WEAKLY BIPARTITE GRAPHS AND THE MAX-CUT PROBLEM

M. GRÖTSCHEL *,**

Institut für Operations Research, Universität Bonn, Nussestr. 2, D-5300 Bonn 1, Fed. Rep. Germany

W.R. PULLEYBLANK *

Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ontario, Canada

Received July 1981
Revised August 1981

A new class of graphs, called weakly bipartite graphs, is introduced. A graph is called weakly bipartite if its bipartite subgraph polytope coincides with a certain polyhedron related to odd cycle constraints. The class of weakly bipartite graphs contains for instance the class of bipartite graphs and the class of planar graphs. It is shown that the max-cut problem can be solved in polynomial time for weakly bipartite graphs. The polynomial algorithm presented is based on the ellipsoid method and an algorithm that computes a shortest path of even length.

Max-cut problem, bipartite graphs, planar graphs, ellipsoid method, shortest paths

1. Introduction and notation

In this paper we introduce a new class of graphs, called weakly bipartite graphs, which for instance contains the class of bipartite graphs and the class of planar graphs. We show that the max-cut problem is solvable in polynomial time for weakly bipartite graphs. The polynomial algorithm given is based on the ellipsoid method and uses the fact that weakly bipartite graphs have a polyhedral characterization.

The graphs we consider are finite and undirected. They may have multiple edges. Loops do not play a role in what follows, so we assume that our graphs have no loops. We denote a graph by $G=[V,E]$, where V is the node set and E the edge set of G. If $e\in E$ is an edge with endnodes i and j we also write ij to denote the edge e. If $H=[W,F]$ is a graph with $W\subseteq V$ and $F\subseteq E$ then H is called a *subgraph* of G.

If $W\subseteq V$, $\emptyset\neq W\neq V$, then $\delta(W)$ is the set of edges with one endnode in W and the other in $V\setminus W$. The edge set $\delta(W)$ is called a *cut*. We write $\delta(v)$ instead of $\delta(\{v\})$ for $v\in V$. For a node $v\in V$ the cardinality of $\delta(v)$ is called the *degree* of v.

* Sponsored by Sonderforschungsbereich 21 (DFG), Universität Bonn, Fed. Rep. Germany.

** Sponsored by NSF-Grant ECS-8005350 to Cornell University.

A *path* P in $G=[V,E]$ is a sequence of edges $e_1,e_2,\dots,e_k$ such that $e_1=v_0v_1$, $e_2=v_1v_2,\dots,e_k=v_{k-1}v_k$ and such that $v_i\neq v_j$ for $i\neq j$. The nodes v_0 and v_k are the endnodes of P and we say that P links v_0 and v_k or goes from v_0 to v_k. The number k of edges of P is called the *length* of P. If $P=e_1,e_2,\dots,e_k$ is a path linking v_0 and v_k and $e_{k+1}=v_0v_k\in E$ then the sequence $e_1,e_2,\dots,e_k,e_{k+1}$ is called a *cycle* of length $k+1$. A cycle (path) is called *odd* if its length is odd, otherwise it is called even.

Clearly, we can also consider a path as the edge set of a connected subgraph H of G such that exactly two nodes of H have degree one while all other nodes of H have degree two. Similarly, a cycle can be considered as the edge set of a connected subgraph of G in which all nodes have degree two. The edge set of a subgraph $H=[W,F]$ of G in which all nodes have even degree is called a *quasi-cycle*. If $F\neq\emptyset$ then F is obviously the union of cycles any two of which are (edge-) disjoint.

A *matching* M in G is a set of edges such that every node of G is contained in at most one edge of M. A matching M is called *perfect* if every node is contained in an edge of M.

A graph is called *bipartite* if its node set can be partitioned into two nonempty, disjoint sets V_1 and V_2 such that no two nodes in V_1 and no two nodes in V_2 are linked by an edge. If G is bipartite and $|V_1|=n$, $|V_2|=m$ and every node in V_1 is linked to every node in

0167-6377/81/0000-0000/$02.75

V_2 by exactly one edge, then G is denoted by $K_{n,m}$ and called *completely bipartite*. Obviously, if $\delta(W)$ is a cut in a graph G, then $[V,\delta(W)]$ is a bipartite subgraph of G.

2. The max-cut problem

The *max-cut problem* can be stated as follows. Given a graph $G=[V,E]$ with edge weights $c_e>0$ for all $e\in E$, find a cut $\delta(W)$ such that $c(\delta(W)) := \sum_{e\in\delta(W)} c_e$ is as large as possible. Replacing "as large as possible" by "as small as possible", we obtain the *min-cut problem*.

It is well known that for rational weights the min-cut problem can be solved in polynomial time using network flow techniques. The max-cut problem, however, is NP-complete for the class of all graphs, cf. Garey and Johnson [6]. Even various restricted max-cut problems are hard. For instance, if the problem is restricted to the class of graphs with nodes having degree at most three, or to the graphs which have a node v whose removal results in a planar graph in which all nodes have degree at most six, then the max-cut problem for these graphs is still NP-complete, cf. Barahona [1], Yannakakis [10], Barahona [2]. This even holds when all edge weights are assumed to equal one.

On the other hand, the max-cut problem is solvable in polynomial time for planar graphs, cf. Hadlock [8] and Orlova and Dorfman [9]. The algorithm is based on planar duality. Namely, if G is planar, connected, and has no cut-edge (i.e. a cut of cardinality one) then every cut in G corresponds to a unique quasi-cycle in the dual graph G^* with the same weight. A maximum weight quasi-cycle can be computed in polynomial time (in any graph) using Chinese postman techniques, i.e. shortest path and matching algorithms.

Recently Barahona [3] has generalized the cardinality version (i.e. all edge weights are equal to one) of the case above to graphs with fixed genus. A surface of genus p is a surface obtained from a sphere by attaching p 'handles'. If a graph G can be drawn on a surface of genus p such that no two edges intersect and if p is minimum with respect to this property, then G is said to have *genus p*. Clearly, planar graphs have genus zero. The genus of a given graph G can be obtained in polynomial time with the algorithm of Filotti and Miller [5]. Using various transformations and matching techniques it was shown in [3] that for a graph G of genus p a maximum cardinality cut can be found in $O(4^p|V|^5)$ time.

Our approach to the max-cut problem is quite different from the ones described above. We use polyhedral techniques and the ellipsoid method.

If $G=[V,E]$ is a graph and $F\subseteq E$ an edge set then the vector $x^F\in\mathbf{R}^E$ with $x_e^F=1$ if $e\in F$ and $x_e^F=0$ if $e\notin F$ is called the *incidence vector* of F. The polytope

$$P_B(G) := \operatorname{conv}\{x^F\in\mathbf{R} \mid [V,F] \text{ is a bipartite subgraph of } G\} \tag{2.1}$$

is called the *bipartite subgraph polytope* of G. Obviously, every edge set of a bipartite subgraph of G is contained in a cut of G. This implies that for positive edge weights c_e, $e\in E$, every optimum basic solution of the linear program

$$\max c^{\mathrm{T}}x, \quad x\in P_B(G) \tag{2.2}$$

corresponds to a cut. Hence, whenever the LP (2.2) can be solved in polynomial time, the max-cut problem can be solved in polynomial time (and vice versa).

3. Weakly bipartite graphs

Problem (2.2) is theoretically a linear program, but the way $P_B(G)$ is given is not suitable for LP-methods. What we need is a description of $P_B(G)$ in the form of an inequality system $Ax\leqslant b$. Such a system always exists, but it is very unlikely that we can find an explicit system for all graphs G. Sometimes partial inequality systems turn out to be quite helpful for solving special cases.

Clearly, the trivial inequalities

$$0\leqslant x_e\leqslant 1, \quad e\in E \tag{3.1}$$

are valid with respect to $P_B(G)$. Since $P_B(G)$ contains the zero vector and all unit vectors, the trivial inequalities even define facets. It is obvious that the inequalities (3.1) determine $P_B(G)$ completely if and only if G is bipartite.

A well-known theorem states that a graph is bipartite if and only if it contains no odd cycle. This implies that every incidence vector of the edge set of a bipartite subgraph of G satisfies the inequalities

$$x(C) := \sum_{e\in C} x_e \leqslant |C|-1, \quad C \text{ an odd cycle in } G. \tag{3.2}$$

In fact, it is not hard to see that for every graph G every inequality (3.2) defines a facet of $P_B(G)$, cf. Barahona [2]. This class of inequalities is a subclass of the so-called rank inequalities. If $F\subseteq E$, then the *rank* $r(F)$ of F is the maximum cardinality of the edge set of a bipartite graph contained in F. Then for every $F\subseteq E$

$$x(F)\leqslant r(F) \tag{3.3}$$

is clearly valid with respect to $P_B(G)$. For general F, the rank $r(F)$ is hard to compute (in fact, this is a max-cut problem). Thus from a computational point of view, only those rank inequalities are of interest where $r(F)$ can be computed easily, as in the case (3.2). There are further edge sets where the rank can be given explicitly, e.g. the edge sets of complete subgraphs. But none of these (as far as we know at present) has such interesting properties as (3.2). We therefore

3.4. Definition. A graph $G=[V,E]$ which satisfies

$$P_B(G) = \{x \in \mathbf{R}^E \mid x \text{ satisfies all inequalities } (3.1) \text{ and } (3.2)\}$$

is called *weakly bipartite*. A graph which is not weakly bipartite is called *strongly nonbipartite*. If $G=[V,E]$ is strongly nonbipartite and every subgraph of G obtained by removing one edge is weakly bipartite then G is called *minimally strongly nonbipartite*.

Every bipartite graph is of course weakly bipartite. Moreover, Barahona [2] has shown that all planar graphs are weakly bipartite. Thus, the polynomial algorithm for the max-cut problem in weakly bipartite graphs that we are going to present includes the planar max-cut problem as a special case.

It is easy to see that a graph is weakly bipartite if and only if its blocks (2-connected components) are weakly bipartite. Thus every graph whose blocks are planar or bipartite is weakly bipartite. So there are weakly bipartite graphs which are neither planar nor bipartite.

We know further graphs which are weakly bipartite and do not belong to one of the classes discussed before. One such example is the graph obtained from $K_{3,3}$ by adding one (nonmultiple) edge. This graph is a block and is neither bipartite nor planar. However, we do not know of any further 'nice' class of weakly bipartite graphs like the class of planar graphs.

Another open problem is the characterization of minimally strongly nonbipartite graphs, or even less, the determination of a large class of such graphs. We initially thought that graphs which are minimally nonbipartite-nonplanar are minimally strongly nonbipartite. But this is not the case. The graph $K_{3,3}$ plus an edge is minimally nonbipartite-nonplanar, but it is weakly bipartite.

On the other hand the minimally nonplanar graph $K_5=[V,E]$ (complete graph on five nodes) is minimally strongly nonbipartite. We can show that for K_5 the polyhedron determined by the trivial inequalities (3.1) and the odd cycle constraints (3.2) has exactly one fractional vertex, namely $x_e=\frac{2}{3}$ for all $e \in E$. It may be that all minimally strong nonbipartite graphs have the property that the polyhedron given by (3.1) and (3.2) has exactly one fractional vertex. The rank of the edge set E of K_5 is clearly six. We can show that by adding the rank inequality $x(E) \leq 6$ to the system (3.1) and (3.2) we get an integral polyhedron, i.e. this inequality system determines $P_B(K_5)$ completely.

4. Computing shortest paths of even length

Our algorithm for the max-cut problem needs a subroutine to compute shortest paths of even length. We do not claim originality for this method, rather we attribute it to 'Waterloo-folklore'. Since we do not know of any existing reference we give a short description of this algorithm here.

Suppose a graph $G=[V,E]$ with nonnegative edge weights c_e, $e \in E$, and two nodes $i, j \in V$ are given, and we want to find a shortest path from i to j with an even number of edges.

Let G_i (G_j) be the graph obtained from G by removing node i (node j). Let H_{ij} be the graph consisting of the node-disjoint union of G_i and G_j where in addition every node in G_i (except j) is linked to its copy in G_j by an edge. The edges in G_i respectively G_j keep their original weight, the weights of the new edges are set to zero. The new graph H_{ij} has an even number of nodes. We claim that the even length shortest path problem in G can be solved by calculating a minimum perfect matching in H_{ij}.

Suppose P is an even length path from i to j in G. We start with the edge e_1 of P incident with i and label the edge in G_j corresponding to e_1. Then we take the edge e_2 of P following e_1 and label the edge in G_i corresponding to e_2. We continue this way by alternately labeling edges in G_j and G_i. For every node v of G not on the path P we label the (new) edge in H_{ij} linking the two copies of v. It is obvious from the construction that the labeled edges in H_{ij} form a perfect matching with the same weight as P. On the other hand, from every perfect matching M in H_{ij} we can obtain an even length path in G from i to j having the same weight by simply skipping all 'new' edges contained in M (and possibly removing even cycles of weight zero).

It follows from the discussion above that G contains an even length path from i to j if and only if H_{ij} contains a perfect matching, and that a shortest path between i and j with an even number of edges can be computed in polynomial time using any of the existng polynomial matching algorithms, e.g. Edmonds [4].

Moreover, we would like to mention that shortest paths of odd length can be computed similarly. We construct a new graph H'_{ij} from G by taking the node disjoint union of G and the graph G_{ij} obtained from G by removing i and j, and add 'new' edges having weight zero linking the copies of the nodes different from i and j. Then we apply the perfect matching algorithm as before. By applying this procedure to every pair of different nodes we obtain:

4.1. Theorem. *Given a graph $G=[V,E]$ with nonnegative edge weights, then there is a polynomial algorithm which checks whether G contains a path of even (or odd) length and computes the shortest path of even (or odd) length between every pair of nodes, if such a path exists.*

5. The polynomial algorithm for the max-cut problem in weakly bipartite graphs

To solve the max-cut problem for weakly bipartite graphs we use the ellipsoid method as described in Grötschel, Lovász and Schrijver [7]. For any graph $G=[V,E]$ we set

$$P_C(G)=\{x\in\mathbf{R}^E \mid x \text{ satisfies inequalities } (3.1) \text{ and } (3.2)\}. \quad (5.1)$$

We shall show that for any $c\in\mathbf{Q}^E$ the linear program

$$\max c^{\mathrm{T}}x, \quad x\in P_C(G) \quad (5.2)$$

can be solved in polynomial time. Since the ellipsoid method presented in [7] delivers an optimum vertex and since for weakly bipartite graphs $P_C(G)=P_B(G)$ holds by definition, we obtain a polynomial algorithm for the max-cut problem in weakly bipartite graphs.

Note that even if G is not weakly bipartite, a solution of (5.2) may be an optimum solution of the max-cut problem. But this cannot be guaranteed. In any case, it seems that the program (5.2) can be used as a reasonable relaxation of the max-cut problem within a branch-and-bound algorithm. It was shown in [7] that the ellipsoid method runs in polynomial time if (with respect to $P_C(G)$) the following problem can be solved in polynomial time.

5.3. Separation Problem for $P_C(G)$. Let $y\in\mathbf{Q}^E$. Determine whether $y\in P_C(G)$, and if y is not in $P_C(G)$ find a vector $d\in\mathbf{Q}^E$ such that $d^{\mathrm{T}}y>d^{\mathrm{T}}x$ for all $x\in P_C(G)$.

We shall now demonstrate how the separation problem for $P_C(G)$ can be solved in polynomial time.

Suppose a vector $y\in\mathbf{Q}^E$ is given. It is trivial to check whether y satisfies the inequalities (3.1). If not, we have found a violated inequality. Thus, we may assume in the sequel that y satisfies $0\leq y_e\leq 1$ for all $e\in E$.

For every edge $e\in E$ define a 'weight' $w_e := 1-y_e$. If C is an odd cycle in G, then clearly $y(C)>|C|-1$ if and only if $w(C)<1$. This implies that we can check whether an odd cycle constraint (3.2) is violated by computing an odd cycle C^* of minimum weight $w(C^*)$. Namely, if $w(C^*)\geq 1$ then y satisfies all constraints (3.2); if $w(C^*)<1$ then $x(C^*)\leq|C^*|-1$ is the desired cutting plane.

To compute a minimum weight odd cycle we proceed as follows. We pick any edge $ij\in E$ with $w_{ij}<1$ and compute the shortest (with respect to the weights w_e) path from i to j of even length with the method described in Section 4. If the weight of this path plus the weight w_{ij} is less than one, an odd cycle C^* is found with $w(C^*)<1$. If the sum of the path weight and w_{ij} is at least one, we pick another edge and continue untill all edges with $w_{ij}<1$ have been considered. Thus, after at most $|E|$ applications of the algorithm of Section 4 we have determined whether y is in $P_C(G)$ and in case y is not we obtained a violated inequality of the form (3.2). Clearly, the overall running time of the separation algorithm for $P_C(G)$ is polynomial in the data, thus we have shown

5.4. Theorem. *There is an algorithm which for any graph $G=[V,E]$ and any $c\in\mathbf{Q}^E$ solves the linear program (5.2) $\max c^{\mathrm{T}}x$, $x\in P_C(G)$ in polynomial time. This algorithm in particular solves the max-cut problem for weakly bipartite graphs in polynomial time.*

We do not claim that the algorithm for problem (5.2) above is fast in practice. It remains a challenging problem to find a practically efficient method for the max-cut problem in weakly bipartite graphs which is of a combinatorial nature and does not suffer from the drawbacks of the ellipsoid method.

We do not know how hard the problem of recognizing weakly bipartite graphs is. The only result we have is that the decision problem "Is a given graph weakly bipartite?" is in co-NP. To prove this we have to show that "Is a graph G strongly nonbipartite?" is in NP.

This goes as follows. Suppose $|E|$ of the inequalities of the form (3.1) or (3.2) are given to us. Using Gaussian elimination we can check whether they are linearly independent or not. if yes, then the unique solution, say x, of the corrsponding $|E|$ equations gives a basic solution of the system (3.1) and (3.2). By running the separation algorithm for $P_C(G)$ described above we can then determine in polynomial time whether x is in $P_C(G)$ or not. If x is in $P_C(G)$, then x is a vertex of $P_C(G)$.

If G is strongly nonbipartite, then $P_C(G)$ has a fractional vertex, say x. Since every vertex of $P_C(G)$ is determined by $|E|$ of the inequalities (3.1) or (3.2) we can guess $|E|$ such inequalities determining x and show in polynomial time as described above that these inequalities indeed determine a fractional vertex. This proves that strongly nonbipartite graphs can be recognized in nondeterministic polynomial time.

In closing, we observe that although a minimum weight odd cycle can be computed in polynomial time, we do not know whether it is possible to find an odd hole (cycle without chord) in polynomial time. This would be of interest for checking whether a graph is perfect or not.

References

[1] F. Barahona, "Sur la complexité du problemè du verre de spins", Rapport de Recherche No. 177, Mathematiques Appliquès et Informatiques, Université Scientifique et Medicale de Grenoble, France, October 1979.

[2] F. Barahona, "On the complexity of max cut", Rapport de Recherche No. 186, Mathematiques Appliquès et Informatiques, Université Scientifique et Medicale de Grenoble, France, February 1980.

[3] F. Barahona, "Balancing signed graphs of fixed genus in polynomial time", Depto. de Matemáticas, Universidad de Chile, Santiago, Chile, June 1981.

[4] J. Edmonds, "Maximum matching and a polyhedron with (0,1)-vertices", *J. Res. Nat. Bur. Standards* **69B**, 125–130 (1965).

[5] I.S. Filotti and G.L. Miller, "On determining the genus of a graph in $O(V^{O(g)})$ steps", *Proc. 10th Annual ACM Symposium on Theory of Computing*, Association of Computing Machinery, New York (1978).

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

[7] M. Grötschel, L. Lovász and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization", *Combinatorica* 1 (1981).

[8] F.O. Hadlock, "Finding a maximum cut of a planar graph in polynomial time", *SIAM J. Comput* **4**, 221–225 (1975).

[9] G.I. Orlova and Y.G. Dorfman, "Finding the maximum cut in a planar graph", *Engrg. Cybernetics* **10**, 502–506 (1975).

[10] M. Yannakakis, "Node- and edge-deletion NP-complete problems", *Proc. 10th Annual ACM Symposium on Theory of Computing*, Association of Computing Machinery, New York (1978) 253–264.