WORKING PAPER No.204

Easy with Difficulty Objective Functions for Max Cut

by

S. Thomas McCormick
M.Rammohan Rao &
Giovanni Rinaldi

December 2002

Please address all correspondence to:

Prof. M. Rammohan Rao
Professor
Indian Institute of Management Bangalore
Bannerghatta Road
Bangalore – 560076, India
E-mail: mrao@iimb.ernet.in
Phone : 080 – 6993136
Fax : 080 - 6584050

Math. Program., Ser. B [redacted]: [redacted]–[redacted] (2002)

Digital Object Identifier (DOI) 10.1007/s10107-002-0328-8

S. Thomas McCormick* · M.R. Rao · Giovanni Rinaldi

Easy and difficult objective functions for max cut

Received: November 29, 2000 / Accepted: August 17, 2001
Published online: [redacted] –

Abstract. This note investigates the boundary between polynomially-solvable Max Cut and NP Hard Max Cut instances when they are classified only on the basis of the sign pattern of the objective function coefficients, i.e., of the orthant containing the objective function vector. It turns out that the matching number of the subgraph induced by the positive edges is the key parameter that allows us to differentiate between polynomially-solvable and hard instances of the problem. We give some applications of the polynomially solvable cases.

A *cut* in a weighted undirected graph $G = (V, E, w)$ is defined by a node set $S \subseteq V$ as the set of all edges with exactly one endpoint in S, and it is denoted by $\delta(S)$. The weight of a cut is the sum of the weights of its edges. Given a weighted graph G, the *Max Cut Problem* is to find a cut of G with maximum weight, while the *Min Cut Problem* is to find a cut with minimum weight. We call $\delta(\{v\})$ the *star* of v, and denote $|V|$ by n.

There has been a lot of interest in Max Cut problems lately. These problems have many applications, but are very hard to solve. They are not only NP Hard, but they are difficult to solve exactly in practice as well. A few special cases of Max Cut that are solvable in polynomial time are described in the literature. Most of them require the graph to have a special structure. This is for example the case for planar graphs, graphs having fixed bandwidth, graphs with bounded tree-width, or graphs with bounded genus. For a list of references, see, for example [4] and [9].

It is also interesting to investigate conditions on the objective function under which Max Cut can be solved in polynomial time. One case where it is easy to see that Max Cut is polynomially solvable is when all edge weights are non-positive. In this case the problem is trivially solved, as the empty set is always an optimal solution. Galluccio and Loebl in [5] give a class of instances characterized by both the structure of the graph and the size of the objective function coefficients for which Max Cut is polynomially solvable.

In this note we investigate some other polynomially-solvable instances of Max Cut that are characterized by properties of the objective function rather than by properties of G. In particular we are interested in classifying instances based on the signs of the

S. Thomas McCormick: Faculty of Commerce and Business Administration, University of British Columbia, Vancouver, BC V6T 1Z2, Canada, e-mail: stmv@adk.commerce.ubc.ca

M.R. Rao: Indian Institute of Management, Bannerghatta Road, Bangalore 560076, India, e-mail: mrao@iimb.ernet.in

G. Rinaldi: Istituto di Analisi dei Sistemi ed Informatica del CNR, Viale Manzoni 30, 00185 Roma, Italy, e-mail: rinaldi@iasi.rm.cnr.it

* The research of this author was partially supported by an NSERC Research Grant.

objective function coefficients, i.e., on the orthant that contains the objective function vector. For a given sign pattern (orthant) of the objective function coefficients, we would like to know if Max Cut is polynomially solvable no matter how the magnitude of the coefficients is chosen, or if it can be proven that this class of instances is NP Hard. To this purpose, for a given weight function $w = \{w_{ij}\}_{\{i,j\}\in E}$, we define E_w^+ to be the set of *positive edges* and E_w^- to be the set of *negative edges*, i.e., the set of edges whose corresponding components of w are strictly positive or strictly negative, respectively. From now on we do not make any assumption on the magnitude of the coefficients of w.

Note that Max Cut is equivalent to Min Cut by negating weights. Therefore, from now on we consider only the minimization version of the problem.

A slight variation of Min Cut that we name *Min Proper Cut* calls for a *nonempty* cut of minimum weight. If E_w^- is empty, Min Proper Cut is polynomially solvable, for example, by the Gomory-Hu algorithm [7].

Another variation of the problem is the *Min s-t Cut*: Here s and t are two distinguished nodes of V and we want an s-t cut, i.e., a cut $\delta(S)$ such that S contains only one of s and t, of minimum weight. Again, when E_w^- is empty this problem is polynomially solvable by max flow (see, e.g., [1]).

The *cover number* $c(V, E_w^-)$ of the subgraph with edges E_w^- is the smallest size of a node subset X such that each edge of E_W^- has an endpoint in X. We call a class of weighted graphs *nearly positive* if $c(V, E_w^-) = O(\log n^k)$ for some fixed $k > 0$ for each graph in the class. For such a class the next section shows that Min Cut, Min Proper Cut, and Min s-t Cut are solvable in polynomial time. This fact means that it is interesting to delineate the boundary between polynomially-solvable special cases of Min Cut and NP Hard cases. Later we show that the Min Cut Problem is strongly NP complete for a class of graphs with $c(V, E_w^-) = \Omega(n^{1/k})$ for a fixed constant $k > 0$, a natural slight generalization of nearly positive graphs. We end by giving several applications where our ideas lead to polynomial algorithms.

Min cut on nearly positive graphs

We call a weighted graph *s-t negative* if E_w^- is a subset of $\delta(\{s\}) \cup \delta(\{t\})$. Note that s-t negative graphs are a special case of nearly positive graphs since $c(V, E_w^-) \leq 2$ in this case.

Theorem 1. *Min s-t Cut is solvable in polynomial time for s-t negative weighted graphs.*

Proof. By adding edges of weight 0 we can assume, without loss of generality, that E contains the edges $\{s, i\}$ and $\{i, t\}$ for all $i \in V \setminus \{s, t\}$. As every s-t cut contains exactly one of each pair of edges $\{s, i\}$ and $\{i, t\}$ for all $i \in V \setminus \{s, t\}$, we can add an arbitrary constant to w_{si} and w_{it} to make them both nonnegative. Analogously, if $\{s, t\} \in E$ and $w_{st} < 0$ we can add an arbitrary constant to w_{st} to make it nonnegative. The resulting weighted graph has nonnegative weights and therefore its minimum weight s-t cut can be found in polynomial time. □

The next special case of Min Cut is when all edges of E_w^- are incident to a single node s (the graph is called *star negative* in this case), which we call *Negative Star Min*

(Proper) Cut. Here we use the following simple construction to reduce Min Cut and Min Proper Cut on such a graph to single Min s-t Cut and a polynomial number (at most $n-1$) of Min s-t Cut problems, respectively, on s-t negative graphs. This will show that Negative Star Min (Proper) Cut is polynomial.

Adjoin a new node t to the original graph G and connect t to all the nodes of G by edges of weight zero to get a new graph G' which is s-t negative. Now we can compute a minimum weight s-t cut $\delta(S')$ in G' in polynomial time, and set $T' = V \setminus S'$. If $T' \neq \{t\}$ then the corresponding cut $\delta(S')$ in G must be a minimum weight proper cut of G. So assume instead that $T' = \{t\}$. If we are interested in solving Min Cut, we are done, since we know that in this case the empty cut is optimal. Otherwise, for each node j such that edge $\{j, t\}$ is in G', temporarily set $w_{jt} = \infty$, and compute a minimum weight s-t cut $\delta(S'_j)$. Certainly $j \notin S'_j$. Let k achieve the minimum in $\min_j w(\delta(S'_j))$. Then it is easy to see that $\delta(S'_k)$ must be a minimum weight cut of G.

It is now easy to solve Min (Proper) Cut in polynomial time for a s-t negative weighted graph G: A minimum (proper) cut $\delta(W)$ of G either separates s from t, thus is a minimum s-t cut of G, or otherwise can be found by computing the minimum (proper) cut in the star negative graph obtained from G by contracting s and t.

In the general case for nearly positive graph, E_w^- can be covered with the stars of the nodes of a set $M \subseteq V$ with $|M| = O(\log n^k) = O(\log n)$. Let $\delta(W)$ be a minimum weight cut of G and let $S = W \cap M$ and $T = (V \setminus W) \cap M$. If either one of S and T is empty, say T, we can consider a graph G' obtained from G by contracting all nodes of S into a single node s. That is, to get G' we remove the nodes of S from G and add the new node s, and we connect s to every node j such that there was an edge $\{j, i\}$ with $i \in S$. This graph is star negative and if $\delta(W')$ is a minimum weight cut of G' with $s \in W'$, then $\delta(S \cup W' \setminus \{s\})$ must be a minimum cut of G.

Similarly, if both S and T are nonempty we can consider a graph G' obtained from G by contracting all nodes of S and T into single nodes s and t, respectively. This graph is s-t negative and if $\delta(W')$ is a minimum weight cut of G', then $\delta(S \cup W' \setminus \{s\})$ must be a minimum cut of G.

There are $2^{O(\log n)}$ (a polynomial in n) possible ways the set M can intersect W. Thus we only have to enumerate each of this possibilities, compute the corresponding cut as suggested above, and take the minimum. This establishes:

Theorem 2. *Min Cut, Min s-t Cut, and Min Proper Cut are all polynomially solvable on nearly positive graphs.*

Negative cut min cut

In Negative Star Min Cut E_w^- is contained in the cut $\delta(\{s\})$. Thus a natural generalization to consider is the case when E_w^- is contained in a (general) cut. We refer to this case as *Negative Cut Min Cut.* One may speculate that this case is also polynomially solvable. On the contrary, there is a simple argument that shows that the problem is NP Hard.

In any instance of Negative Cut Min Cut the set V is partitioned into two subsets U and W and all edges with both endpoints in U or in W have non-negative weight, while all the others have arbitrary weights. Clearly any bipartite graph with arbitrary edge weights is of this kind. C. De Simone [2] showed that Max Cut is NP Hard for bipartite graphs with arbitrary weights. We report here the simple proof in terms of Min Cut.

Lemma 3. *Min Cut is NP Hard in bipartite graphs with arbitrary edge weights.*

Proof. Let $G' = (V', E')$ be the bipartite graph with weights w' obtained from G by subdividing each edge $\{i, j\}$ with one new node v. The weight of the edges $\{i, v\}$ and $\{v, j\}$ are given by $w'_{iv} = w_{ij}$ and $w'_{vj} = |w_{ij}|$. We claim that there exists a minimum cut $K = \delta(S)$ of G' having the following two properties:

1. for each edge $\{i, j\}$ in E at most one of the two edges $\{i, v\}$ and $\{v, j\}$ belongs to K; and
2. if $\{v, j\}$ belongs to K, then $w'_{vj} = w_{ij}$.

Suppose, without loss of generality, that S contains j. Property 1 holds because if both $\{i, v\}$ and $\{v, j\}$ belong to K, then i also belongs to S. Consequently, including v in S defines a cut which does not include both the edges $\{i, v\}$ and $\{v, j\}$ and the weight of this cut is no greater than the weight of K. Property 2 holds because otherwise $w'_{vj} = -w_{ij} > 0$. Adding v to S will replace $-w_{ij} > 0$ by $w_{ij} < 0$, decreasing the weight of the cut.

Cut K has the same weight as the corresponding cut in G, so solving Min Cut on G' is equivalent to solving Min Cut on G. □

Negative matching min cut

In the NP Hardness proof of Lemma 3 $|E_w^-|$ could be as large as $\Theta(n2)$, while in all the polynomially-solvable instances that we have analyzed so far $|E_w^-| = O(n)$. Thus it is tempting to conjecture that Min Cut is polynomially solvable for any weighted graph with $|E_w^-| = O(n)$. The following lemma shows that this is not true.

Consider instances where the edges in E_w^- form a matching, which we call *Negative Matching Min Cut.*

Lemma 4. *Negative Matching Min Cut is strongly NP Complete.*

Proof. The reduction will be from Min 2-Sat [8]. An instance of Min 2-Sat has a set of logical variables $L = \{x_1, x_2, \ldots, x_p\}$; variable x_i gives rise to literals x_i and $\bar{x}_i$. We also have a set of clauses $C_1, C_2, \ldots, C_m$, where each C_j is a two-element subset of literals. The question is whether there is a truth assignment such that at most k clauses are true. Since Min 2-Sat is trivial if $k = m$, we can assume that $k < m$.

We now construct an instance of Negative Matching Min Cut from this instance of Min 2-Sat. Make a node for each literal with the same name as the literal, and a single node 0 to represent all the clauses. Connect literal nodes x_i and $\bar{x}_i$ by a *variable edge* of weight $-2m$; these edges form the negative matching. For clause $C_j = \{l_1, l_2\}$, make a *triangle* of three edges, $\{0, l_1\}$, $\{l_1, l_2\}$, and $\{l_2, 0\}$, all of weight 1 (note that this is likely to create multiple edges $\{0, l\}$ for most literals l). Finally, set the target weight for Negative Matching Min Cut to be $2k - 2pm$. Note that all numbers here are strongly polynomial, so this reduction will show *strong* NP Completeness.

We now claim that there is a truth assignment having at most k satisfied clauses if and only if there is a minimum cut having weight at most $2k-2pm$. Suppose that τ is a truth assignment having at most k satisfied clauses. Define S to be the node subset consisting of 0 and all the false literals under τ, and T to be the node subset containing all the true literals. Then C_j's triangle of edges crosses $\delta(S)$ if and only if C_j is true under τ, and when this triangle does cross $\delta(S)$ it contributes weight exactly 2. Thus contribution of the clause triangles to the cut weight is twice the number of satisfied clauses under τ, which is at most $2k$. Every variable edge crosses $\delta(S)$, contributing $-2pm$ to the cut weight, for a total cut weight of at most $2k-2pm$.

Suppose now that $\delta(S)$ is a cut of weight at most $2k-2pm$ with $0 \in S$. Since $k < m$, $\delta(S)$ must cut every variable edge. Thus $\delta(S)$ defines a truth assignment τ, where $\tau(l)$ is true if $l \notin S$. Then C_j is true under τ if and only if two edges of C_j's triangle are cut by $\delta(S)$. Since $\delta(S)$ has weight at most $2k-2pm$, at most k of the clauses can be true under τ. □

Define the *matching number* $m(V, E_w^-)$ to be the size of a maximum cardinality matching in the subgraph (V, E_w^-).

Corollary 5. *For any fixed k independent of n, the problems Min Cut, Proper Min Cut, and Min s-t Cut are all strongly NP Hard to solve in classes of graphs with* $m(V, E_w^-) = \Omega(n^{1/k})$.

Proof. Since adding negative edges can only make instances harder, it suffices to consider instances where the negative edges form a matching.

Suppose that we adjoin n^k nodes to V to get V', and connect them to the rest of the graph by zero-weight edges in the construction of Lemma 4. Then this is still a strongly polynomial-time reduction, and there are only $O(|V'|^{1/k})$ negative edges. For Proper Min Cut and s-t Min Cut, note that for any negative matching edge, its ends must appear on opposite sides of any minimum weight cut, so these problems are Strongly NP Hard also. □

The border line

Theorem 2 implies that Min Cut is polynomially solvable if $c(V, E_w^-)$ is $O(\log n^k)$ for some fixed k. By contrast, Corollary 5 shows that if $m(V, E_w^-)$ is $\Omega(n^{1/k})$ for some fixed k, the problem is strongly NP Hard. It is easy to see that $m(V, E_w^-) \leq c(V, E_w^-) \leq 2m(V, E_w^-)$, so that $c(V, E_w^-)$ and $m(V, E_w^-)$ are the same up to a small constant factor. This establishes the following fairly sharp characterization of the boundary between polynomially-solvable and NP Hard instances of Min Cut:

Theorem 6. *For any fixed integer* $k > 0$, *Min Cut is polynomially solvable for classes of weighted graphs where* $m(V, E_w^-) = O(\log n^k)$; *for classes where* $m(V, E_w^-) = \Omega(n^{1/k})$ *it is strongly NP Hard.*

Therefore, if we classify the Min Cut instances only on the basis of the edge set E_w^-, the structural property of such a set that allows us to differentiate between easy and difficult instances is the matching number of (V, E_w^-).

There is a fairly small unresolved gap here between the easy cases at $O(\log n^k)$ and the hard cases at $\Omega(n^{1/k})$ for fixed k. Given that the polynomial algorithm for small independence number depends on an explicit enumeration that is exponential in the independence number, it seems unlikely that polynomial algorithms exist above an independence number of $O(\log n^k)$. On the other hand, it appears to be difficult to adapt the proof of Lemma 4 to any function of n smaller than $\Omega(n^{1/k})$ for fixed k. It seems that it would require a more sophisticated proof to narrow this gap.

Applications

We give here several applications that Theorem 2 shows can be solved in polynomial time.

1. Polynomial separation of the fractional capacity inequalities of the CVRP. The *Capacitated Vehicle Routing Problem* (CVRP) can be stated as follows. Given are two positive integers k and C and a graph $G = (V, E)$ with edge weights w, node weights d, and a distinguished node s with $d_s = 0$. A feasible solution to the problem (called *feasible k-route*) is a set of k simple cycles, all containing node s and having no other node in common, that spans G and is such that the d weight of each cycle is bounded by C. The degenerate case of a two node cycle is allowed. The problem is to find a minimum w weight feasible k-route.

An integer formulation of CVRP (see, e.g., [13] for the details) whose feasible solutions are the incidence vectors of the feasible k-routes is given by:

$$\begin{array}{lll}
\min \sum_{e\in E} c_e x_e & & \\
\text{s.t.} & & \\
\quad x(\delta(i)) = 2 & \text{for } i \in V \setminus \{s\} & \\
\quad x(\delta(\{s\})) = 2k & & \\
\quad x(\delta(W)) \geq 2\frac{d(W)}{C} & \text{for } \emptyset \neq W \subseteq V \setminus \{s\} & \qquad (1)\\
\quad 0 \leq x_e \leq 1 & \text{for } e \in E \setminus \delta(\{s\}) & \\
\quad 0 \leq x_e \leq 2 & \text{for } e \in \delta(\{s\}) & \\
\quad x_e \text{ integer} & \text{for } e \in E &
\end{array}$$

All the sets of linear constraints in (1) have size that is polynomial in n except the third one, which are the so-called *fractional capacity inequalities*, whose size is 2^{n-1}. When the vector x is integer, the fractional capacity inequalities guarantee that the cocycle of each set W intersects at least as many simple cycles of the corresponding k-route as $\frac{d(W)}{C}$.

The linear programming relaxation to (1) can thus be solved in polynomial time, provided that the separation problem for the fractional capacity inequalities is also solvable in polynomial time. Therefore, we want to find a polynomial time algorithm that solves the following problem:

Given a nonnegative weight vector $\bar{x}$ for the graph G, find a fractional capacity inequality violated by this vector or prove that no such an inequality exists.

This problem can be solved in following way. Assume, by possibly adding zero weight edges, that the star of node s is complete. For each $i \in V \setminus \{s\}$ replace the weight $\bar{x}_{si}$ by $\bar{x}_{si} - 2d_i/C$. The resulting graph is clearly star negative and its minimum cut $\delta(W^*)$ has negative weight if and only if there is a fractional capacity inequality (namely the one defined by W^*) violated by $\bar{x}$.

Based on this algorithm, in [3] a polynomial algorithm that solve the separation for a strengthened version of the fractional inequalities is described. These inequalities, called the *rounded capacity inequalities*, are obtained from their fractional counterpart by rounding the right hand side to the nearest larger integer.

2. *Cuts with Two Sets of Weights.* Suppose that we have positive weights p and d on the edges. Here are three questions we might ask:

Min Ratio Cut We want to solve $\min_{W \subseteq V} p(\delta(W))/d(\delta(W))$ (where $p(\delta(W))$ is defined as $\sum_{\{i,j\} \in \delta(W)} p_{ij}$), i.e., we want to find a fractional Min Cut. (See, e.g., [10] for a survey of fractional cut applications and algorithms.)
A standard algorithm in such cases is to multiply the denominator by parameter λ and subtract it from the numerator, and so to consider the linear objective function

$$\min_{W \subseteq V} p(\delta(W)) - \lambda d(\delta(W)).$$

We can solve this by finding a Min Cut with the weight function $p - \lambda d$. If this linear objective can be optimized in polynomial time, Radzik [11], [12] shows how to use a discrete version of Newton's Algorithm to compute an optimal solution to the fractional problem in polynomial time.

Does p cut-dominate d We want to answer the question of whether $p(\delta(W)) \geq d(\delta(W))$ for all $W \subseteq V$. That is, is the weight of every cut under p at least as great as its weight under d? We can answer this by finding a Min Cut with the weight function $p - d$.

Ratio of products We want to solve

$$\min_{W \subseteq V} \frac{\prod_{e \in \delta(W)} p_e}{\prod_{e \in \delta(W)} d_e}.$$

Taking logs, this becomes $\min_{W \subseteq V} \sum_{e \in \delta(W)} \log p_e - \sum_{e \in \delta(W)} \log d_e$, so we can answer the question by finding a Min Cut with weights $\log p - \log d$.

In all three cases, we end up with a graph with a difference of weights on its edges. If the subgraph with negative weights has a small matching number, then we can solve this in polynomial time.

3. *Min Cut with forced edges.* Suppose that we have k pairs of nodes $\{s_1, t_1\}$, $\{s_2, t_2\}$, $\ldots$, $\{s_k, t_k\}$, where each s_i is different from every s_j and t_j. We want to solve Min Cut with the added constraint that we require that the cut $\delta(W)$ be such that W contains exactly one of s_i and t_i for $i = 1, \ldots, k$.

We can force this to be true by adding in edges $\{s_i, t_i\}$ for $i = 1, 2, \ldots, k$ with weights $w_{s_i, t_i} = -M$ for a suitably large M. Then, as long as $k = O(\log n)$, we see that we can solve this in polynomial time.

Acknowledgements. We thank the two referees for their helpful comments.

References

[1] Ahuja, R.K., T.L. Magnanti, and J.B. Orlin (1993). *Network Flows: Theory, Algorithms, and Applications*. (Prentice-Hall, Englewood Cliffs).
[2] De Simone, C. (1996). Private communication.
[3] De Vitis, A., M. Queyranne and G. Rinaldi (2001). *Separating the capacity inequalities of the vehicle routing problem in polynomial time*. IASI-CNR Research Report. In preparation.
[4] Deza, M.M. and M. Laurent (1997). *Geometry of Cuts and Metrics*. Springer, Berlin.
[5] Galluccio, A., M. Loebl and J. Vondrák (1999). *Optimization via enumeration: A new algorithm for the MAX-CUT problem*. IASI-CNR Research Report 507. To appear in *Mathematical Programming*.
[6] Garey, M.R. and D.S. Johnson (1979). *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York.
[7] Gomory, R.E. and T.C. Hu (1961). Multi-terminal network flows. *SIAM Journal of Applied Mathematics*, **9**, 551–570.
[8] Kohli, R., R. Khrishnamurti, and P. Mirchandani (1994). The Minimum Satisfiability Problem. *SIAM Journal of Discrete Mathematics*, **7**, 275–283.
[9] Laurent, M. (1997). *The max-cut problem*. In M. Dell'Amico, F. Maffioli, and S. Martello, eds., Annotated Bibliographies in Combinatorial Optimization. Wiley, Chichester, 1997.
[10] McCormick, S.T., and T.R. Ervolina (1994). Computing maximum mean cuts. *Discrete Applied Mathematics* **52**, 53–70.
[11] Radzik, T. (1992). Newton's method for fractional combinatorial optimization. *Proceedings of the 33rd IEEE Annual Symposium of Foundations of Computer Science*, pp. 659–669.
[12] Radzik, T. (1993). Parametric flows, weighted means of cuts, and fractional combinatorial optimization. In *Complexity in Numerical Optimization*, P. Pardalos, ed., World Scientific, pp. 351–386.
[13] Toth, P. and D. Vigo (2001), eds.. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications. Forthcoming.