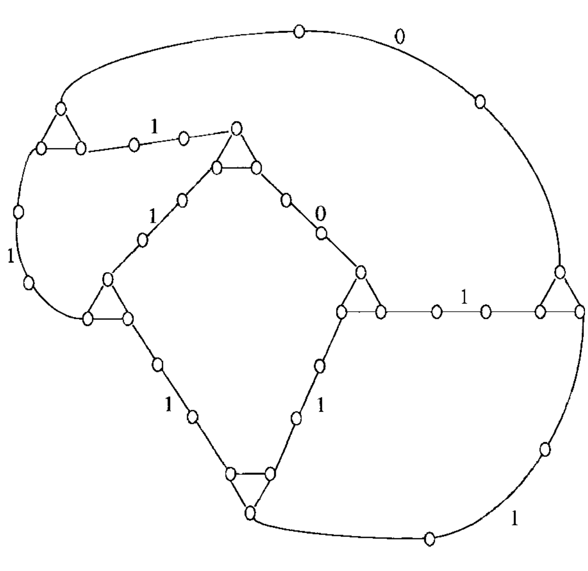

FIG. 3. The "exploded" graph $\widetilde{G}$ obtained from G^*.

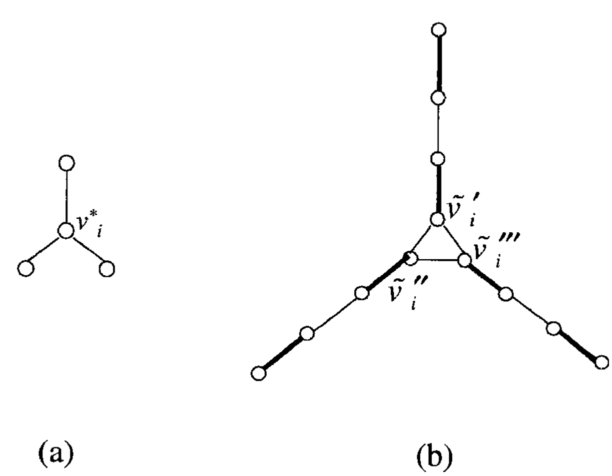

FIG. 4. Case of a vertex v_i^* not belonging to the Eulerian subgraph.

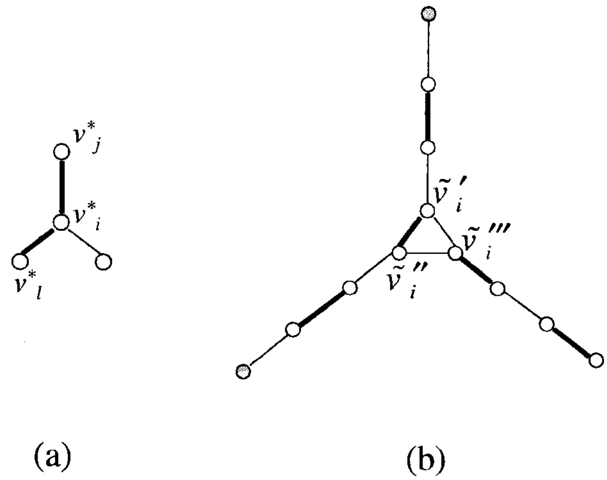

FIG. 5. Case of a vertex v_i^* belonging to the Eulerian subgraph.

Proposition 2. *When $G = (V, E)$ is a planar graph the existence version of the k-cardinality cut problem is in $\mathcal{P}$ with time complexity $O(n^5 logn)$.*

Proof. We know from Proposition 1 that the existence version of the k-cardinality cut problem in G is equivalent to the existence version of a perfect matching of weight k in a suitable planar graph $\widetilde{G}$ whose number of vertices is $O(n)$. In [3], the problem of finding a perfect matching of a given weight is solved with time complexity $O((n^3 + p^2)p^2(logp))$, where $p = nU$ and $U = \max\{w(e)\}$. Because in $\widetilde{G}$ the maximum weight of an edge is 1, then $p = n$, which completes the proof. ∎

Hereafter, we prove that the above time complexity can be improved at the expense of using a randomized algorithm.

Definition 3. *A random (pseudo-)polynomial algorithm for a decision problem is an algorithm that always answers correctly in the case of a no-instance, whereas for a yes-instance the answer may be wrong, with probability less than a positive constant $\varepsilon < 1$ independent of the input size. Furthermore, it requires a time that is (pseudo-)polynomial in the input size.*

Proposition 3. *When $G = (V, E)$ is a planar graph the existence version of the k-cardinality cut problem can be solved with a random polynomial algorithm with time complexity $O(n^4)$.*

Proof. We know from Proposition 1 that the existence version of the k-cardinality cut problem in G is equivalent to the existence version of a perfect matching of weight k in a suitable planar graph $\widetilde{G}$ whose number of vertices is $O(n)$. Theorem 8 of [17] proves that the problem of finding a perfect matching of a given weight can be solved using a randomized algorithm that works in a finite field with time complexity $O(pn^2 \log(U + 1) + pn^3)$, where $p = nU$ and $U = \max\{w(e)\}$. Because in $\widetilde{G}$, $U = 1$, the overall complexity is $O(n^4)$. ∎

4. A (RANDOM) PSEUDOPOLYNOMIAL ALGORITHM FOR PLANAR *k*-CARDINALITY CUT

To solve or approximate the solution of the min k-cardinality cut problem, that is, to find a cut such that the cut edge set has cardinality k and the sum of the weights of the edges belonging to it is minimal, we propose three techniques: a random pseudopolynomial algorithm, a dual greedy procedure, and a Lagrangian heuristic. We apply these techniques once the existence of a feasible solution has been proved as described in the previous section. In this section we describe the random pseudopolynomial algorithm, whereas the other two methods will be described in the next two ones. Now we need to introduce a set of definitions that also will be used in the next section.

Given a skew-symmetric matrix M of even order the *Pfaffian* of M, $pf(M)$, is defined as

$$pf(M) := \sqrt{det(M)}$$

where $det(M)$ denotes the determinant of M. Given a graph $G = (V, E)$ with vertex set $V = \{1, \ldots, 2n\}$, edge set E, $|E| = m$ and an edge-weight function w: $E \to \mathbb{N}$, let us consider the $2n \times 2n$ skew-symmetric matrix C

$$C_{i,l} = \begin{cases} t_{i,l}y^{w(e)} & \text{if } e = \{i, l\} \in E, i < l, \\ -t_{i,l}y^{w(e)} & \text{if } e = \{i, l\} \in E, i > l, \\ 0 & \text{otherwise.} \end{cases}$$

Because the determinant of a skew-symmetric matrix of even order is a perfect square, we can express $pf(C)$ as a polynomial in the variables y and $t_{i,l}$, $\{i, l\} \in E$:

$$pf(C) = \sum_{j=0}^{\overline{W}-1} q_j(\mathbf{t})y^j,$$

where $\overline{W}$ is a strict upper bound for the weight of any perfect matching of G, $q_j(\mathbf{t})$ is a polynomial in $\mathbf{t}$ and $\mathbf{t}$ is the vector that collects the $t_{i,l}$ for all $\{i, l\} \in E$.

Lemma 1. *There exists an exact perfect matching of weight j in graph G if and only if $q_j(\mathbf{t})$ is not identically zero in $pf(C)$.*

Proof. It has been proved in [22] and [21] that $q_j(\mathbf{t})$ is the sum of all monomials corresponding to exact perfect matchings of weight j. ∎

The nice property of $pf(C)$ described in Lemma 1 cannot directly be exploited because it would be a task of exponential complexity to obtain explicitly the monomials of $pf(C)$, because, in general, the number of perfect matchings in a graph is exponentially large. However, for fixed $\mathbf{t} = \bar{\mathbf{t}}$, $pf(C(\bar{\mathbf{t}}))$ can be evaluated in polynomial time applying a modified version of the Edmond's algorithm for computing the determinant of $C(\bar{\mathbf{t}})$ where $C(\bar{\mathbf{t}})$ is seen as a matrix with elements in $\mathbb{Z}[y]$, the integrality domain of polynomials in variable y (see [12, 16]).

For planar graphs there exists a special orientation of the edges (called pfaffian orientation) from which we can deduce the values of the elements of $\bar{\mathbf{t}}$ so that the pfaffian counts exactly the number of perfect matchings. This is also known as the Kasteleyn theorem (see Theorem 8.3.4. of [23]). Using Lemma 1 and the Kasteleyn theorem, the existence of a perfect matching of a given weight can be verified through either the algorithm cited in the proof of Proposition 2 or the one cited in the proof of Proposition 3.

Now we show how the techniques adopted to test if a given planar graph contains a k cardinality cut can be used to build a procedure for testing if the same graph contains a k cardinality cut of a given weight W. Such a procedure can be readily used to find a k cardinality cut of minimum weight once its existence has been proved. Indeed, starting from a lower bound $\underline{W}$ on the value of an optimal solution (e.g., the maximum between the value of min cut and the sum of the k minimum edge-weights) one can call the procedure with input k and increasing values $\underline{W}$, $\underline{W} + 1, \ldots$ until the procedure gives a "yes" answer. We notice that in this case a binary search between a lower bound $\underline{W}$ and an upper bound $\overline{W}$ (no matter how obtained) would not improve over the above technique because in the case of a no-answer we cannot halves the current range of search.

Proposition 4. *Given an edge-weighted planar graph $G = (V, E)$ with $|V| = n$, $|E| = m$, every k-cardinality cut of weight W of G is equivalent to an exact perfect matching of weight k, with respect to an edge-weight function w_1, and of weight W, with respect to an edge-weight function w_2, in a suitable planar graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ with $|\widetilde{V}| = 12(n-2)$, $|\widetilde{E}| = 15(n-2)$.*

Proof. The graph $\widetilde{G}$ is derived like in the proof of Proposition 1, the edge-weight function w_1 is defined like $\tilde{w}$, whereas the edge-weight function w_2 is defined setting the weight of the central edge in each chain of $\widetilde{G}$ equal to the weight of the corresponding edge in G and the weights of all other edges equal to 0. With the same arguments used in proof of Proposition 1, one can derive that to a perfect matching of weight W with respect to the edge-weight function w_2 in $\widetilde{G}$, corresponds a cut of weight W in G and vice versa. ∎

Given a planar graph $G = (V, E)$ with vertex set $V = \{1, \ldots, 2n\}$, edge set E, $|E| = m$ and two edge-weight functions w_1: $E \to \mathbb{N}$, w_2: $E \to \mathbb{N}$ let us consider the $2n \times 2n$ skew-symmetric matrix C

$$C_{i,l} = \begin{cases} t_{i,l} y^{w_1(e)} z^{w_2(e)} & \text{if } e = \{i, l\} \in E, i < l, \\ -t_{i,l} y^{w_1(e)} z^{w_2(e)} & \text{if } e = \{i, l\} \in E, i > l, \\ 0 & \text{otherwise} \end{cases}$$

where y and z are variables, whereas the vector $\mathbf{t}$ is derived from the Kasteleyn theorem. Because the determinant of a skew-symmetric matrix of even order is a perfect square we can express $pf(C)$ as a polynomial in the two variables y and z:

$$pf(C(\mathbf{t}, x, y)) = \sum_{i=0}^{\overline{W}_1 - 1} \sum_{j=0}^{\overline{W}_2 - 1} q_{ij}(t) y^i z^j,$$

where $\overline{W}_1$ and $\overline{W}_2$ are strict upper bounds for the weights of any perfect matching of G with respect to the edge-weight function w_1 or w_2, respectively, and $q_{ij}(\mathbf{t})$ is a polynomial in $\mathbf{t}$.

Lemma 2. *Graph G contains an exact perfect matching of weight i with respect to the edge-weight function w_1 and of weight j with respect to w_2 if and only if $q_{ij}(\mathbf{t})$ is not identically zero in $pf(C(\mathbf{t}, x, y))$.*

Proof. Straightforward from the proof of Lemma 1. ∎

By extending the algorithm of [3] (see proof of Proposition 2) one could derive a pseudopolynomial algorithm for testing if the monomial $q_{ij}(\mathbf{t})$ is not identically zero in $pf(C(\mathbf{t}, x, y))$. A lower complexity can be obtained by extending the work of [17] to derive a *random* pseudopolynomial algorithm. To this aim it is first necessary to define two finite fields Z_{q_1} and Z_{q_2} in which to compute the coefficient $q_{ij}(\mathbf{t})$ of the term $y^i z^j$ in $pf(C(\mathbf{t}, x, y))$ by applying the direct and inverse Discrete Fourier Transform (DFT) for the evaluation and interpolation of the polynomial. Working in Z_{q_1} and Z_{q_2} we can compute a unique coefficient of the inverse DFT, that is, we can compute the coefficient $q_{ij}(\mathbf{t})$ by means of the well-known formula

$$q_{ij} = \overline{W}_1^{-1} \overline{W}_2^{-1} \sum_{r=0}^{\overline{W}_1 - 1} \sum_{s=0}^{\overline{W}_2 - 1} pf(C(\mathbf{t}, \omega_1^r, \omega_2^s)) \omega_1^{-ir} \omega_2^{-js}$$

where ω_1 and ω_2 are suitable chosen roots of unities in Z_{q_1} and Z_{q_2}, respectively.

Proposition 5. *When $G = (V, E)$ is a planar graph to test if a k cardinality cut of weight W exists can be done with a random pseudopolynomial algorithm with time complexity $O(n^4 U log(U) + n^5 U)$.*

Proof. By extending Algorithm 3 of [17] for computing the value of q_{kW} and adapting their Theorem 8, we obtain an algorithm with computational complexity $O(\overline{W}_1 \overline{W}_2 n^2 log(\overline{W}_1 + \overline{W}_2) + \overline{W}_1 \overline{W}_2 n^3)$. Because in $\widetilde{G}$, $\overline{W}_1 = n$ and $\overline{W}_2 = nU$, where $U = \max_{e \in E}\{w(e)\}$, the above complexity reduces to $O(n^4 U log(U) + n^5 U)$. ∎

We observe that the random pseudopolynomial algorithm used to solve the problem in Proposition 5 can be converted into an approximated random polynomial algorithm using rounding techniques (see, e.g., section 8.2 of [29]).

Finally, we notice that the same problem is also faced in [24] with interpolation techniques based on the use of the determinant of a special Tutte matrix and on the hypothesis of a specific distribution of the edge weights. However, as indicated in [8], the details of the procedure given there are incorrect, because taking square roots of the determinants obliterates the sign of the pfaffians and therefore yields a wrong interpolating polynomial. This inaccuracy can be fixed, for instance, by first interpolating the values of the determinant and then extracting the square root of the polynomial so obtained, using, for instance, the probabilistic method described in [30]. Therefore, the technique presented in [24] readily becomes impractical because it requires to manipulate really huge numbers. For this reason we selected to work in finite field with a double interpolation technique.

5. A DUAL GREEDY PROCEDURE

The dual greedy procedure receives in input the graph $\widetilde{G}$ introduced in the proof of Proposition 1, an ordered list T of edges of $\widetilde{G}$, the edge-weight function $\tilde{w}$ of $\widetilde{G}$, the value k, the skew symmetric-matrix C associated to graph $\widetilde{G}$, and an initially empty set PM. After termination the procedure returns the set PM, which contains a perfect matching in $\widetilde{G}$ of value k with respect to the edge-weight function $\tilde{w}$.

The list T contains the m edges of $\widetilde{G}$, which have weight equal to 1. By construction of $\widetilde{G}$ those edges are in one to one correspondence with the edges in G and we sort them in nonincreasing order of weight with respect to the edge-weight function w of G. The procedure then examines each edge in T in the given order and determine either that the considered edge is necessary to the existence of the solution under construction or that it can be removed without compromising the ongoing computation. The procedure is listed below.

Algorithm 1. ***DualGreedy**($\widetilde{G}, T, \tilde{w}, k, C, PM$)*

1. **If** $|PM| = 6(n-2)$ **then return** PM;
2. *Remove from $\widetilde{G}$ the first edge $\bar{e} = (j_1, j_2)$ of T in the given order;*
3. **If** *$\widetilde{G}$ does not contain a k cardinality cut* **then**
4. $\quad PM := PM \cup \{\bar{e}\}$;
5. $\quad$ *remove the two edges e' and e'' connected to the edge $\bar{e}$ from $\widetilde{G}$;*
6. $\quad$ *erase the rows and columns j_1 and j_2 from matrix C;*
7. $\quad k := k - 1$;
8. **Else**
9. $\quad PM := PM \cup \{e', e''\}$;
   $\quad$ *{where $e' = (j_1, j_3)$ and $e'' = (j_2, j_4)$ are the two edges connected to $\bar{e}$}*
10. $\quad$ *remove from $\widetilde{G}$ e', e'' and each edge e connected with them;*
11. $\quad$ *erase the rows and columns j_1, j_2, j_3 and j_4 from matrix C;*
12. **DualGreedy**($\widetilde{G}, T, \tilde{w}, k, C, PM$);

Proposition 6. *The dual greedy procedure builds a feasible solution with time complexity $O(nT_A)$, where T_A is the complexity of the algorithm used to test the existence of a k cardinality cut.*

Proof. The time complexity of the dual greedy algorithm is $O(mT_A)$ and $m = O(n)$. To see this we observe that the heaviest part of Algorithm 1 is the call to the existence procedure in step 3, and this call is done once for each removed element in T. To see that m calls are enough to identify a perfect matching of $2m$ elements, we observe which are the consequences of steps from 4 to 11 on the original graph G. Every time steps from 4 to 7 are executed the edge $\bar{e}$, which is added to the perfect matching in $\widetilde{G}$, sets the corresponding edge in G as part of the k-cut. Otherwise, the execution of steps from 9 to 11 which adds two 0 valued edges to PM, consists in the removal from G of the edge corresponding to $\bar{e}$ and in the consequent updating of the exploded graph $\widetilde{G}$. Hence, at each iteration an edge is either identified as part of the k-cut in G or it is removed from G.

Finally, the dual greedy algorithm builds a feasible solution *deterministically*. To see this it is sufficient to adopt in step 3 the same procedure used, before the call of the dual greedy algorithm, to prove that a feasible solution exists. In this case, as reported in [17], the random procedure in step 3 always answers correctly when it is iteratively called in the constructive procedure based on the removals of elements. ∎

Proposition 7. *The dual greedy procedure returns a solution whose value is at most k times the optimal one and the bound is tight.*

Proof. Let M denote the value of the heaviest edge, say e, in an optimal solution of the min k-cardinality cut problem, which is discarded by the dual greedy solution. Hence, the value of an optimal solution is at least M. On the other hand,

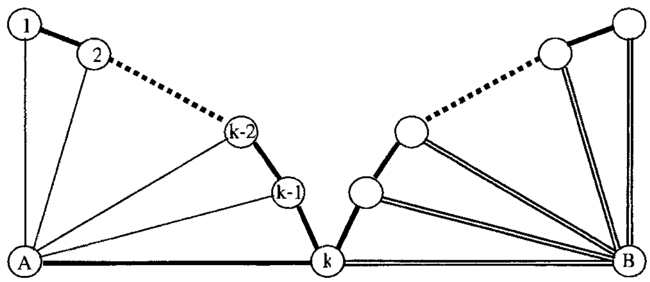

FIG. 6. The "tight" instance.

the dual greedy procedure can output a solution of value at most $k(M - \varepsilon)$ after discarding the edge e. Figure 6 shows an instance in which this situation is verified. The edges in bold, thin, and double line have weight M, 0 and $M - \varepsilon$, respectively. The optimal solution of the min k-cardinality cut problem on this instance is given by the edges incident to vertex A. On the other hand, on this instance the dual greedy procedure returns the solution composed by all the edges incident to vertex B. ∎

Remark 1. We notice that the DualGreedy procedure always finds an optimal solution for the min-max k-cardinality cut problem, that is, the problem of finding a k-cardinality cut such that is minimal the weight of the heaviest edge in the cut.

6. LAGRANGIAN RELAXATION

Given an undirected graph $G = (V, E)$ and an edge-weight function w: $E \to \mathbb{N}$, the min k-cardinality cut problem can be formulated as

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} w(e) x_e \\ \text{subject to :} \quad & x \text{ be the incidence vector of a cut} \\ & \sum_{e \in E} x_e = k \end{aligned} \tag{3}$$

The Lagrangian relaxation of cardinality constraints yields

$$\theta(\lambda) = \lambda k + mincut(G, w - \lambda) \tag{4}$$

where $mincut(G, w - \lambda)$ indicates the optimal value of min cut problem on the graph G with weight function $w - \lambda$. Because in literature the min cut problem is only defined for nonnegative edge-weights, whereas the max cut problem is also defined for negative edge-weights it is more correct to rewrite

$$\theta(\lambda) = \lambda k - maxcut(G, \lambda - w) \tag{5}$$

We notice that the *min cut* problem requires to find a nonempty cut because otherwise we always obtain the empty cut like trivial solution of this problem. Whereas the classical *max cut* problem also allows an empty cut like solution. Therefore, in order for (5) to be equivalent to (4), in (5) we indicate by $maxcut(G, \lambda - w)$ the optimal value of max cut problem with the additional constraint that the cut found is not empty.

Exploiting the fact that in planar graphs the classical *max cut* problem can be solved in polynomial time through the LP formulation (1), we can also polynomially solve $maxcut(G, w)$ using the following procedure:

Algorithm 2. ***Maxcut***(G, w)

1. **If** $w(e) \leq 0\ \forall e \in E$, *then set* $z^* := -mincut(G, -w)$;
2. **Else if** $w(e) \geq 0\ \forall e \in E$, **then** *let* z^* *the optimal value of LP (1) applied to graph* (G, w);
3. **Else**
4. *Set* $z^* := -\infty$, $M := w(E)$;
5. **While** $|V| > 1$
6. *Choose an arbitrary edge* $\{s, t\} \in E$ *and set* $w(\{s, t\}) := w(\{s, t\}) + M$;
7. *Let* $\bar{z}$ *be the optimal value and let* $\overline{C}$ *be the optimal solution of LP (1)* *for the current graph (G,w);*
8. **If** $\bar{z} - M > z^*$ **then** $z^* := \bar{z} - M$ *and* $C^* := \overline{C}$;
9. *Shrink G by merging vertices s and t and update the edge weights accordingly;*
11. **Return** z^*

This procedure is a self-reduction that follows the algorithm proposed in [28]. We notice that through the merging operation of step 9 this algorithm requires to solve just $n - 1$ LP problems like (1) with decreasing size instead of $o(n^2)$ problems. The min cut problem required by step 1 has been efficiently solved using the algorithm of [28]. From (5) we can rewrite

$$\begin{aligned} \theta(\lambda) &= \lambda k - maxcut(G, \lambda - w) \\ &= (k - |E_\lambda|)\lambda + w(E_\lambda) \end{aligned} \tag{6}$$

where E_λ is the edge set of a solution of *maxcut*$(G, \lambda - w)$ and $w(E_\lambda) := \sum_{e \in E_\lambda} w(e)$. Relation (6) emphasizes that $\theta(\lambda)$ is a piecewise linear function.

Now we state some properties of $\theta(\lambda)$ that we will use in the sequel and that can be easily derived from the fact that $\theta(\lambda)$ is a concave and piecewise linear (see, i.e., [25]).

Proposition 8. *For any real λ, if E_λ is the edge set of a solution of maxcut$(G, \lambda - w)$ and $|E_\lambda| = k'$ then E_λ is a min k'-cardinality cut of (G, w).*

Proposition 9. *Given $\lambda_1, \lambda_2 \in \mathbb{R}$ with $\lambda_1 \leq \lambda_2$, let E_{λ_1} a solution of maxcut$(G, \lambda_1 - w)$ with $|E_{\lambda_1}| = k_1$ and let E_{λ_2} a solution of maxcut$(G, \lambda_2 - w)$ with $|E_{\lambda_2}| = k_2$. Then $k_1 \leq k_2$.*

In particular, from Proposition 8 we have that when E_λ is the edge set of a solution of *maxcut*$(G, \lambda - w)$ such that $|E_\lambda| = k$ then E_λ is an optimal solution of k-cardinality cut. In fact, any optimal solution of the Lagrangian relaxation of k-cardinality cut that satisfies the cardinality constraint must to be optimal for the original problem because the relaxed constraint is an equality constraint. The same fact can be deduced by the following argument. If we subtract λ from every edge, all k-cardinality cuts lose exactly a $k\lambda$ value. Thus, whichever cut is the minimum k-cardinality cut, it stays minimum. So, if the minimum cut has cardinality k, it is the minimum k-cardinality cut.

From relation (6) and Proposition 9 we have that for a small enough λ, function $\theta(\lambda)$ is a straight line having slope $k -$ *min cardinality cut*, whereas for a big enough λ, $\theta(\lambda)$ is a straight line having slope $k -$ *max cardinality cut*. The corner points of function $\theta(\lambda)$ are due to the fact that in correspondence of the same $\bar{\lambda}$ we can have two solutions $E'_{\bar{\lambda}}$ and $E''_{\bar{\lambda}}$ of *maxcut*$(G, \bar{\lambda} - w)$ with different cardinality. The behavior of function $\theta(\lambda)$ is represented in Figure 7.

A possible choice of the values λ_1 and λ_2 such that for $\lambda \leq \lambda_1$ and for $\lambda \geq \lambda_2$ the slope of $\theta(\lambda)$ does not change is established by the following proposition:

Proposition 10.
Let

$$\lambda_1 := \max \left\{ \frac{w(\widehat{E})}{|\widehat{E}| - k}, w(E^*) - w(\widehat{E}) \right\}$$

and let

$$\lambda_2 := \min \left\{ \frac{w(\widetilde{E})}{|\widetilde{E}| - k}, w(\widetilde{E}) - w(E^*) \right\}$$

where $\widehat{E}$ is the edge set of a min cardinality cut, E^* is the edge set of a min cut and $\widetilde{E}$ is the edge set of a max cardinality cut. For any $\lambda \leq \lambda_1$ the solution of maxcut$(G, \lambda - w)$ does not change from the min $|\widehat{E}|-$cardinality cut and for any $\lambda \geq \lambda_2$ the solution of maxcut$(G, \lambda - w)$ does not change from the min $|\widetilde{E}|-$cardinality cut.

Proof. We prove the proposition only for λ_1, because for λ_2 the proof is similar. We show that when λ_1 is equal to $\frac{w(\widehat{E})}{|\widehat{E}|-k}$ then λ_1 is on the left of the intersection, $\bar{\lambda}$, of the most left straight line in $\theta(\lambda)$ with the λ axis. From (6) we have

$$\bar{\lambda} = \frac{w(\overline{E})}{|\overline{E}| - k}$$

where $\overline{E}$ is the minimal cardinality cut with the minimal weight. Because $|\overline{E}| - k < 0$ and $|\overline{E}| = |\widehat{E}|$ it results

$$\frac{w(\widehat{E})}{|\widehat{E}| - k} \leq \frac{w(\overline{E})}{|\overline{E}| - k} = \bar{\lambda}$$

Now we show that when $\lambda_1 = w(E^*) - w(\widehat{E})$ then λ_1 is on the left of the first breakpoint, $\tilde{\lambda}$, of $\theta(\lambda)$, (i.e. the first point where the slope of $\theta(\lambda)$ changes). From (6) $\tilde{\lambda}$ must to be such that

$$\tilde{\lambda}(k - |\overline{E}|) + w(\overline{E}) \leq \tilde{\lambda}(k - |E'|) + w(E')$$

for all cut E'.

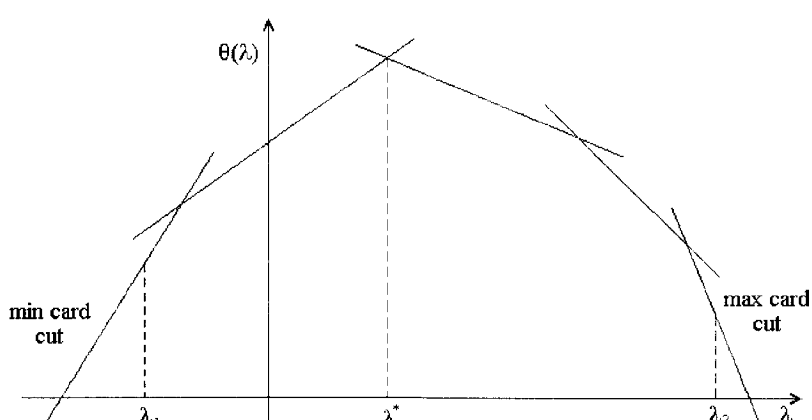

FIG. 7. Behavior of function $\theta(\lambda)$.

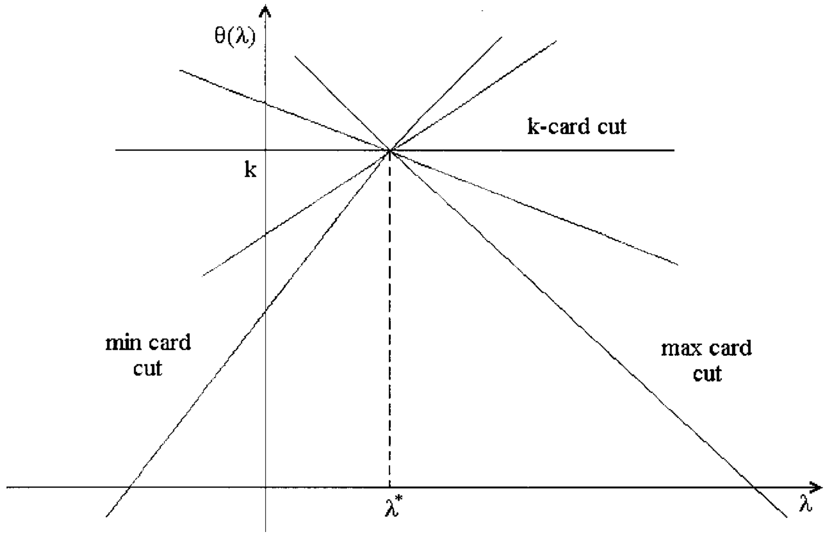

FIG. 8. Behavior of function $\theta(\lambda)$ for the uniform edge-weight case.

Therefore $\tilde{\lambda}$ is given by

$$\tilde{\lambda} = \min\left\{ \frac{w(E') - w(\overline{E})}{|E'| - |\overline{E}|} : E' \text{ is a cut of } G \right\}$$

Hence, it results

$$\lambda_1 = w(E^*) - w(\widehat{E}) \le \tilde{\lambda}$$

■

6.1. Solving the Lagrangian Dual

The best Lagrangian lower bound of k-cardinality cut is given by

$$\theta(\lambda^*) = \max_{\lambda \text{ free}} \theta(\lambda) \tag{7}$$

The optimization problem (7) is known as the *Lagrangian dual* problem. From Proposition 10 we have that (7) is equivalent to

$$\theta(\lambda^*) = \max_{\lambda \in [\lambda_1, \lambda_2]} \theta(\lambda) \tag{8}$$

From the properties of $\theta(\lambda)$ established before it is easy to see that $\theta(\lambda)$ is differentiable almost everywhere. When all the solutions of $maxcut(G, \lambda - w)$ have the same cardinality then λ is not a corner point and the derivative of $\theta(\lambda)$ is given by

$$\theta'(\lambda) = k - |E_\lambda|$$

where E_λ is a solution of $maxcut(G, \lambda - w)$. Whereas when for arbitrarily small $\varepsilon \ge 0$, $maxcut(G, \lambda - \varepsilon - w)$ and $maxcut(G, \lambda + \varepsilon - w)$ have two solutions $E_\lambda{}'$ and $E_\lambda{}''$, respectively, with $|E_\lambda{}'| \ne |E_\lambda{}''|$, then λ is a corner point of $\theta(\lambda)$ and we can define the left derivative $\theta'_-(\lambda)$ and the right derivative $\theta'_+(\lambda)$ in the following way:

$$\theta'_-(\lambda) = k - |E_\lambda{}'| \tag{9}$$

$$\theta'_+(\lambda) = k - |E_\lambda{}''| \tag{10}$$

The solution λ^* of (8) is the only corner point of $\theta(\lambda)$ given by the intersection of a straight line having non negative slope with a straight line having non positive slope. So λ^* is the only corner point of $\theta(\lambda)$ such that $\theta'_-(\lambda^*) \ge 0$ and $\theta'_+(\lambda^*) \le 0$. Therefore, λ^* can be determined within an uncertainty interval of width $\varepsilon > 0$ through a dichotomic search algorithm on λ in the interval $[\lambda_1, \lambda_2]$ where λ_1 and λ_2 are computed as described in Proposition 10.

An additional stopping criterium of the dichotomic search algorithm is the condition $|E_\lambda| = k$, where E_λ is a solution of $maxcut(G, \lambda - w)$, because in this case, according to Proposition 8, E_λ is an optimal min k-cardinality cut.

When a cut with cardinality k is not found the algorithm ends returning a final uncertainty interval $[\lambda_1^{end}, \lambda_2^{end}]$. In this case, the algorithm yields two cuts $E_{\lambda_1^{end}}, E_{\lambda_2^{end}}$, which are the solutions of $maxcut(G, \lambda_1^{end} - w)$ and $maxcut(G, \lambda_2^{end} - w)$,

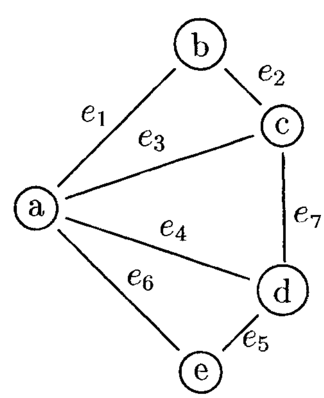

FIG. 9. Planar graph G having bad Lagrangian lower bound.

respectively, such that according to Proposition 9

$$k_1 := |E_{\lambda_1^{end}}| < k < k_2 := |E_{\lambda_2^{end}}|$$

and according to Proposition 8 $E_{\lambda_1^{end}}$ is a min k_1-cardinality cut and $E_{\lambda_2^{end}}$ is a min k_2-cardinality cut.

We observe that unfortunately there is no approximation guarantee that the cardinalities of $E_{\lambda_1^{end}}$ and $E_{\lambda_2^{end}}$ are close to k. Indeed, if we consider the uniform edge-weight case we obtain the worst approximation result because $E_{\lambda_1^{end}}$ is reduced to a min cardinality cut, whereas $E_{\lambda_2^{end}}$ is reduced to a max cardinality cut, for any value of k. In fact, when $w(e) = 1 \forall e \in E$ it results

$$\begin{aligned} \theta(\lambda) &= \lambda k - maxcut(G, \lambda - w) \\ &= \lambda k - maxcut(G, \lambda - 1) \\ &= (k - |E_\lambda|)\lambda + |E_\lambda| \end{aligned} \tag{11}$$

where E_λ is a solution of $maxcut(G, \lambda - 1)$. So E_λ is a min cardinality cut for $\lambda < 1$, E_λ can be any cut for $\lambda = 1$, and E_λ is a max cardinality cut for $\lambda > 1$, and therefore, the behavior of $\theta(\lambda)$ is that one represented in Figure 8. Hence, for any $\varepsilon > 0$, although $\lambda_2^{end} - \lambda_1^{end} \leq \varepsilon$, if $\lambda_1^{end} < \lambda^* < \lambda_2^{end}$, $E_{\lambda_1^{end}}$ and $E_{\lambda_2^{end}}$ do not change from the min cardinality cut and the max cardinality cut, respectively, and so they result to be the two cuts with the farthest cardinality.

We observe that in the uniform edge-weight case the optimal Lagrangian lower bound is $\theta(\lambda^*) = k$ and so it coincides with the optimal min k-cardinality cut. An example where the optimal Lagrangian lower bound is arbitrarily far from the optimal min k-cardinality cut is the following one.

Consider the graph $G = (V, E)$ drawn in Figure 9 with the edge-function w defined below

$$w(e) = \begin{cases} 1 & \text{for all } e \in E \backslash \{e_7\} \\ 1 + M & \text{for } e = e_7 \end{cases}$$

where M represents an arbitrarily large positive constant.

When k is equal to 3, the behavior of $\theta(\lambda)$ is that represented in Figure 10. Therefore, the optimal Lagrangian lower bound is $\theta(\lambda^*) = 3$, obtained for $\lambda^* = 1$, whereas the optimal min 3-cardinality cut value is $3 + M$.

Another efficient method for solving the Lagrangian dual problem (7) is by way of parametric linear program (see [4]). In fact, the total number of breakpoints of function $\theta(\lambda)$ is upper bounded by $m = |E|$ because each breakpoint corresponds to a change of cut cardinality: therefore, the parametric linear program (7) can be solved in polynomial time by the simplex algorithm even if in the worst case we visit all the breakpoints.

7. NUMERICAL RESULTS

Computational experiments have been carried on only for testing the Lagrangian heuristic because the computational complexity of the dual greedy procedure and of the random pseudopolynomial algorithm makes them useful only for very small instances.

7.1. Description of the Instances for k-Cardinality Cut

Because no validation instances are publicly available for the min k-cardinality cut problem we have generated instances ad hoc for this problem. We have generated connected planar graphs considering as vertices uniformly distributed random points in a square and linking pairs of them by an edge only if the edge does not cross any edges previously generated and until an edge density non less than 50% is achieved (with respect to the maximum number of edges in a planar graph). All graphs have random integer weights on the edges uniformly distributed between 1 and 100. We have built 10 different planar graphs having 30 vertices and a number of edge between 60 and 70. Finally, we have created 300 instances of min k-cardinality cut considering for each

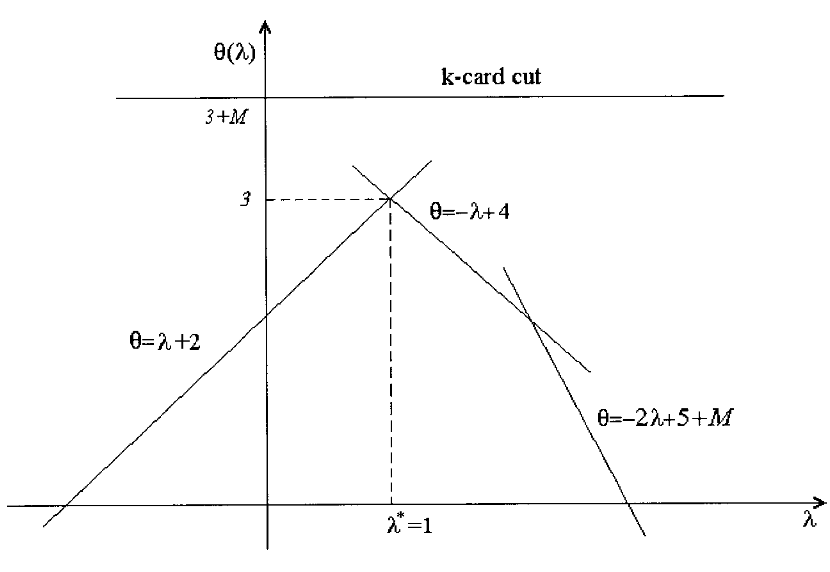

FIG. 10. Behavior of function $\theta(\lambda)$ for the graph drawn in Figure 9.

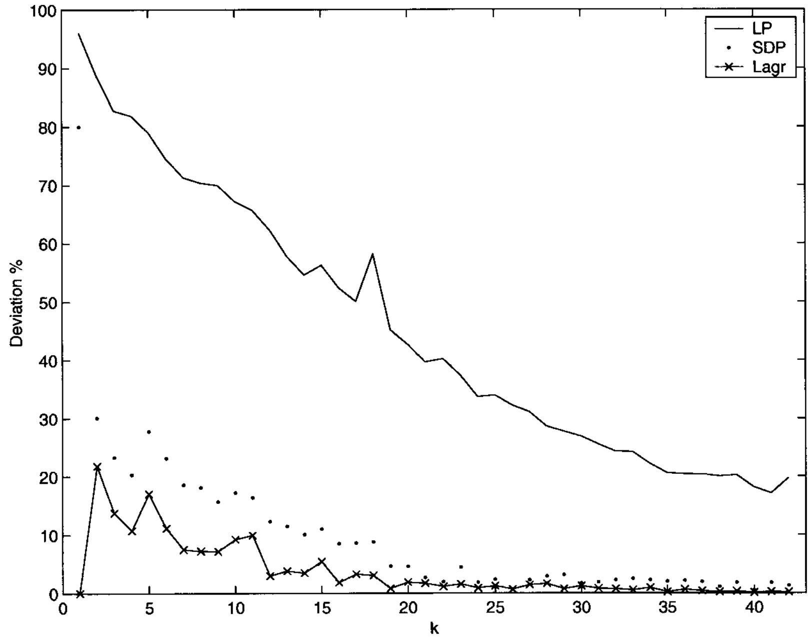

FIG. 11. Deviations of lower bounds from the optimal values.

planar graph 30 different values of k obtained computing the cardinalities of the cuts generated by random partitions of the vertex set.

7.2. *Solution of the Instances and Final Remarks*

All the algorithms presented were implemented in *C* and run on a AMD K7 1 GHz computer with 1.2 GB RAM under the Linux 2.2.14 operating system.

For the solution of the LP problems (1) in the Algorithm 2 we used the LP solver `Cplex 7.0` where for efficiency reasons we do not add all triangle inequalities at once, but we include them successively according to the amount of violation.

For all the instances we have found the optimal solution solving a mixed integer linear formulation for min k-cardinality cut presented in [6] by the MIP solver `Cplex 7.0`. Therefore, on this set of instances we calculated the average

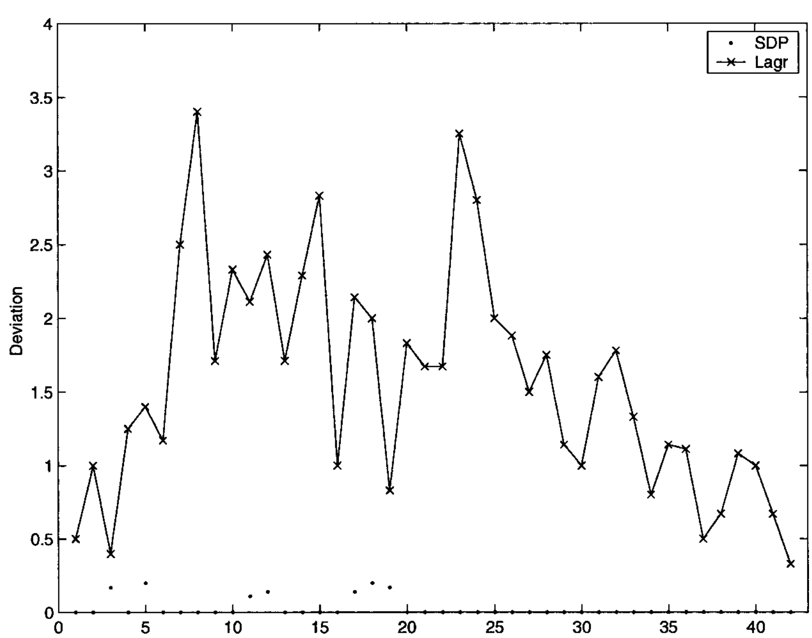

FIG. 12. Deviations of cardinalities of heuristic solutions from the required value of k.

relative deviation of the Lagrangian lower bounds from the optimal values for each value of k. In Figure 11 we compare the behavior of the deviation of the Lagrangian lower bounds with that one of Linear Programming relaxation (LP) and of Semidefinite Programming relaxation (SDP) presented in [6].

From this figure we can note that on our set of instances both SDP relaxation and the Lagrangian relaxation yield a lower bound by far better than the LP relaxation and the first one is ever dominated by the second one (or at the maximum equal). Both SDP relaxation and Lagrangian relaxation yield the tightest bounds for big values of k. Although in Subsection 6.1 we have detected an example where the Lagrangian bound can be arbitrarily far from the optimal value of min k-cardinality cut the experiments seem to reveal that in the mean case the Lagrangian bound is very tight. A reason why the Lagrangian relaxation works well is suggested by the article of [20], who shows that when random weights are assigned to a graph, every cut gets value close its expectation with high probability. Therefore, every cut of cardinality significantly larger than k ends up with weight significantly larger than the expected weight of a k-cardinality cut, so that it does not compete with the k-cardinality cuts in the search for a minimum. In particular, given that the expected weight of a k-cardinality cut is $50km$ presumably $\lambda \approx 50$ at the Lagrangian optimum.

Moreover, we have noted that for all instances that are similar to the example presented in Subsection 6.1, the SDP relaxation yields exactly the same bound of the Lagrangian relaxation. Finally, about the CPU time we notice that although for the set of instances considered the Lagrangian relaxation and the SDP relaxation require roughly the same time, less than 1.5 minutes, the increase in CPU time is much less for the Lagrangian relaxation than the SDP relaxation when the size of the instances increases.

Concerning to the heuristic solutions we report in Figure 12, the graphics of the average absolute deviation of the cardinalities of Lagrangian heuristic solutions and of SDP heuristic solutions from the required value of k. From this figure we can deduce that the SDP heuristic approximates the wanted cardinality by far better than the Lagrangian heuristic. However, we remember that each time the Lagrangian heuristic returns a k-cardinality cut this is certainly an optimal min k-cardinality cut, whereas this does not happen for the SDP heuristic.

REFERENCES

[1] F. Barahona, On cuts and matchings in planar graphs, Mathematical Programming 60 (1993), 53–68. Springer, Berlin, 1990.

[2] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt, An application of combinatorial optimization to statistical physics and circuit layout design, Oper Res 36(3) (1988), 493–513.

[3] F. Barahona and W.R. Pulleyblank, Exact arborescences, matchings and cycles, Discr Appl Math 16 (1987), 91–99.

[4] D. Bertsimas and J.N. Tsitisklis, Introduction to Linear Optimization, 217:221, Athena Scientific, Belmont, Massachusetts, 1997.

[5] B. Borchers, CSDP, a C library for semidefinite programming, Optimization Methods and Software 11 (1999), 613–623. http://www.nmt.edu/~borchers/csdp.html.

[6] M. Bruglieri, M Ehrgott, and F. Maffioli, Cardinality constrained minimum cut problems: Complexity and algorithms, Discr Appl Math 137 (2004), 311–341.

[7] R.G. Busacker and T.L. Saaty, Finite Graphs and Networks: An Introduction with Applications, McGraw-Hill, New York, 1965.

[8] P.M. Camerini, G. Galbiati, and F. Maffioli, Random pseudo-polynomial algorithms for exact matroid problems, J Algorithms 13 (1992), 258–273.

[9] T.J. Chang, N. Meade, J.E. Beasley, and Y.M. Sharaiha, Heuristics for cardinality constrained portfolio optimisation, Comp and Oper Res 27 (2000), 1271–1302.

[10] J. Cong and Y. Ding, FlowMap: An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs, IEEE Trans on Computer-Aided Design (1994), 1–12.

[11] M. Dell'Amico and S. Martello, The k-cardinality assignment problem, Discr Appl Math 76 (1997), 103–121.

[12] J. Edmonds, System of distinct representatives and linear algebra, J Res National Bureau of Standards (1967), B:241–245.

[13] M. Ehrgott, J. Freitag, H.W. Hamacher, and F. Maffioli, Heuristics for the k-cardinality tree and subgraph problem, Asia Pacific J Oper Res 14(1) (1997), 87–114.

[14] M. Ehrgott, H.W. Hamacher, and F. Maffioli, Fixed cardinality combinatorial optimization problems—A survey, Report in Wirtschaftsmathematik, 56, Fachbereich Mathematik, Universität Kaiserslautern, 1999.

[15] L.R. Foulds, H.W. Hamacher, and J.M. Wilson, Integer programming approaches to facilities layout models with forbidden areas, Annals of Oper Res 81 (1998), 405–417.

[16] G. Galbiati and F. Maffioli, On the computation of Pfaffians, Discr Appl Math 51 (1994), 269–275.

[17] G. Galbiati, F. Maffioli, and C. Viola, Randomized Algorithms over Finite Fields for the Exact Parity Base Problem, J Complexity 15 (1999), 537–556.

[18] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman & Co., San Francisco, 1979.

[19] R.E. Gomory and T.C. Hu, Multi-terminal network flows, SIAM J Appl Math 9 (1961), 551–570.

[20] D.R. Karger, Random sampling in cut, flow, and network design problems, Math Oper Res 24(2) (1999), 383–413.

[21] R.M. Karp, E. Upfal, and A. Widgerson, Finding a maximum matching is in RNC, Proceedings of the 17th ACM Symposium on the Theory of Computing, pp. 22–32. Association for Computing Machinery, New York, 1985.

[22] L. Lovasz, On determinants, matchings and random algorithms, Fund Comput Theo 79 (1979), 565–574.

[23] L. Lovasz and M.D. Plummer, Annals of Discrete Mathematics (29), Matching theory, 322–333. North-Holland, 1986.

[24] K. Mulmuley, U.V. Vazirani, and V.V. Vazirani, Matching is as easy as matrix inversion, Combinatorica 7(1) (1987), 105–113.

[25] G.L. Nemhauser and L.A. Wolsey, Integer and Combinatorial Optimization, 323, 330, John Wiley & Sons, New York, 1988.

[26] G.I. Orlova and Y.G. Dorfman, Finding the maximum cut in a graph, Engineering Cybernetics 10 (1972), 502–506.

[27] M. Padberg and G. Rinaldi, An efficient algorithm for the minimum capacity cut problem, Math Program 47 (1990), 19–39.

[28] M. Stoer and F. Wagner, A simple min-cut algorithm, J ACM 44 (1997), 585–591.

[29] V.V. Vazirani, Approximation Algorithms, Springer, Belin, Second Printing, 2003.

[30] J. Von Zur Gathen, Parallel algorithms for algebraic problems, Proceedings, 15th ACM, STOC, 1983, pp. 17–23.