MAX DICUT WITH GSP AND UNBALANCED GRAPHS

Ali Shahwali
shahwali@kth.se
Author

Per Austrin
austrin@kth.se
Supervisor

KTH Royal Institute of Technology

Abstract

The MAX DICUT WITH GSP (given sizes of parts) problem is the following, given a directed graph $G = (V, E, w)$ and a part size p, find a cut $X \subseteq V$ that maximizes the sum of weights of the outgoing edges from X such that $|X| = p$. The problem is **NP**-hard and thus we naturally resort to approximation algorithms to obtain solutions in polynomial time. The current state of the art is a 0.5 approximation algorithm devised by Ageev, Hassin and Sviridenko [AHS01]. The algorithm utilizes the method of Pipage Rounding devised by Ageev and Sviridenko [AS04] and a key insight to the problem structure. In this report we will briefly analyze special types of graphs which we call *unbalanced graphs*, as well as fully cover the current state of the art algorithm for MAX DICUT WITH GSP.

1 Introduction

The maximum directed cut (MAX DICUT) problem is the following, let $G = (V, E, w)$ be a weighted directed graph where V is the set of vertices, E is the set of edges and w is an edge weight function $E(G) \to \mathbb{R}_+$. A directed cut is a set of edges leaving some subset $X \subseteq V$, thus, MAX DICUT is the problem of finding such a subset X that maximizes the weight of the outgoing edges. MAX DICUT with given sizes of parts (MAX DICUT WITH GSP) is MAX DICUT with an additional cardinality constraint on X such that $|X| = p$. MAX DICUT is an **NP**-hard problem as MAX CUT, one of the 21 problems on Karp's list of **NP**-complete problems [Kar72], can be reduced to it. MAX DICUT can be reduced to MAX DICUT WITH GSP by a Turing reduction where you compute MAX DICUT WITH GSP for all part sizes $0 < p < n$ and return the largest of these cuts, and as such MAX DICUT WITH GSP is also **NP**-hard. Therefore, the best we can do in polynomial time is approximate solutions to MAX DICUT WITH GSP. It was only until Goemans and Williamson seminal paper [GW95] where an approximation algorithm using semidefinite programming was introduced that an approximation factor greater than 1/2 was proven to be possible. This novel technique provided an approximation factor of 0.878 for MAX CUT and 0.796 for MAX DICUT. Feige and Goemans later improved the approximation factor of MAX DICUT to 0.859 [FG95], the current best approximation algorithm is by Brakensiek, Huang, Potechin and Zwick [Bra+23]. As for MAX CUT WITH GSP, Feige and Langberg [FL01] expanded on the work of Ageev and Sviridenko [AS99] in which their method of pipage rounding was applied to the problem,

this extension of their work resulted in a $0.5 + \epsilon$ approximation, for some $\epsilon > 0$. It turns out that a direct application of pipage rounding is not possible for MAX DICUT WITH GSP as Ageev, Hassin and Sviridenko showed [AHS01], however with some key insights of the problem structure they could nevertheless construct a 0.5 approximation algorithm for the problem. In this report we will cover the algorithm for MAX DICUT WITH GSP by Ageev, Hassin and Sviridenko as well as analyse a special type of problem instance we call *unbalanced graphs*.

2 Current best approximation algorithm for MAX DICUT WITH GSP

The current state of the art approximation algorithm for MAX DICUT WITH GSP is due to Ageev, Hassin and Sviridenko [AHS01]. The algorithm consists of an application of the pipage rounding method devised by Ageev and Sviridenko [AS04] in combination with some key observations of the problem structure.

2.1 Pipage rounding

Consider a function $F(x)$ defined on the rational points of a n-dimensional hypercube. Consider another function $L(x)$ defined on the same points which is a relaxtion of $F(x)$ and solvable in polynomial time. Pipage rounding is a method of obtaining an integer solution for $F(x)$ by rounding a fractional solution produced by $L(x)$ to a binary integer solution with a constant approximation factor in polynomial time. The following two conditions need to hold for a direct application of pipage rounding:

F/L lower bound condition:

There exists a constant $0 < C < 1$ such that $F(x) \geq CL(x)$ for each rational point x in the n-dimensional hypercube $[0, 1]^n$.

ε-convexity condition:

The function

$$\varphi(\varepsilon, x, i, j) = F(x_1, \ldots, x_i + \varepsilon, \ldots, x_j - \varepsilon, \ldots, x_n) \tag{1}$$

is convex with respect to

$$\varepsilon \in [-\min(x_i, 1 - x_j), \min(1 - x_i, x_j)]$$

for pair (i, j) and each $x \in [0, 1]^n$.

2.1.1 Description of pipage rounding

We define a nonlinear binary program as an optimization problem where the feasible region is determined by nonlinear constraints or the objective function is nonlinear. Assume that a problem can be formulated as the following nonlinear binary program

$$\max\ F(x) \tag{2a}$$

$$\text{s.t}\ \sum_{i=1}^{n} x_i = p \tag{2b}$$

$$0 \leq x_i \leq 1,\ i = 1, \ldots, n \tag{2c}$$

$$x_i \in \{0, 1\},\ i = 1, \ldots, n \tag{2d}$$

Consider a fractional solution x to a linear program such as the one defined by (2b) - (2c), the pipage rounding procedure takes such an input and outputs an integer solution $\tilde{x}$ that satisfies (2a) - (2d) and has the following property

$$F(\tilde{x}) \geq F(x)$$

The procedure of rounding from x to $\tilde{x}$ can be divided up in to the following steps which are repeated at most $n - 1$ times.

1. If x only has integer components, output x and terminate, otherwise continue.

2. Since x is not an integer solution, then there exists at least two different components which are fractional due to (2b). From (1) we get $\varphi(\varepsilon, x, i, j) \geq F(x)$ for $\varepsilon = \min(1 - x_i, x_j)$ or $\varepsilon = -\min(x_i, 1 - x_j)$, this is due to the convexity of φ which means it is possible that $\varphi(\varepsilon, x, i, j) > F(x)$ for some ε, thus $\varphi(\varepsilon, x, i, j) \geq F(x)$. We now have a new solution

   $$x' = (x_1, \ldots, x_i + \varepsilon, \ldots, x_j - \varepsilon, \ldots, x_n)$$

   with fewer fractional components than the previous solution due to the subtraction of ε to x_i and x_j. This new solution has the property $F(x') \geq F(x)$.

2.2 Formulating the problem

MAX DICUT WITH GSP can be formulated as the following non-linear binary program

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

We can formulate it as the following integer program

$$\max \sum_{ij \in E} w_{ij} z_{ij} \tag{3a}$$

$$\text{s.t. } z_{ij} \leq x_i \qquad \forall_{ij} \in E, \tag{3b}$$

$$z_{ij} \leq 1 - x_j \qquad \forall_{ij} \in E, \tag{3c}$$

$$\sum_{i \in V} x_i = p, \tag{3d}$$

$$0 \leq x_i \leq 1 \qquad \forall_i \in V, \tag{3e}$$

$$x_i, z_{kj} \in \{0, 1\} \qquad \forall_i \in V, \forall_{kj} \in E \tag{3f}$$

By substituting $z_{ij} = \min(x_i, 1 - x_j)$ and excluding (3f), we get the following linear relaxtion

$$\max L(x) = \sum_{ij \in E} w_{ij} \min(x_i, 1 - x_j) \tag{4a}$$

$$\text{s.t. } \sum_{i \in V} x_i = p, \tag{4b}$$

$$0 \leq x_i \leq 1 \qquad \forall_i \in V \tag{4c}$$

2.3 Properties of the IP and the LP

Remark 1. *The integrality gap of* (4a) - (4c) *is 0.5.*

Proof. Consider a graph $G = (V, E, w)$ where the given part size $p \leq \frac{n}{2}$. The weight w_{ij} from i to j is 1 if they lie in some fixed $X \subseteq V$ where $|X| = 2p \leq n$, otherwise $w_{ij} = 0$, in other words X is a clique of G, see figure 1. The optimal directed cut to such an instance is p^2. The reason is simple, since X is a complete digraph then simply set the solution to p arbitrary vertices from X, they will all have an outgoing edge of weight 1 to each remaining p vertices in X and thus the weight of the cut is $p \cdot p = p^2$. As for the optimal value of the linear relaxation, the optimal value is $p(2p - 1)$. The reason is the following, maximizing (4a) for G can be done by setting all p nodes $x_i \in X$ to 0.5, X is a complete digraph and thus has $2p(2p - 1)$ edges. And since each edge has a weight of 1 we are left with

$$\underbrace{0.5 + 0.5 + \cdots + 0.5}_{2p(2p-1)} = p(2p - 1)$$

Now we can analyze the ratio between the optimal cut and the optimal value for the linear relaxtion.

$$\frac{p^2}{p(2p - 1)} = \frac{p}{2p - 1}$$

We can see that as p increases we can get arbitrarily close to 0.5. Thus the integrality gap of (4a) - (4c) is 0.5. □

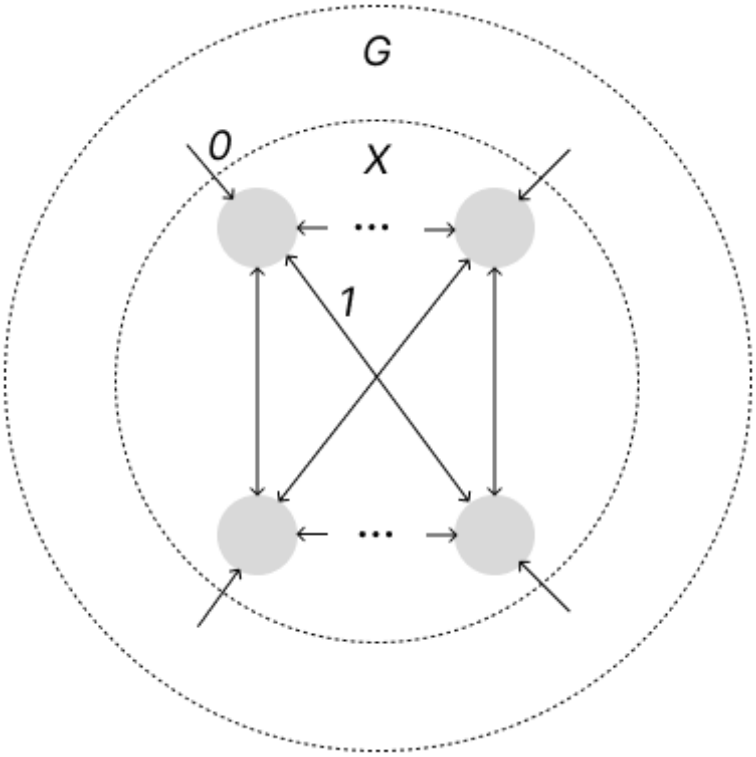

Figure 1: Example of a graph proving that the integrality gap of (4a) - (4c) is 0.5.

Because the integrality gap is 0.5, any rounding scheme applied to (4a) - (4c) can not result in an approximation factor better than 0.5.

Remark 2. *The ε-convexity condition holds for the function $F(x)$.*

Proof. For all (i, j) we want to show that $\phi(\varepsilon, x, i, j)$ is convex with respect to $\varepsilon \in [-\min(x_i, 1 - x_j), \min(1 - x_i, x_j)]$. For each edge $(i, j) \in E$ we get the following

$$\begin{aligned}
\phi(\varepsilon, x, i, j) &= F(x_1, \ldots, x_i + \varepsilon, \ldots, x_j - \varepsilon, \ldots, x_n) \\
&= \sum_{ij \in E} w_{ij} x_i (1 - x_j) \\
&= w_{ij}(x_i + \varepsilon)(1 - (x_j - \varepsilon)) + w_{ji}(x_j - \varepsilon)(1 - (x_i + \varepsilon)) + \underbrace{l(\varepsilon, x, i, j)}_{\text{Linear in } \varepsilon} \\
&= 2w_{ij}\varepsilon^2 + l(\varepsilon, x, i, j)
\end{aligned}$$

$l(\varepsilon, x, i, j)$ is simply the terms linear in ε that remain after expansion. We can now see that for all pairs of edges (i, j), $\phi(\varepsilon, x, i, j)$ is a quadratic polynomial in ε with positive coefficients, thus $\phi(\varepsilon, x, i, j)$ is convex with respect to ε, and the ε-convexity condition holds for $F(x)$. □

While the ε-convexity condition holds for $F(x)$, the F/L lower bound condition does not. Thus, a straight forward application of pipage rounding will not suffice to get a constant factor approximation.

Remark 3. *The F/L lower bound condition does not hold for the functions F and L.*

Proof. This can be proven with an instance where $F(x)/L(x)$ tends to 0. Consider the graph $G = (V, E)$ where $V = V_1 \cup V_2 \cup V_3$ and $|V_1| = k$, $|V_2| = |V_3| = 2$, assume $p \leq k$. $E = E_1 \cup E_2$, every node in V_1 has an edge to both nodes in V_2, thus $|E_1| = 2k$. Both nodes in V_2 have an edge to both nodes in V_3, thus $|E_2| = 4$, assume all edge weights $w_e = 1$, see figure 2. It is obvious that an optimal solution to such an instance is $2p$,

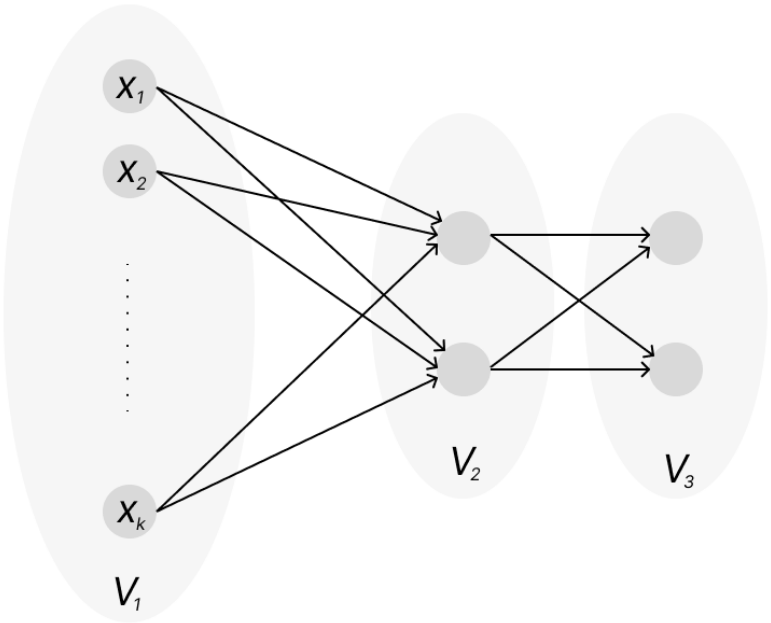

Figure 2: An example of an instance where the F/L lower bound condition does not hold.

which you get by simply choosing p arbitrary nodes from V_1 as your cut. We can see that a solution x to the linear relaxation (4a) - (4c) is be found by setting all values in V_1 to $\frac{p-2}{k-2}$, and the values in V_2 to $1 - \frac{p-2}{k-2}$, and the values in V_3 to 0. $L(x)$ is then

$$L(x) = |E_1| \min(\frac{p-2}{k-2}, \frac{p-2}{k-2}) + |E_2| \min(1 - \frac{p-2}{k-2}, 1)$$

$$= 2k\frac{p-2}{k-2} + 4(1 - \frac{p-2}{k-2}) = 2p$$

and $F(x)$ is

$$F(x) = |E_1|\frac{p-2}{k-2}\frac{p-2}{k-2} + |E_2|(1 - \frac{p-2}{k-2}) = \frac{2k(p-2)^2}{(k-2)^2} - \frac{4(p-2)}{k-2} + 4$$

if we set $k = p^2$ and let p go to infinity we get the following

$$\lim_{p\to\infty} F(x)/L(x) = \lim_{p\to\infty} \frac{\frac{2p^2(p-2)^2}{(p^2-2)^2} - \frac{4(p-2)}{p^2-2} + 4}{2p} = 0$$

Thus, the F/L lower bound condition doesn't hold for F and L. □

2.4 0.5 Approximation algorithm

Before we describe the algorithm, we provide some definitions. We say a solution x to a linear program is a *feasible solution* if it satisfies all constraints of the linear program. A solution x to a linear program is a *basic feasible solution* if x is a feasible solution and there exists no two other feasible solutions (y, z) such that $x = (y + z)/2$. Another definition of a basic feasible solution, which will be used going forward, is the following. We say a feasible solution is a basic feasible solution if it satisfies exactly n, where n is the number of variables, linearly independent rows with equality. Geometrically we can interpret the basic feasible solutions to a linear program as the vertices, or corners, of the polyhedron spanned by the feasible region.

Theorem 1 (Ageev, Hassin, Sviridenko [AHS01])**.** *Let (x, z) be a basic feasible solution to* (3a) - (3e)*. Then*

$$x_i \in \{0, \delta, 1/2, 1-\delta, 1\} \text{ for each } i \tag{5}$$

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

Proof. Consider a basic feasible solution (x, z), (x, z) is the unique solution to the linear system of equations formed by the constraints in (3a) - (3e). If we study the two inequalities

$$z_{ij} \leq x_i$$

$$z_{ij} \leq 1 - x_j$$

we can notice the following, either both equations hold with equality, or one holds with equality and the other holds with strict inequality. The former case can be expressed as $x_i = 1 - x_j$, thus we can replace the equalities with the following equation in the linear system.

$$x_i + x_j = 1$$

The latter occurs when $x_i \neq 1 - x_j$, in this case we get one of the following equations

$$z_{ij} = x_i$$

$$z_{ij} = 1 - x_j$$

which can obviously be removed from the linear system as it is redundant. If we do this for all edges $(i, j) \in E$, we end up with a new system of linear equations for a new subset

of edges $E' \subseteq E$. We know that x is also a unique solution to the new system as every z can be restored from x. The new system with all z_i removed is the following

$$x_i + x_j = 1, \qquad \forall ij \in E' \tag{6a}$$

$$\sum_{x_i \in x} x_i = p, \tag{6b}$$

$$x_i = 0, \qquad \forall_{x_i \in x} \mid x_i = 0 \tag{6c}$$

$$x_i = 1 \qquad \forall_{x_i \in x} \mid x_i = 1 \tag{6d}$$

We are interested in all $x_i \in x$ such that $x_i \notin \{1, 0, 1/2\}$, if we remove all such x_i, we are left with a new subvector x', a corresponding new subset of vertices $V' \subseteq V$, a new subset of edges $E'' \subseteq E'$ and $p' \leq p$ since $|V'| \leq |V|$. The new system is the following

$$x'_i + x'_j = 1, \qquad \forall ij \in E'' \tag{7a}$$

$$\sum_{x'_i \in V'} x'_i = p' \tag{7b}$$

We can show through contradiction that any subsystem of $|V'|$ independent equations from (7a) - (7b) must include (7b). Assume that this isn't the case. We can view the equations (7a) as a graph. Let $H = (V_H, E'')$ be an undirected subgraph of G formed by the edges E'', it has the following property

$$|E''| = |V'|$$

We know that $x_i \neq \frac{1}{2}$ for all $x_i \in V'$, thus H can not contain cycles of odd length. The reason being that any odd cycle results in the following equations

$$x_{2k+1} = 1 - x_{2k} = \cdots = x_1$$

$$x_{2k+1} + x_1 = 1$$

which has the only solution $x_{2k+1} = x_1 = \frac{1}{2}$. In addition, H can not contain even cycles as we get the following equations.

$$x_{2k} = \cdots = 1 - x_1$$

$$x_{2k-1} + x_1 = 1$$

which is only solvable for $x_1 \in \{0, 1\}$ and $x_{2k} \in \{0, 1\}$. This is however a contradiction, if H is acyclic then

$$|E''| \leq |V'| - 1$$

Thus our initial assumption must be false, and any subsystem of (7a) - (7b) must include (7b). By fixing the $|V'|$ equations that are independent from (7a) - (7b) we obtain a new system with E^* edges

$$x'_i + x'_j = 1, \qquad \forall ij \in E^* \tag{8a}$$

$$\sum_{x'_i \in V'} x'_i = p' \tag{8b}$$

We know that $|V'| = E^* + 1$ and that the subgraph spanned by E^* is acyclic. Together this implies that the subgraph is a tree. Since x' is still the unique solution to (8a) - (8b) and we have removed all components $x'_i \in \{0, 1, \frac{1}{2}\}$, we can infer from the structure of the linear equation (8a) and the aforementioned properties, that the remaining components of x' equal δ or $1 - \delta$ for some $0 < \delta < \frac{1}{2}$. □

Since a direct application of pipage rounding does not yield a constant factor approximation, an algorithm with a constant factor approximation will require some additional steps. First, two rounding functions will be defined, we will call them $ROUND1$ and $ROUND2$. Let $ROUND1$ be a direct application of pipage rounding. Let $ROUND2$ be the following procedure, consider a optimal basic solution x to (4a) - (4c), construct a new solution x' where for all $x'_i \in x'$

$$x'_i = \begin{cases} \min(1, \delta + (1-\delta)|V_{1-\delta}|/|V_\delta|) & \text{if } i \in V_\delta, \\ \max(0, (1-\delta) - (1-\delta)|V_\delta|/|V_{1-\delta}|) & \text{if } i \in V_{1-\delta}, \\ x_i & \text{if } i \in V_s \cup V_{1/2} \end{cases} \tag{9}$$

where $V_\delta = \{i : x_i = \delta\}$, $V_{1-\delta} = \{i : x_i = 1 - \delta\}$, $V_{1/2} = \{i : x_i = 1/2\}$ and $V_s = \{i : x_i = 1 \text{ or } 0\}$ which we know the union of contains all $x_i \in x$ by theorem 1. The last step is to perform pipage rounding on x'. The algorithm is given by 1.

Algorithm 1 0.5 approximation of MAX DICUT WITH GSP

Require: $G = (V, E)$
1: $x \leftarrow OptimalBasicSolution(G)$
2: $x_{ROUND1} \leftarrow PipageRounding(x)$
3: $x_{ROUND2} \leftarrow PipageRounding(ROUND2(x))$
4: **if** $CutSize(G, x_{ROUND1}) > CutSize(G, x_{ROUND2})$ **then**
5: **return** x_{ROUND1}
6: **else**
7: **return** x_{ROUND2}
8: **end if**

2.4.1 Analysis

The idea behind $ROUND2$ is to redistribute values from V_δ and $V_{1-\delta}$ uniformly, and as a result guaranteeing that the resulting cut is at least half the size of the optimal cut. We begin by considering an edge $(i, j) \in E$ contributed weight to the cut, we define a contributed weight as $z_{ij} = w_{ij} \min(x_i, 1 - x_j)$. We can now see that

$$F(x) = \sum_{ij \in E} w_{ij} x_i (1 - x_j) = \sum_{ij \in E} w_{ij} \underbrace{\min(x_i, 1 - x_j) \max(x_i, 1 - x_j)}_{= x_i(1-x_j)}$$

$$= \sum_{ij \in E} z_{ij} \max(x_i, 1 - x_j) \tag{10}$$

Let W_{ij} denote the sum of contributed weights over all edges from V_i to V_j, that is,

$$W_{ij} = \sum_{e \in E_{ij}} w_e \min(x_i, (1 - x_j))$$

then we define

$$W_0 = W_{\delta\delta} + W_{(1-\delta)\delta} + W_{(1-\delta)(1-\delta)} + W_{(1-\delta)\frac{1}{2}} + W_{\frac{1}{2}\delta}$$

$$W_1 = W_{\frac{1}{2}\frac{1}{2}} + W_{\delta\frac{1}{2}} + W_{\frac{1}{2}(1-\delta)}$$

$$W_2 = W_{\delta s} + W_{s\delta} + W_{(1-\delta)s} + W_{s(1-\delta)} + W_{\frac{1}{2}s} + W_{s\frac{1}{2}} + W_{ss} + W_{ss}$$

At this point it should be noted that it follows from the definition of the LP (4a) - (4c) and sum of contributed weights that

$$W_{\delta(1-\delta)} + W_0 + W_1 + W_2$$

is exactly equal to the size of the cut produced by the LP, see figure 3.

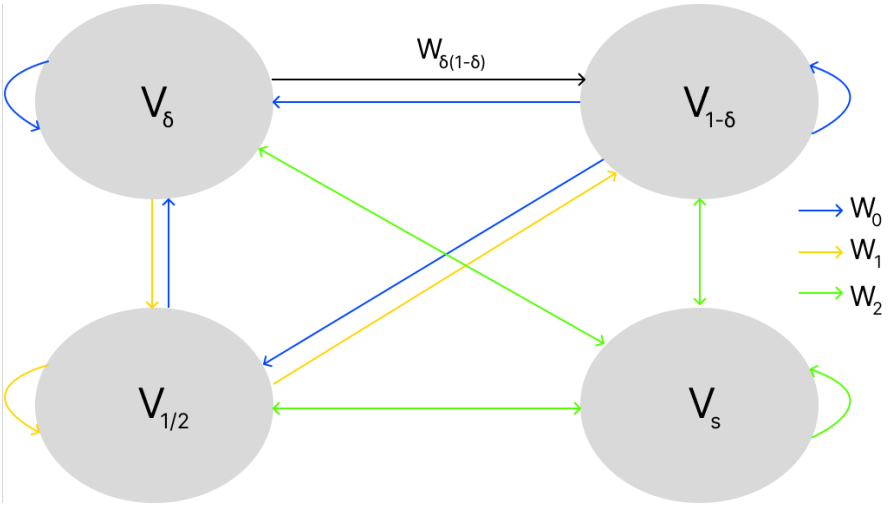

Figure 3: Visualization of the sum of contributed weights.

Lemma 1. *$F(\tilde{x}) \geq F(x) \geq \delta W_{\delta(1-\delta)} + (1-\delta)W_0 + \frac{1}{2}W_1 + W_2$, where $\tilde{x}$ is the output of $ROUND1$ and x is an optimal basic solution to* (4a) - (4c).

Proof. It follows from the description of pipage rounding that $F(\tilde{x}) \geq F(x)$. By equation (10) it follows that W_0 is the sum of contributed weights of $1 - \delta$, W_1 is the sum of contributed weights of $\frac{1}{2}$ and W_2 is the sum of contributed weights of 1. Lastly are the sum of contributed weights of δ which is solely $W_{\delta(1-\delta)}$. Thus

$$F(x) \geq \delta W_{\delta(1-\delta)} + (1-\delta)W_0 + \frac{1}{2}W_1 + W_2$$

□

Lemma 2. *$F(x') \geq W_{\delta(1-\delta)} + \frac{1}{2}W_1$, where x' is given by (9).*

Proof. We can analyze (9) in the case that $|V_\delta| \geq |V_{1-\delta}|$. We notice that any value $i \in V_{1-\delta}$ will result in $x'_i = 0$, and any $i \in V_\delta$ will result in $x'_i \geq \delta$. It then follows due to (10) that

$$F(x') \geq W_{\frac{1}{2}(1-\delta)} + W_{\delta(1-\delta)} + \frac{1}{2}W_{\delta\frac{1}{2}} + \frac{1}{2}W_{\frac{1}{2}\frac{1}{2}}$$

In the case that $|V_\delta| \leq |V_{1-\delta}|$, any $i \in V_\delta$ will result in $x'_i = 1$ and any $i \in V_{1-\delta}$ will result in $x'_i \leq 1 - \delta$. It follows again due to (10) that

$$F(x') \geq \frac{1}{2}W_{\frac{1}{2}(1-\delta)} + W_{\delta(1-\delta)} + W_{\delta\frac{1}{2}} + \frac{1}{2}W_{\frac{1}{2}\frac{1}{2}}$$

recall that

$$W_1 = W_{\frac{1}{2}\frac{1}{2}} + W_{\delta\frac{1}{2}} + W_{\frac{1}{2}(1-\delta)}$$

thus in either case

$$F(x') \geq W_{\delta(1-\delta)} + \frac{1}{2}W_1$$

□

Theorem 2. *Algorithm 1 outputs a maximum directed cut with given sizes of part at least half that of the optimum.*

Proof. It follows from lemma 1 and lemma 2 that the output of algorithm 1 is at least

$$\max(W_{\delta(1-\delta)} + \frac{1}{2}W_1, \delta W_{\delta(1-\delta)} + (1-\delta)W_0 + \frac{1}{2}W_1 + W_2)$$

which is bounded by

$$q = \max(W_{\delta(1-\delta)}, \delta W_{\delta(1-\delta)} + (1-\delta)(W_0 + W_2)) + \frac{1}{2}W_1$$

We have the following 2 cases, if $W_{\delta(1-\delta)} \geq (1-\delta)(W_0 + W_2)$, then

$$q = W_{\delta(1-\delta)} + \frac{1}{2}W_1 \geq \frac{1}{2}(\underbrace{W_{\delta(1-\delta)} + W_0 + W_1 + W_2}_{=\mathrm{LP}})$$

conversely, if $W_{\delta(1-\delta)} < (1-\delta)(W_0 + W_2)$, then

$$q = \delta W_{\delta(1-\delta)} + (1-\delta)(W_0 + W_2) + \frac{1}{2}W_1 > \frac{1}{2}(\underbrace{W_{\delta(1-\delta)} + W_0 + W_1 + W_2}_{=\mathrm{LP}})$$

In either case, this results in a solution which is at least half that of the optimum. □

3 MAX DICUT WITH GSP on unbalanced graphs

We define an unbalanced graph as the following, a graph $G = (V, E)$ is α-unbalanced if for all vertices $v \in V$, the outdegree $deg^+(v) \geq \alpha \cdot deg(v)$, or the indegree $deg^-(v) \geq \alpha \cdot deg(v)$ for some constant $\alpha \in [\frac{1}{2}, 1]$, see figure 4. We consider the outgoing edges from vertices with $deg^-(v) = \alpha \cdot deg(v)$ "bad" edges, and the outgoing edges from vertices with $deg^+(v) = \alpha \cdot deg(v)$ "good" edges. Consider the greedy algorithm **ALG** applied to an unbalanced graph, the greedy algorithm works by adding the the vertice that increases the cut size the most or decreases the cut size the least at each iteration. We can notice the following properties.

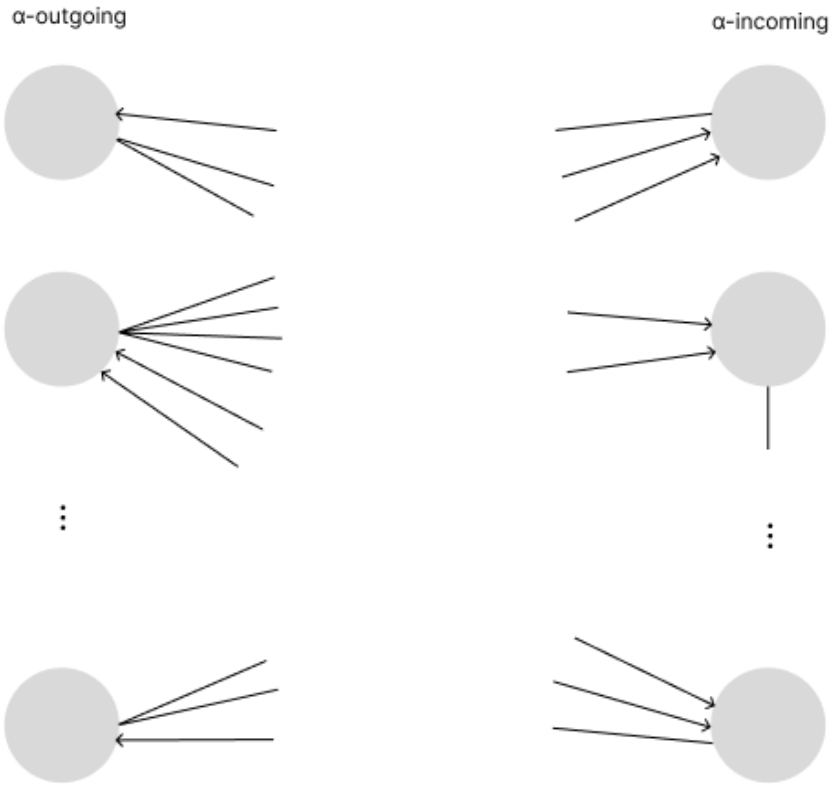

Figure 4: Example of $\alpha = \frac{2}{3}$ unbalanced graph.

Lemma 3. *For any α-unbalanced graph $G = (V, E)$ with $\alpha = 1$, $\boldsymbol{ALG}(G) = \boldsymbol{OPT}(G)$*

Proof. Since $\alpha = 1$ we know that every node either has only outgoing edges or incoming edges. We have 2 cases, either the part size p is less than the number of vertices that only have outgoing edges, or the opposite is true, meaning p is greater than the number of vertices with only outgoing edges. In the former case, the greedy algorithm obviously outputs the optimal cut. In the latter case, the optimal solution must contain all the

vertices with only outgoing edges, this is exactly what the greedy algorithm will also do. As for the remaining vertices with only incoming edges, the optimal cut is obtained by adding the vertices with the least amount of incoming weight to the cut, this is also exactly what the greedy algorithm will do. □

Lemma 4. *For any α-unbalanced graph $G = (V, E)$ with some $\alpha \in [\frac{1}{2}, 1)$, we can construct a new graph $G' = (V, E')$ where $\alpha' = 1$ and $|E| - |E'| \leq (1 - \alpha)|E|$*

Proof. Consider the set B of all vertices v with the "bad" edges, that is,

$$B = \{\forall_{v \in V} \mid deg^-(v) \geq \alpha \cdot deg(v)\}$$

We can construct a new graph $G' = (V, E')$ with $\alpha' = 1$ by removing all these edges. In this new graph we have removed at most

$$|E| - |E'| = \sum_{v \in B} deg^-(v) \leq (1 - \alpha)|E|$$

□

Lemma 5. *For any graph $G = (V, E)$, $\boldsymbol{OPT}(G) \geq \gamma(1 - \gamma)|E|$ where $\gamma = \frac{p}{n}$.*

Proof. A lower bound for $\mathbf{OPT}(G)$ in relation to $|E|$ can be found by analyzing the expected value of the randomized algorithm on a MAX DICUT WITH GSP instance. The algorithm works in the following way, pick p vertices out of the n total vertices uniformly random. Output the resulting cut.

Analysis: Let C be the resulting cut from the random assignment, and let C^* be the optimal cut. Consider an edge $e \in E$, the probability that the edge is in the cut is

$$Pr[e \in C] = \frac{p}{n} \cdot \frac{n - p}{n - 1} \geq \frac{p}{n}(1 - \frac{p}{n}) = \gamma(1 - \gamma)$$

For each $e \in E$, let X_e be a random variable such that $X_e = 1$ if the edge e is in the cut and $X_e = 0$ otherwise. Thus

$$|C| = \sum_{e \in E} w_e X_e$$

We get the following expected value

$$E[|C|] = \sum_{e \in E} E[X_e] = \sum_{e \in E} Pr[e \in C] = \gamma(1 - \gamma)|E| \geq \gamma(1 - \gamma)|C^*|$$

Thus we get the following lower bound for $\mathbf{OPT}(G)$

$$\mathbf{OPT}(G) \geq \gamma(1 - \gamma)|E|$$

□

We define a new algorithm **ALG*** that takes as input an $\alpha-$unbalanced graph G, removes the "bad" edges, it then performs the greedy algorithm on the resulting graph.

Theorem 3. *For any α-unbalanced graph $G = (V, E)$ where $n = |V|$ and part size p, $\boldsymbol{ALG*}(G) \geq (1 - \frac{1-\alpha}{\gamma(1-\gamma)})\boldsymbol{OPT}(G)$, where $\gamma = \frac{p}{n}$.*

Proof. It follows from lemma 3 and 4 that

$$\mathbf{OPT}(G) - \mathbf{ALG*}(G) \leq (1 - \alpha)|E|$$

and thus

$$\mathbf{ALG*}(G) \geq \mathbf{OPT}(G) - (1 - \alpha)|E|$$

It then follows from lemma 5 that

$$\mathbf{ALG*}(G) \geq (1 - \frac{1 - \alpha}{\gamma(1 - \gamma)})\mathbf{OPT}(G)$$

□

4 Discussion

There's certainly more that can be explored regarding unbalanced graph, in this report we have specifically covered how the greedy algorithm for MAX DICUT WITH GSP performs on such graphs but this analysis can be extended to also analyze how the LP (4a) - (4c) behaves on such graphs. The hope is that there exists some properties unique to unbalanced graphs that can be exploited to obtain even better approximations of such graphs than the algorithm devised by Ageev, Sviridenko and Hassin.

References

[AHS01] Alexander Ageev, Refael Hassin, and Maxim Sviridenko. "A 0.5-Approximation Algorithm for MAX DICUT with Given Sizes of Parts". In: *SIAM Journal on Discrete Mathematics* 14.2 (Jan. 2001), pp. 246–255. ISSN: 0895-4801. DOI: 10.1137/S089548010036813X. URL: https://epubs.siam.org/doi/abs/10.1137/S089548010036813X (visited on 09/22/2023).

[AS04] A.A. Ageev and M.I. Sviridenko. "Pipage Rounding: A New Method of Constructing Algorithms with Proven Performance Guarantee". In: *Journal of Combinatorial Optimization* 8.3 (Sept. 1, 2004), pp. 307–328. ISSN: 1573-2886. DOI: 10.1023/B:JOCO.0000038913.96607.c2. URL: https://doi.org/10.1023/B:JOCO.0000038913.96607.c2 (visited on 09/27/2023).

[AS99] Alexander A Ageev and Maxim I Sviridenko. "Approximation algorithms for maximum coverage and max cut with given sizes of parts". In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 1999, pp. 17–30.

[Bra+23] Joshua Brakensiek et al. *Separating MAX 2-AND, MAX DI-CUT and MAX CUT*. 2023. arXiv: 2212.11191 [cs.CC].

[FG95] Uriel Feige and Michel Goemans. "Approximating the Value of Two Prover Proof Systems, With Applications to MAX 2SAT and MAX DICUT". In: (1995).

[FL01] Uriel Feige and Michael Langberg. "Approximation Algorithms for Maximization Problems Arising in Graph Partitioning". In: *Journal of Algorithms* 41.2 (Nov. 1, 2001), pp. 174–211. ISSN: 0196-6774. DOI: 10.1006/jagm.2001.1183. URL: https://www.sciencedirect.com/science/article/pii/S0196677401911836 (visited on 09/23/2023).

[GW95] Michel X. Goemans and David P. Williamson. "Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming". In: *Journal of the ACM* 42.6 (Nov. 1995), pp. 1115–1145. ISSN: 0004-5411, 1557-735X. DOI: 10.1145/227683.227684. URL: https://dl.acm.org/doi/10.1145/227683.227684 (visited on 09/22/2023).

[Kar72] Richard M. Karp. "Reducibility among Combinatorial Problems". In: *Complexity of Computer Computations: Proceedings of a symposium on the Complexity of Computer Computations, held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department*. Ed. by Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger. Boston, MA: Springer US, 1972, pp. 85–103. ISBN: 978-1-4684-2001-2. DOI: 10.1007/978-1-4684-2001-2_9. URL: https://doi.org/10.1007/978-1-4684-2001-2_9.