A 0.5-APPROXIMATION ALGORITHM FOR MAX DICUT WITH GIVEN SIZES OF PARTS

ALEXANDER AGEEV*, REFAEL HASSIN†, AND MAXIM SVIRIDENKO‡

Abstract. Given a directed graph G and an arc weight function $w : E(G) \to \mathbb{R}_+$, the maximum directed cut problem (MAX DICUT) is that of finding a directed cut $\delta(X)$ with maximum total weight. In this paper we consider a version of MAX DICUT—MAX DICUT with given sizes of parts or MAX DICUT WITH GSP—whose instance is that of MAX DICUT plus a positive integer p, and it is required to find a directed cut $\delta(X)$ having maximum weight over all cuts $\delta(X)$ with $|X| = p$. Our main result is an 0.5-approximation algorithm for solving the problem. The algorithm is based on a tricky application of the pipage rounding technique developed in some earlier papers by two of the authors and a remarkable structural property of basic solutions to a linear relaxation. The property is that each component of any basic solution is an element of a set $\{0, \delta, 1/2, 1-\delta, 1\}$ where δ is a constant that satisfies $0 < \delta < 1/2$ and is the same for all components.

Key words. approximation algorithm, directed cut, linear relaxation, basic solution

AMS subject classifications. 68W25, 05C85, 90C27, 90C35

1. Introduction. Let G be a directed graph. A directed cut in G is defined to be the set of arcs leaving some vertex subset X (we denote it by $\delta(X)$). Given a directed graph G and an arc weight function $w : E(G) \to \mathbb{R}_+$, the maximum directed cut problem (MAX DICUT) is that of finding a directed cut $\delta(X)$ with maximum total weight. In this paper we consider a version of MAX DICUT (MAX DICUT with given sizes of parts or MAX DICUT WITH GSP) whose instance is that of MAX DICUT plus a positive integer p, and it is required to find a directed cut $\delta(X)$ having maximum weight over all cuts $\delta(X)$ with $|X| = p$. MAX DICUT is well-known to be NP-hard and so is MAX DICUT WITH GSP as the former evidently reduces to the latter.

The NP-hardness of MAX DICUT follows from the observation that the well-known undirected version of MAX DICUT — the maximum cut problem (MAX CUT), which is on the original Karp's list of NP-complete problems [Ka72] — reduces to MAX DICUT by substituting each edge for two oppositely oriented arcs. This means that for both problems there is no choice but to develop approximation algorithms. Nevertheless, this task turned out to be highly nontrivial, as for a long time it was an open problem whether it is possible to design approximations with factors better than trivial 1/2 for MAX CUT and 1/4 for MAX DICUT. Only quite recently, using a novel technique of rounding semidefinite relaxations Goemans and Williamson [GW95] worked out algorithms solving MAX CUT and MAX DICUT approximately within factors of 0.878 and 0.796 respectively. A bit later Feige and Goemans [FG95] developed an algorithm for MAX DICUT with a better approximation ratio of 0.859. Recently, using a new method of rounding linear relaxations—the pipage rounding—Ageev and Sviridenko [AS99] developed an 0.5-approximation algorithm for the version of MAX CUT in which the parts of a vertex set bipartition are constrained to have given sizes (MAX CUT with given sizes of parts or MAX CUT WITH GSP). Later Hassin and Rubinstein [HR00]

*Sobolev Institute of Mathematics, Novosibirsk, Russia (ageev@math.nsc.ru). The research of this author was partially supported by the Russian Foundation for Basic Research, grant 99-01-00601.

†School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel (hassin@math.tau.ac.il).

‡IBM T. J. Watson Research Center, Yorktown Heights, NY (sviri@us.ibm.com). The research of this author was partially supported by the Russian Foundation for Basic Research, grant 99-01-00510.

presented a different 0.5-approximation with a better running time. The paper [AS00] presents an extension of the algorithm in [AS99] to a hypergraph generalization of MAX CUT WITH GSP. Feige and Langberg [FL99] combined the method in [AS99] with the semidefinite programming approach to design an $0.5+\varepsilon$-approximation for MAX CUT WITH GSP where ε is some unspecified small positive number.

It is easy to see that MAX CUT WITH GSP reduces to MAX DICUT WITH GSP in the same way as MAX DICUT reduces to MAX CUT. However, unlike MAX CUT WITH GSP, MAX DICUT WITH GSP provides no possibilities for a straightforward application of the pipage rounding since the F/L lower bound condition in the description of the method (see § 2) does not hold in general.

Fortunately, the other main condition, ε-convexity, always holds. In the final section of this paper, we show that the F/L lower bound condition is still satisfied with $C = 0.5$ in the case when the arc weights form a circulation in the given graph as well as when the parts of a cut are restricted to have the same size (the DIGRAPH BISECTION problem). Thus, these cases can be approximated within a factor of 0.5 by the direct application of the pipage rounding method.

The main result of this paper is an algorithm that finds a feasible dicut of weight within a factor of 0.5 in the case of arbitrary weights. It turns out that to construct such an algorithm one needs to carry out a more profound study of the problem structure. A heaven-sent opportunity is provided by a remarkable structural property of basic solutions to a linear relaxation (Theorem 4.1). At this point we should notice the papers of Jain [Ja98], and Melkonian and Tardos [MT99] where exploiting structural properties of basic solutions was also crucial in designing better approximations for some network design problems.

The resulting algorithm (DIRCUT) is of rounding type and as such consists of two phases: the first phase is to find an optimal (fractional) solution to a linear relaxation; the second (rounding) phase is to transform this solution to a feasible (integral) solution. A special feature of the rounding phase is that it uses two different rounding algorithms (ROUND1 and ROUND2) based on the pipage rounding method and takes the best solution for the output. The worst-case analysis of the algorithm heavily relies on Theorem 4.1.

2. Pipage rounding: a general scheme. In this section, to make the paper self-contained, we give a general description of the pipage rounding method as it was presented in [AS99].

Assume that a problem P can be formulated as the following nonlinear binary program:

$$
\begin{aligned}
\max\quad & F(x) && (2.1)\\
\text{s. t.}\quad & \sum_{i=1}^{n} x_i = p, && (2.2)\\
& 0 \le x_i \le 1, \quad i = 1,\dots,n, && (2.3)\\
& x_i \in \{0,1\}, \quad i = 1,\dots,n && (2.4)
\end{aligned}
$$

where p is a positive integer, $F(x)$ is a function defined on the rational points $x = (x_i)$ of the n-dimensional cube $[0,1]^n$ and computable in polynomial time. Assume further that one can associate with $F(x)$ another function $L(x)$ which is defined on the same set, coincides with $F(x)$ on binary x satisfying (2.2), and the program

$$\max L(x) \tag{2.5}$$

$$\text{s. t.} \quad \sum_{i=1}^{n} x_i = p \tag{2.6}$$

$$0 \le x_i \le 1, \quad i = 1, \dots, n, \tag{2.7}$$

which we call *a nice relaxation,* is polynomially solvable. Assume next that the following two main conditions hold:

F/L lower bound condition: there exists a constant C such that $0 < C \le 1$ and $F(x) \ge CL(x)$ for each rational point x in $[0,1]^n$;

ε-convexity condition: the function

$$\varphi(\varepsilon, x, i, j) = F(x_1, \dots, x_i + \varepsilon, \dots, x_j - \varepsilon, \dots, x_n) \tag{2.8}$$

is convex with respect to $\varepsilon \in [-\min\{x_i, 1 - x_j\}, \min\{1 - x_i, x_j\}]$ for each pair of indices i and j and each $x \in [0,1]^n$.

We describe now *the pipage rounding* procedure. Its input is a fractional solution x satisfying (2.2)–(2.3) and its output is an integral solution x satisfying (2.2)–(2.4) and having the property that $F(x) \ge F(\tilde{x})$. The pipage rounding consists of uniform 'pipage steps'. We describe the first step. If the solution x is not binary, then due to (2.2) it has at least two different components x_i and x_j with values lying strictly between 0 and 1. By ε-convexity condition, $\varphi(\varepsilon, x, i, j) \ge F(x)$ either for $\varepsilon = \min\{1 - x_i, x_j\}$, or for $\varepsilon = -\min\{x_i, 1 - x_j\}$. Thus we obtain a new feasible solution $x' = (x_1, \dots, x_i + \varepsilon, \dots, x_j - \varepsilon, \dots, x_n)$ with smaller number of non-integer components and such that $F(x') \ge F(x)$. After repeating the 'pipage' step at most $n - 1$ times we arrive at a binary feasible solution $\tilde{x}$ with $F(\tilde{x}) \ge F(x)$. Since each step can be performed in polynomial time, the overall running time of the described procedure is polynomially bounded.

Now suppose that x is an optimal solution to (2.5)-(2.7), satisfying both ε-convexity and F/L lower bound condition. Then $F(\tilde{x}) \ge F(x) \ge CL(x) \ge CF^*$, where F^* is the optimal value of (2.1)–(2.4). Thus the algorithm consisting of a polynomial-time procedure to solve the nice relaxation (2.5)-(2.7) and the pipage rounding, finds a feasible solution to (2.1)–(2.4) of weight within a factor of C of the optimum. Note that if instead of a procedure for solving (2.5)-(2.7) we use *any* polynomial-time procedure to find a solution x satisfying (2.2)–(2.3) and $F(x) \ge CF^*$, then we also obtain a C-approximation algorithm.

3. Application: max dicut with gsp. In this section, we show an implementation of the above scheme in the case of MAX DICUT WITH GSP and, on the side, specify the character of obstacles to the direct application of the pipage rounding method.

In what follows $G = (V, A)$ stands for the graph in the input of MAX DICUT WITH GSP . Since assigning zero weights to missing arcs yields an equivalent problem, we may assume that A contains all possible arcs. Let $|V| = n$.

First, note that MAX DICUT WITH GSP can be formulated as the following nonlinear binary program:

$$\begin{aligned} \max \quad & F(x) = \sum_{ij \in A} w_{ij} x_i (1 - x_j) \\ \text{s. t.} \quad & \sum_{i \in V} x_i = p, \\ & x_i \in \{0, 1\}, \quad \forall i \in V. \end{aligned}$$

Second, just like MAX CUT WITH GSP in [AS99], MAX DICUT WITH GSP can be formulated as the following integer program:

$$\max \quad \sum_{ij \in A} w_{ij} z_{ij} \tag{3.1}$$

$$\text{s. t.} \quad z_{ij} \le x_i \text{ for all } ij \in A, \tag{3.2}$$

$$z_{ij} \le 1 - x_j \text{ for all } ij \in A, \tag{3.3}$$

$$\sum_i x_i = p, \tag{3.4}$$

$$0 \le x_i \le 1 \text{ for all } i \in V, \tag{3.5}$$

$$x_i, z_{kj} \in \{0, 1\} \text{ for all } i \in V, kj \in A. \tag{3.6}$$

Observe now that the variables z_{ij} can be excluded from (3.1)–(3.5) by setting

$$z_{ij} = \min\{x_i, (1 - x_j)\} \text{ for all } ij \in A.$$

Hence (3.1)–(3.5) is equivalent to maximizing

$$L(x) = \sum_{ij \in A} w_{ij} \min\{x_i, (1 - x_j)\} \tag{3.7}$$

subject to (3.4),(3.5).

Thus we have functions F and L that can be considered as those involved in the description of the pipage rounding (see § 2). Notice now that for each pair of indices i and j the function $\varphi(\varepsilon, x, i, j)$ defined by (2.8) is the sum of $w_{ij}(x_i + \varepsilon)[1 - (x_j - \varepsilon)] + w_{ji}(x_j - \varepsilon)[1 - (x_i + \varepsilon)]$ and a term linear in ε. It follows that $\varphi(\varepsilon, x, i, j)$ is a quadratic polynomial in ε having a nonnegative leading coefficient for each pair of indices i and j and each $x \in [0, 1]^n$. Thus, the function F obeys the ε-convexity condition. Unfortunately, the other, F/L lower bound condition does not hold for any $C > 0$. We present below an example showing that the ratio $F(x)/L(x)$ may be arbitrarily close to 0 even when the underlying graph is bipartite.

Example 1. Consider the following instance of MAX DICUT WITH GSP . Let $V = V_1 \cup V_2 \cup V_3$ where $|V_1| = k$, $|V_2| = |V_3| = 2$. Let $A = A_1 \cup A_2$ where A_1 is the set of $2k$ arcs from V_1 to V_2 inducing a complete bipartite graph on (V_1, V_2) and A_2 is the set of 4 arcs from V_2 to V_3 inducing a complete bipartite graph on (V_2, V_3) (see Fig. 3.1). Assume the arc weights to be units and $p \le k$.

Then for any feasible solution x,

$$\begin{aligned} L(x) &= \sum_{ij \in A} \min\{x_i, 1 - x_j\} \le \sum_{ij \in A} x_i \\ &= 2 \sum_{i \in V_1} x_i + 2 \sum_{i \in V_2} x_i \le 2p. \end{aligned}$$

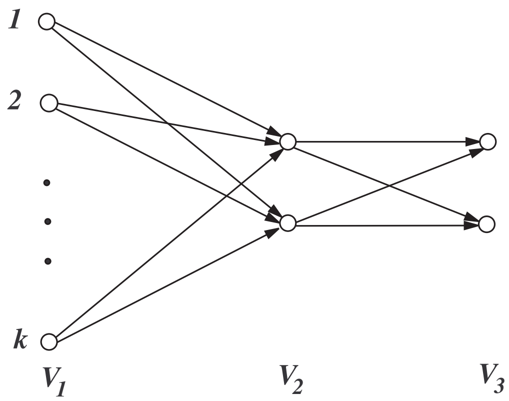

FIG. 3.1. *An example demonstrating that the ratio $F(x)/L(x)$ may be arbitrarily close to* 0.

In fact, $2p$ is the optimal value of the nice relaxation and it can be obtained in more than one way. One way is to let $x_i = r = \frac{p-2}{k-2}$ for $i \in V_1$, $x_i = 1 - r$ for $i \in V_2$ and $x_i = 0$ for $i \in V_3$. Then $F(x) = 2kr^2 + 4(1-r)$. Now if p/k tends to 0 as p tends to infinity (for example, set $k = p^2$), $F(x)/L(x)$ tends to 0.

Note that the same can be done with $|V_3| > 2$ and then the above x will be the unique optimal solution to the nice relaxation.

Thus straightforward application of the pipage rounding method does not provide a constant-factor approximation.

Moreover, Example 1 shows that the greedy algorithm (at each step add a vertex which increases most or decreases least the weight of the cut) does not yield any constant-factor approximation, too. For this instance the greedy algorithm may first choose the vertices of V_2 and then no more arcs can be added and a solution with only 4 arcs will be the outcome (while the optimal one is to choose p vertices from V_1, which gives a cut of size $2p$).

4. The structure of basic solutions. The following fact, which may be of interest beyond the purposes of this paper, is crucial in constructing a 0.5-approximation for MAX DICUT WITH GSP in the general case.

THEOREM 4.1. *Let (x, z) be a basic feasible solution to the linear relaxation* (3.1)–(3.5). *Then*

$$x_i \in \{0, \delta, 1/2, 1-\delta, 1\} \text{ for each } i, \tag{4.1}$$

for some $0 < \delta < 1/2$.

Proof. Let (x, z) be a basic feasible solution. By definition (x, z) is the unique solution to the system of linear equations formed by those constraints in (3.2)–(3.5) which hold with equality. First, observe that for any $ij \in A$, either both $z_{ij} \le x_i$ and $z_{ij} \le 1 - x_j$ hold with equalities or exactly one holds with equality and the other with strict inequality. In the former case we exclude z_{ij} by replacing these equations with the single equation $x_i + x_j = 1$. In the latter case we delete that equality from the linear system. In either case the variable z_{ij} retires from the system while the number of equations reduces by one and, moreover, the value of z_{ij} can be uniquely determined by the values of x_i and x_j. After performing this operation for each

$ij \in A$, we arrive at a system that can be written in the following form:

$$y_i + y_j = 1 \text{ for } ij \in A' \subseteq A, \tag{4.2}$$
$$\sum_i y_i = p, \tag{4.3}$$
$$y_i = 0 \text{ for every } i \text{ such that } x_i = 0, \tag{4.4}$$
$$y_i = 1 \text{ for every } i \text{ such that } x_i = 1. \tag{4.5}$$

Since the removed z can be uniquely restored from the vector x, x must be the unique solution to this system. Remove all components of x equal to either 0 or 1 or 1/2 and denote the set of the remaining indices by I^*. Denote by x' the subvector of x consisting of the components with indices in I^*. Now consider the system that is obtained from (4.2)–(4.5) by substituting $y_i = x_i$ for each i such that $x_i \in \{0, 1/2, 1\}$. Since x is the unique solution to (4.2)–(4.5), x' is the unique solution to the resulting system, which can be written in the following form:

$$y'_i + y'_j = 1 \text{ for } ij \in A'' \subseteq A', \tag{4.6}$$
$$\sum_i y'_i = p', \tag{4.7}$$

where $p' \leq p$. It follows that one can choose $|I^*|$ independent equations from (4.6)–(4.7). We claim that any subsystem of this sort must contain the equation (4.7). Assume to the contrary that $|I^*|$ equations from the set (4.6) form an independent system. Consider the (undirected) subgraph H of G (we ignore orientations) corresponding to these equations. Note that $|E(H)| = |I^*|$. Since $x'_i \neq 1/2$ for all $i \in I^*$, H does not contain odd cycles. Moreover, H cannot have even cycles as the subsystem corresponding to such a cycle is clearly dependent. Thus H is acyclic. But then $|E(H)| \leq |I^*| - 1$, a contradiction.

Now fix $|I^*|$ independent equations from (4.6)–(4.7). Then, by the above claim, we obtain the following system:

$$y'_i + y'_j = 1 \text{ for } ij \in A^*, \tag{4.8}$$
$$\sum_{i \in I^*} y'_i = p' \tag{4.9}$$

where $A^* \subseteq A'$. Since all the equations in (4.8)–(4.9) are independent, $|I^*| = |A^*| + 1$. Above we have proved that the subgraph spanned by A^* is acyclic, which together with $|I^*| = |A^*| + 1$ implies that it is a tree. Recall also that x' is the unique solution to (4.8)–(4.9) and $x'_i = x_i \neq 0, 1, 1/2$ for all $i \in I^*$. All these facts together with the structure of equalities (4.8) imply that the components of x with indices in I^* split into two sets—those equal to some $0 < \delta < 1/2$ and those equal to $1 - \delta$. ∎

5. Algorithm DIRCUT. Section §3 demonstrates that MAX DICUT WITH GSP in the general setting does not admit a direct application of the pipage rounding method. In this section we show that by using Theorem 4.1 and some tricks one is able to design not only a constant factor but even a 0.5-approximation for solving MAX DICUT WITH GSP. Moreover, the performance bound of 0.5 cannot be improved using different methods of rounding as the integrality gap of (3.1)–(3.5) can be arbitrarily close to 1/2 (this can be shown exactly in the same way as it was done for MAX CUT WITH GSP in [AS99]).

For $ij \in A$, call the number $w_{ij}\min\{x_i,(1-x_j)\}$ the *contributed weight* of the arc ij. Observe that for any $a,b \in [0,1]$, $ab = \max\{a,b\}\min\{a,b\}$. It follows that

$$\begin{aligned} F(x) &= \sum_{ij\in A} w_{ij}x_i(1-x_j) = \sum_{ij\in A} w_{ij}\max\{x_i,1-x_j\}\min\{x_i,1-x_j\} \\ &= \sum_{ij\in A} \max\{x_i,1-x_j\}(\text{“ the contributed weight of } ij\text{ ”}). \end{aligned} \tag{5.1}$$

Algorithm DIRCUT consists of two phases: the first phase is to find an optimal (fractional) basic solution to the linear relaxation (3.1)–(3.5); the second (rounding) phase is to transform this solution to a feasible (integral) solution. The rounding phase runs two different rounding algorithms based on the pipage rounding method and takes the best solution for the output. Let (x,z) denote a basic optimal solution to (3.1)–(3.5) obtained at the first phase. Recall that, by Theorem 1, the vector x satisfies (4.1). Set $V_1 = \{i : x_i = \delta\}$, $V_2 = \{i : x_i = 1-\delta\}$, $V_3 = \{i : x_i = 1/2\}$, $V_4 = \{i : x_i = 0 \text{ or } 1\}$. Denote by l_{kl} $(k,l = 1,2,3,4)$ the sum of contributed weights over all arcs going from V_k to V_l. Set $l_0 = l_{11}+l_{21}+l_{22}+l_{23}+l_{31}$, $l_1 = l_{33}+l_{13}+l_{32}$, $l_2 = \sum_{i=1}^4 (l_{i4}+l_{4i})$ (Fig. 5.1 might be helpful to the reader).

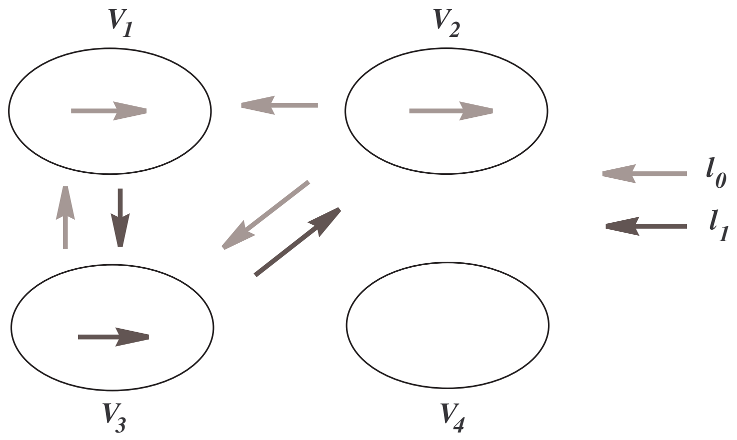

FIG. 5.1. *l_0 and l_1 are the sums of contributed weights over the displayed collections of arcs.*

The second phase of the algorithm successively calls two rounding algorithms—ROUND1 and ROUND2—and takes a solution with maximum weight for the output.

ROUND1 is the pipage rounding applied to the optimal basic solution x. Let $\tilde{x}$ denote the output of ROUND1. Then by the description of pipage rounding, $F(\tilde{x}) \geq F(x)$. Since the number $\max\{x_i,1-x_j\}$ remains unchanged for any $ij \in A$ such that $i \in V_k$ and $j \in V_l$, and can be easily computed, (5.1) implies that

$$F(\tilde{x}) \geq F(x) \geq \delta l_{12} + (1-\delta)l_0 + l_1/2 + l_2. \tag{5.2}$$

Algorithm ROUND2 is the pipage rounding applied to a different fractional solution x' which is obtained by an alteration of x.

ALGORITHM ROUND2. Define a new vector x' by the following formulas:

$$x'_i = \begin{cases} \min\{1, \delta+(1-\delta)|V_2|/|V_1|\} & \text{if } i \in V_1, \\ \max\{0, (1-\delta)-(1-\delta)|V_1|/|V_2|\} & \text{if } i \in V_2, \\ x_i & \text{if } i \in V \setminus (V_1 \cup V_2). \end{cases} \tag{5.3}$$

Apply the pipage rounding to x'.

Analysis. The vector x' is obtained from x by redistributing uniformly the values from the components in V_2 to the those in V_1 and keeping the same the remaining ones. It follows from the description of ROUND2 that x' is feasible. Applying the pipage rounding to x' results in an integral feasible vector of weight at least $F(x')$. We claim that $F(x') \geq l_{12}+l_1/2$. Consider first the case when $|V_1| \geq |V_2|$. Then by (5.3), $x'_i = 0$ for all $i \in V_2$ and $x'_i \geq \delta$ for all $i \in V_1$. So, by (5.1) and definitions of V_k it follows that $F(x') \geq l_{32} + l_{12} + 1/2(l_{13} + l_{33})$. Now assume that $|V_1| \leq |V_2|$. Then by (5.3), $x'_i = 1$ and $x'_i \leq 1-\delta$ for all $i \in V_1$. Hence, again by (5.1), $F(x') \geq l_{13} + l_{12} + 1/2(l_{32} + l_{33})$. So, in either case $F(x') \geq l_{12} + l_1/2$. Thus ROUND2 outputs a solution of weight at least $l_{12} + l_1/2$. Together with (5.2) this implies that the output of DIRCUT has weight at least

$$\max\{l_{12} + l_1/2, \delta l_{12} + (1-\delta)l_0 + l_1/2 + l_2\},$$

which is bounded from below by

$$q = \max\{l_{12}, \delta l_{12} + (1-\delta)l^*\} + l_1/2$$

where $l^* = l_0 + l_2$. Recall now that $0 < \delta < 1/2$. Hence, if $l_{12} \geq l^*$, then $q = l_{12} + l_1/2 \geq (l_{12} + l^* + l_1)/2$ and if $l_{12} < l^*$, then $q = \delta l_{12} + (1-\delta)l^* + l_1/2 > (l_{12} + l^*)/2 + l_1/2$. Thus, in either case algorithm DIRCUT outputs a solution of weight at least $(l_{12} + l_0 + l_1 + l_2)/2$, which is at least half of the optimum.

6. Directly tractable special cases. In the final section we consider two special cases of MAX DICUT WITH GSP which admit direct application of the pipage rounding method.

6.1. The circulation case. We first consider the case when the weight function w is a circulation in the given graph. This means that the function w obeys the condition

$$\sum_{j:ij\in A} w_{ij} = \sum_{k:ki\in A} w_{ki}$$

for each vertex $i \in V$. We will show that the circulation case of MAX DICUT WITH GSP admits a 0.5-approximation algorithm which is a straightforward implementation of the scheme described in §2. In the next subsection we will show that it also finds a cut of weight within a factor of 0.5 of the optimum in the case when the cuts are constrained to have equal parts (the DIGRAPH BISECTION problem).

Note first that for any a and b between 0 and 1,

$$2a(1-b) = a(1-b) + b(1-a) + a - b,$$
$$2\min\{a, 1-b\} = \min\{a+b, 2-a-b\} + a - b.$$

Using these identities we can rearrange the functions F and L in the following way:

$$F(x) = 1/2 \sum_{ij\in A} w_{ij}\big[x_i(1-x_j) + x_j(1-x_i) + x_i - x_j\big],$$

$$L(x) = 1/2 \sum_{ij\in A} w_{ij}\big[\min\{x_i + x_j, 2 - x_i - x_j\} + x_i - x_j\big].$$

Since

$$\sum_{ij\in A} w_{ij}(x_i - x_j) = \sum_{i\in V} q_i x_i,$$

where q_i is the difference between the sum of weights of arcs leaving the node i and the sum of weights of arcs entering the node i, the both functions can be expressed as the sums of two summands:

$$F(x) = 1/2 \sum_{ij\in A} w_{ij}\big[x_i(1-x_j) + x_j(1-x_i)\big] + 1/2 \sum_{i\in V} q_i x_i, \tag{6.1}$$

$$L(x) = 1/2 \sum_{ij\in A} w_{ij}\big[\min\{x_i + x_j, 2 - x_i - x_j\}\big] + 1/2 \sum_{i\in V} q_i x_i. \tag{6.2}$$

Ageev and Sviridenko [AS99] proved that for $0 \le x_i, x_j \le 1$,

$$\frac{x_i(1-x_j) + x_j(1-x_i)}{\min\{x_i + x_j, 2 - x_i - x_j\}} \ge 1/2. \tag{6.3}$$

It follows that $F(x)/L(x) \ge 1/2$ if $\sum_{i\in V} q_i x_i \ge 0$. In the case when the weights w form a circulation in G each q_i is equal to zero. This proves the bound of 1/2 for this case.

6.2. Digraph Bisection. The DIGRAPH BISECTION problem is the special case of MAX DICUT WITH GSP where $n = 2p$. Let (x, z) be an optimal solution to the linear relaxation (3.1)–(3.5). We claim that in this case $\sum_{i\in V} q_i x_i \ge 0$, which means by (6.1), (6.2), and (6.3) that the algorithm presented for the circulation case has an approximation ratio of 1/2 for DIGRAPH BISECTION too. Indeed, assume to the contrary that $\sum_{i\in V} q_i x_i < 0$. Since $n = 2p$, the vector y defined by

$$y_i = 1 - x_i, \quad \forall\ i \in V$$

is also feasible for the nice relaxation (3.7), (3.4), (3.5). Moreover,

$$\sum_{i\in V} q_i x_i = -\sum_{i\in V} q_i y_i$$

and

$$\min\{x_i + x_j, 2 - x_i - x_j\} = \min\{y_i + y_j, 2 - y_i - y_j\}$$

for every $ij \in A$. This means that $L(y) > L(x)$, which contradicts the optimality of x.

REFERENCES

[AS99] A. A. AGEEV AND M.I. SVIRIDENKO, *Approximation algorithms for Maximum Coverage and Max Cut with given sizes of parts*, in IPCO'99, G. Cornuéjols, R. E. Burcard, and G. J. Woeginger, eds., Lecture Notes in Comput. Sci. 1610, Springer-Verlag, New-York, 1999, pp. 17–30.

[AS00] A. A. AGEEV AND M. I. SVIRIDENKO, *An approximation algorithm for Hypergraph Max k-Cut with given sizes of parts*, in ESA'00, M. Paterson, ed., Lecture Notes in Comput. Sci. 1879, Springer-Verlag, New-York, 2000, pp. 32–41.

[FG95] U. Feige and M.X. Goemans, *Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT*, in Proceedings of the Third Israel Symposium on Theory of Computing and Systems, Tel Aviv, Israel, 1995, pp. 182–189.

[FL99] U. Feige and M. Langberg, *Approximation algorithms for maximization problems arising in graph partitioning*, manuscript, 1999.

[GW95] M. X. Goemans and D. P. Williamson, *Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming*, J. ACM 42 (1995), pp. 1115–1145.

[HR00] R. Hassin and S. Rubinstein, *Approximation algorithms for maximum linear arrangement*, in SWAT'00, M. M. Halldórsson, ed., Springer-Verlag, New-York, Lecture Notes in Comput. Sci. 1851, Springer-Verlag, New-York, 2000, pp. 231–236.

[Ja98] K. Jain, *A factor 2 approximation algorithm for the generalized Steiner network problem*, in Proceedings of FOCS'98, IEEE Computer Society, 1998, pp. 448–457.

[Ka72] R. M. Karp, *Reducibility among Combinatorial Problems*, in Complexity of Computer Computations, R. E. Miller and J. W. Thatcher, eds., Plenum Press, New-York, 1972, pp. 85–103.

[MT99] V. Melkonian and É. Tardos, *Approximation algorithms for a directed network design problem*, in IPCO'99, G. Cornuéjols, R. E. Burcard, and G. J. Woeginger, eds., Lecture Notes in Comput. Sci. 1610, Springer-Verlag, New-York, 1999, pp. 345–360.