Correlation Clustering under perturbations

Pierre Dubreuil
Telecom ParisTech
pierre.dubreuil@telecom-paristech.fr

Zaccharie Ramzi
Telecom ParisTech
zaccharie.ramzi@telecom-paristech.fr

Supervised by
Géraud Le Falher

Abstract

We applied theoretical results on graphs with positive edge weights under perturbation, provided by [3], on graphs with positive and negative edge weights. This way, we extended the notion of stability and experimentally exhibited how this stability is preserved for Max Cut and Minimum Multiway Cut problems. All the implemented code and driven experiments are made publicly available.

1 Introduction

Many discrete optimization problems like clustering belongs to the class of unsolvable problems (considering NP $\neq$ P) . Even if those problems are provably hard, we know that this property only refers to worst case graphs. In practice, meaningful instances of graphs have additional structure which allows to design polynomial time solution for clustering as shown in [1]. In a nutshell, clustering is difficult only when we don't care about it.
Here we will focus on two problems that relate to clustering problems:
- Max-Cut problem: we are given a weighted graph $G(V, E, w)$ on n vertices with an adjacency matrix w. The goal is to find a cut $(S, V \backslash S)$ that maximize the total weight of the edges between S and $V \backslash S$. Max Cut is a classic NP-hard problem.
- Minimum Multiway Cut problem: we are given a weighted graph $G(V, E, w)$ on n vertices with an adjacency matrix w and a set of terminals $S = s_1, s_2, ..., s_k \subseteq V$. A multiway cut is a set of edges that leaves all vertices of S in a separate component. The goal of the Minimum Multiway Cut is to find a set of edges $E_2 \subseteq E$ with minimum weight such that removing E_2 from G separates all terminals. In other words, $G(V, E - E_2)$ should have |S| components and each component should contain one and only one terminal from S. It can be seen as a generalization of min-cut to k components with initialization.

Algorithms that solve this problems are used as a basis in [3] to prove strength of γ-stability on graphs. Here, we are looking for partitioning solutions that are robust to some perturbation over the edges weights. The main idea behind that, is that the partitioning solution should be similar if we change the weights on the edges only by a small amount. For a practical example, if the edge weights are rates given by viewers to movies, they might be imprecise (rating 3, 3.1 or 3.2 to a movie is really subjective) or they might be badly scaled (depending on who is rating the movie, the scale is not the same: a 4 for someone is potentially a 3.5 for someone else). Knowing that, a recommendation

system based on graphs is meaningful only if the solution doesn't change under a small perturbation of the ratings (edges weights). Here are some formal definitions of γ-stability concepts that set this idea of "disrupted graphs" and "stable solutions".

Definition 1 (γ-perturbation by [2]) *Let $G = (V, E, w)$ be a weighted graph with edge weights $w(e)$ and let $\gamma > 1$. A weighted graph $G' = (V, E, w')$ is a γ-perturbation of G if for every $(u, v) \in V$,*

$$w(u, v) \leq w'(u, v) \leq \gamma.w(u, v)$$

Definition 2 (γ-stable instance by [2]) *We say that G is a γ-stable instance of Max Cut / Minimum Multiway Cut if there is a unique cut which forms a maximal cut / minimum multiway cut for every γ-perturbation G' of G.*

Definition 3 (by [2]) *Let $\gamma \geq 1$. A weighted graph G with maximal cut (S, S') is γ-stable instance of Max Cut if for every vertex set $T \neq S$ and $T \neq S'$:*

$$w(E(S, S')\ E(T, T')) > \gamma.w(E(T, T')\ E(S, S'))$$

In [3], the authors obtain an algorithm for γ-stable Max Cut instances if γ is sufficiently large. If the instance is γ-stable, it finds the maximum cut, otherwise, it either finds the maximum cut or certifies that the instance is not γ-stable. On the other side, they prove that there is no robust algorithm for γ-stable instances of Max Cut when γ is too small.

Subsequently we will consider a larger class of graphs and try to extend results from [3] to this class of graphs.

Indeed, we consider the problem of partitioning a graph of n vertices, where each edge is labeled either $+$ or $-$ depending on whether the vertices have been deemed to be similar or different. The clustering goal remains to produce a partitioning that maximize the $+$ edges and minimize the $-$ edges in the clusters for Max Cut, and to find a set of edges $E_2 \subseteq E$ with minimum weight such that removing E_2 from the graph separates all terminals for Minimum Multiway Cut.

2 Main results

2.1 Max Cut

2.1.1 SDP relaxations of Max Cut

Formally, we can write the Max Cut problem in the following way. We consider a graph $G = (V, E, w)$.

$$\begin{aligned} \underset{x}{\text{maximize}} \quad & \frac{1}{2} \sum_{(u,v)\in E} w(u, v)(1 - x_u x_v) \\ \text{subject to} \quad & \forall u \in V, x_u \in \{0, 1\} \end{aligned}$$

In this formulation, x is an indicator variable indicating in which cluster belongs each vertex. The matrix form of this formulation is the following.

$$\begin{aligned} \underset{X}{\text{maximize}} \quad & \frac{1}{4} L \bullet X \\ \text{subject to} \quad & X \succeq 0 \\ & rank(X) = 1 \\ & diag(X) = \mathbf{1} \end{aligned}$$

Where $X \bullet Y = Tr(X^T Y)$.
The link with the previous formulation is $X = xx^T$. The first relaxation of this problem is to remove the constraint on the rank of the matrix (which is equivalent to saying that the indicators no longer belong to $\{0, 1\}^n$ but to $\mathbb{R}^n$ and have to be unitary). Therefore the problem becomes an SDP and can be solved efficiently using Goemans-Williamson algorithm [5] which gives a 0.879 approximation. It uses a rounding procedure to determine the best assignment of the vertex to each cluster based on

their indicator vector x_u.
[6] gives another idea : triangle inequalities. This strengthened relaxation allows for further results on instance stability. The final formulation is as follows:

$$\begin{aligned} \underset{X}{\text{maximize}} \quad & \frac{1}{4} L \bullet X \\ \text{subject to} \quad & X \succeq 0 \\ & rank(X) = 1 \\ & diag(X) = 1 \\ & \forall i,j,k, X_{ij} + X_{jk} + X_{ik} \geq -1 \\ & \forall i,j,k, X_{ij} - X_{jk} - X_{ik} \geq -1 \\ & \forall i,j,k, -X_{ij} + X_{jk} - X_{ik} \geq -1 \\ & \forall i,j,k, -X_{ij} - X_{jk} + X_{ik} \geq -1 \end{aligned}$$

2.1.2 Max Cut on perturbed graphs

In this section we will resume the major outcomes from [3] about Max Cut problem on perturbed positive graphs.

Definition 4 (integral solution by [3]) *Let G be an instance of Max Cut. We say that an SDP solution $\{\bar{u}\}$ is integral if there exists a vector $\bar{e}$ such that $\bar{u} = \bar{e}$ or $\bar{u} = -\bar{e}$ for every $u \in V$.*

The authors show that if the SDP problem is γ-stable for a big enough γ, which implies that the graph can be perturbed, then the solution of the SDP is integral. Formally we have this theorem:

Theorem 1 *The SDP relaxation for every γ-stable Max Cut instance with $\gamma > D_{l_1^2 \to l_2}(n)$ is integral. Where $D_{l_1^2 \to l_2}(n)$ is the least distortion with which every n point metric space of negative type embeds into l_1*

This result was previously established but the real breakthrough in this work is the improvement of the bound that has been reduced from $c.n$ for some constant c, to $c.\sqrt{\log n} \log \log n$.

Using this theorem when solving a SDP for real graphs, if the found solution for V is integral, then the Max Cut instance is γ-stable and even if the input graph is restrictedly perturbed, the output of the SDP will stay integral and will exhibit the same clustering as for the non-perturbed graph.

2.1.3 Negative results

If the SDP related to the Max Cut of the graph is γ-stable but γ too small, the graph is not very robust to perturbations, hence the solution will not be integral. Formally we have the following theorem given by [3]:

Theorem 2 *There is no polynomial-time tractable relaxation for Max Cut that is integral on γ-stable instances if $\gamma < D_{l_1^2 \to l_2}(\frac{n}{2})$.*

Unfortunately, there is gap between $D_{l_1^2 \to l_2}(\frac{n}{2})$ and $D_{l_1^2 \to l_2}(n)$ so we cannot conclude on an equivalence between γ-stable instances and integral solutions. However, as $D_{l_1^2 \to l_2}(n) = O(\sqrt{(}log(n))loglogn)$, we can intuitively think that for large n the gap $D_{l_1^2 \to l_2}(n) - D_{l_1^2 \to l_2}(\frac{n}{2})$ will tend to 0.

As a consequence, for large n, in practice we can discriminate graphs as γ-stable or not with the output of the SDP solver. In further applications, it allows us to keep only the robust solutions, like for example for recommendation systems.

On the bad side, as the bound is growing with n, if the graph is large, at some point the definition of robustness isn't very satisfying. A large graph can be 1000-stable but classified as non robust because the solution of the SDP is not integral.

2.2 Minimum Multiway Cut

2.2.1 LP relaxations for Minimum Multiway Cut

Formally, we can write the Minimum Multiway Cut problem in the following way ([4]). Given a weighted graph $G = (V, E, w)$:

$$\begin{aligned} \underset{d}{\text{minimize}} \quad & \sum_{e \in E} w(e)d(e) \\ \text{subject to} \quad & (V, d) \text{ is a semimetric} \\ & \forall t_1, t_2 \in T, t_1 \neq t_2, d(t_1, t_2) = 1 \\ & \forall u, v \in V, d(u, v) \in \{0, 1\} \end{aligned}$$

Where T is the set of terminal nodes and d is an indicator for all edges: we keep the 0 and cut the 1. We also need the definition for a semimetric.

Definition 5 (semimetric by [4]) *A semimetric is a pair (V, d) where V is a set and d is a function $d : V \times V \mapsto \mathbb{R}$ such that $d(u, v) = d(v, u) \geq 0$ for all u, v; $d(u, u) = 0$ for all u; and $d(u, w) \leq d(u, v) + d(v, w)$ for all u, v, w. We sometimes refer to the elements of V as points, and to $d(u, v)$ as the distance between u and v.*

In order to obtain an LP, we relax the last constraint and set it to $0 \leq d(u, v) \leq 1$. However this LP still needs a rounding procedure to give a solution. To avoid this, we strengthen the relaxation by adding two more constraints as suggested in [4], which gives the final formulation :

$$\begin{aligned} \underset{d}{\text{minimize}} \quad & \sum_{e \in E} w(e)d(e) \\ \text{subject to} \quad & (V, d) \text{ is a semimetric} \\ & \forall t_1, t_2 \in T, t_1 \neq t_2, d(t_1, t_2) = 1 \\ & \forall u, v \in V, 0 \leq d(u, v) \leq 1 \\ & \forall u \in V, \sum_{t \in T} d(u, t) = |T| - 1 \\ & \forall u, v \in V, \forall S \subseteq T, d(u, v) \geq \sum_{t \in S} d(u, t) - d(v, t) \end{aligned}$$

This formulation is seemingly different of that suggested in [3], but the proposition 1 of [4] proves that they are equivalent.

2.2.2 Minimum Multiway Cut on perturbed graphs

In this section we will explain the principal theorem from [3] about Minimum Multiway Cut solutions with perturbed positive graphs.

For this problem that implies finding multiple clusters, [3] proves weaker results than for Max Cut where only two clusters are considered. Whereas we add a lower-bound on γ for Max Cut, here we consider 4-stable graphs and one can prove that the linear programming relaxation of Calinescu, Karloff, and Rabani is integral. Thus it also implies that there is a robust polynomial-time algorithm for 4-stable instances of Minimum Multiway Cut. We can sum up this result by a short theorem.

Theorem 3 *The LP relaxation is integral if the instance is 4-stable.*

3 Experimental results

3.1 General purpose

Having set these properties for Max Cut and Minimum Multiway Cut instances, we tried to extend the class of graph that are used as inputs of the problems. To this end, we introduce correlated graphs. Those graphs are a class of graphs where we don't have direct information on nodes but we have a secondary information on whether two nodes are similar or different. In practice it means that the weight of the edge between two nodes is positive if this nodes are similar and negative if this nodes are different. The aim of this section is to explore to what extent the previous theorems are extendable to correlated graphs.

On the implementation side, we choose to implement our work with Python (version 3) mainly to take advantage of *Networkx* a graph library, and of *Picos* an interface for conic optimization solvers that were useful for our project.

3.2 Max Cut

3.2.1 Application of theorem 1 on a γ-stable graph

First, we tried our implementation of the SDP relaxation for Max cut on a hardly γ-stable graph (with only 4 nodes) to give an example of what is an integral solution and to illustrate *theorem 1*.

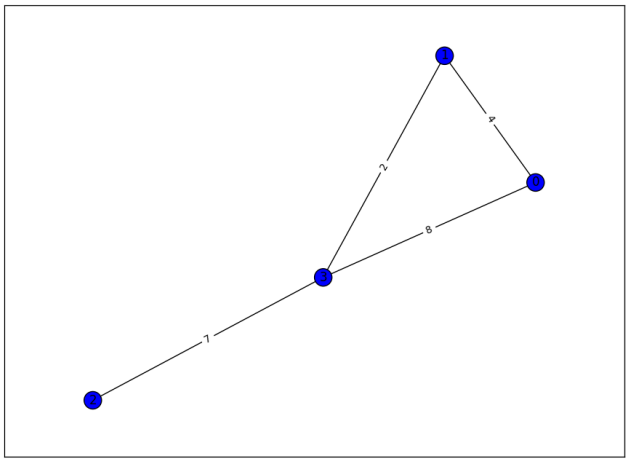

Figure 1: A 2-stable graph G_1 to cluster with Max Cut.

We get as output of our SDP solver the following matrix X:

$$X = \begin{pmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{pmatrix}$$

As X has only 1 and -1, we can conclude that x is integral since $X = x^T.x$. Thus, this result aligns with $theorem1$ as we get an integral solution with a γ-stable graph with γ large enough.
Here we can easily compute $\gamma = 2$, but to definitively set this affirmation we take as input of our SDP solver a 2-perturbed version of G_1 called G_1'. Then, we compare the output clustering for G_1 and G_1'. As expected we get the same clustering for both graph as shown on figure 2.

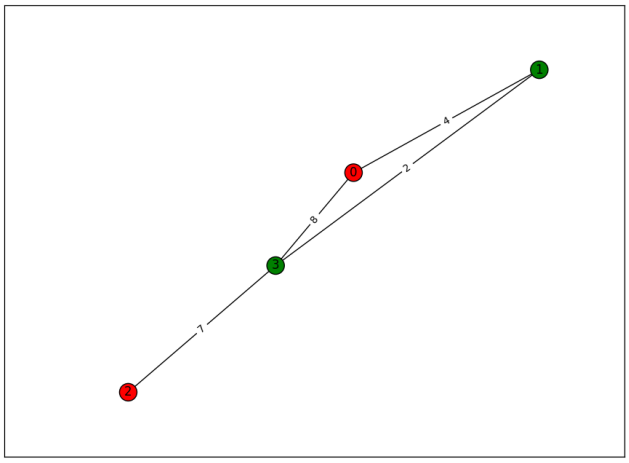

Figure 2: Clustering exhibited for G_1 and G_1' by our SDP solver.

3.2.2 A single example with a correlated graph

The idea here is to see if we can have similar properties on instances of Max Cut if the graph is a correlated graph. To take as most as possible a "real life" example we simulated a graph were viewers where rating movies between -2.5 and 2.5; -2.5 meaning "I hated this movie" and 2.5 being "I highly recommend this movie". We introduce a γ-stability concept for correlated graphs:

Definition 6 (γ-perturbation for correlated graphs) *Let $G = (V, E, w)$ be a correlated graph with edge weights $w(e)$ and let $\gamma > 1$. A weighted graph $G' = (V, E, w')$ is a γ-perturbation of G if for every $(u, v) \in V$,*

$$w(u, v) \leq w'(u, v) \leq \gamma.w(u, v) \text{ if } w(u, v) > 0$$

$$\gamma.w(u, v) \leq w'(u, v) \leq .w(u, v) \text{ if } w(u, v) < 0$$

This way, Definition 2 of γ-stable instances still holds if we distinguish positive and correlated γ-perturbation of a graph.

The graph G_2 we considered here , represents 3 users rating 3 movies as follows.

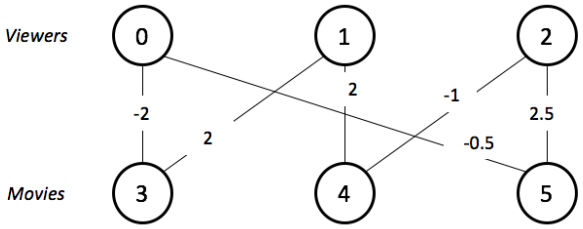

Figure 3: Bipartie representation of the simulated graph G_2

Afterward, we will represent graphs using *Networkx* for a matter of simplicity and we will have to keep in mind that this kind of graph is always bipartite.

To have a meaningful result where clusters represent a set of users and movies that fit together, we performed our SDP solver on the laplacian of $-W$ (equivalent to min cut problem). One can prove that G_2 is 1.99-stable. To check if theorem 1 still holds we performed the SDP solver on G_2 and G_2', his γ-perturbed instance and got an *integral solution* and the same clustering as shown on figure 3

Therefore, we can experimentally make the assumption that theorem 1 still holds for correlated graph.

3.2.3 Correlation clustering on "real life" graphs

With theorems 1 and 2 and the previous extension on correlated graph, we used our algorithm to do clustering on "real life" graphs. We can summarize our approach saying that if the input graph is γ-stable for a sufficiently large γ, the output will be a meaningful clustering. On the other hand if the output is not integral, the algorithm doesn't exhibit any clustering. But even if such a clustering exists, it would be too much linked to the specific input graph whereas we want the clustering to be the same for γ-perturbed instance of the graph and the input graph itself. In that way, we can say that our algorithm exhibit a clustering only when it is robust and returns $false$ when there isn't such a clustering or it's not robust.

To construct "real life" graph we used a .tsv file extracted from themoviedb.org. From this file, we selected a restricted wisely the amount of data to get a small number of vertices n as the number of constraints is cubic in n (due to triangle inequalities). A challenge was to extract a not too sparse graph as many viewers in the database have only seen few movies. To tackle this limitation, we selected viewers that have seen a minimum number of movies, then we consider movies seen at least by two of the selected viewers.

Here are two examples of correlated graphs that have been clustered by the algorithm solving a min cut. A posteriori, we can try to find the value of γ for which the graphs are stable.

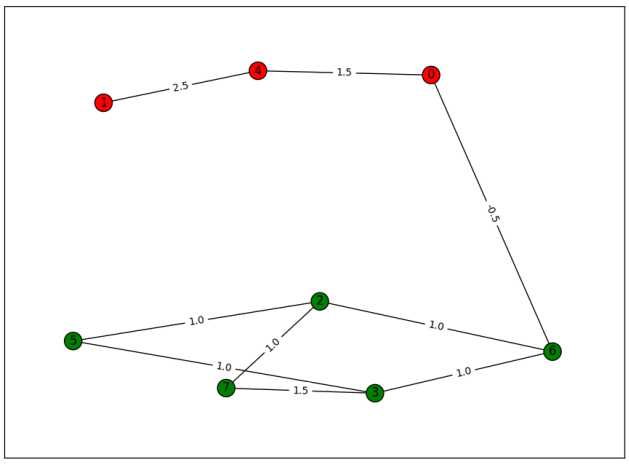

Figure 4: Correlation clustering for real graph G_3

Analysis of G_3 clustering The min cut problem is easy to solve on this graph is has only one negative edge that can be cut to separate G_3 in two clusters. As a consequence, G_3 is ∞-stable and our algorithm output the only clustering for all perturbed instances of the graph.

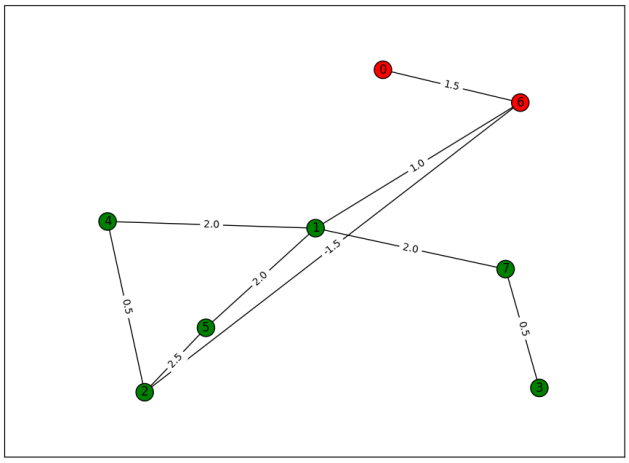

Figure 5: Correlation clustering for real graph G_4

Analysis of G_4 clustering Min cut on this graph is also easy to compute as it only contains one negative edge. Nevertheless, it is only 1.99 stable because if viewer 1 had loved movie 6, the min cut solution is different. What's interesting is that we can't exhibit the good clustering for the 2-perturbated instance of G_4 as it is not γ-stable for a sufficiently large γ and our algorithm outputs $false$.

3.3 Minimum Multiway Cut

We followed the same methodology for Minimum Multiway Cut than for Max Cut. We first proceeded by clustering what we knew to be a 4-stable graph for Minimum Multiway Cut.

3.3.1 Application of theorem 3 to 4-stable graphs

The goal of this simple example is to give an example of theorem 3 constructing a cyclic graph G_5 with positive edges. We can easily see that G_5 is 4 stable. In the LP relaxation for Minimum Multiway Cut problem, an integral solution is represented by a matrix d filled with only 0 and 1. 0 if the edge needs to be cut and 1 if it stays in the cut graph. For G_5 the output of our algorithm gives us an integral distance matrix with large precision and we constructed the clustering that goes along with it.

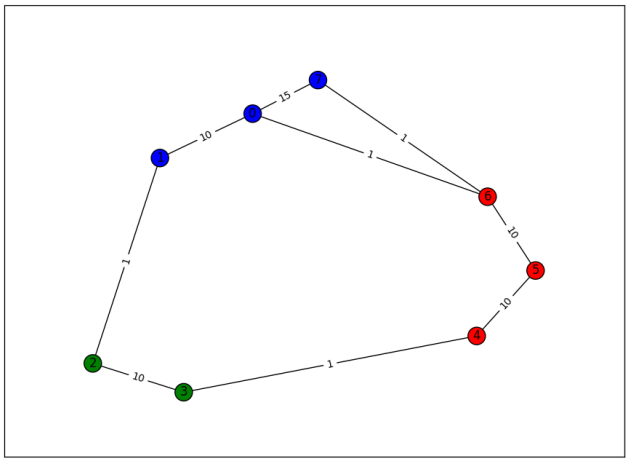

Figure 6: Solution of the Minimum Multiway Cut for G_5 having T=[0, 2, 5]

3.3.2 A single example with correlated graph

As for the Max Cut problem, we extended theorem 3 to correlated graphs using the same class of graphs simulating viewers rating movies.

To give some reality to outputs, we set the terminal nodes to be 3 viewers. The clusters of this graph G_6 will afterward represent a set of movies and viewers that fits together. Here we initialize our algorithm by giving 3 viewers as terminal nodes. The idea is to attribute the set of movies between those viewers.

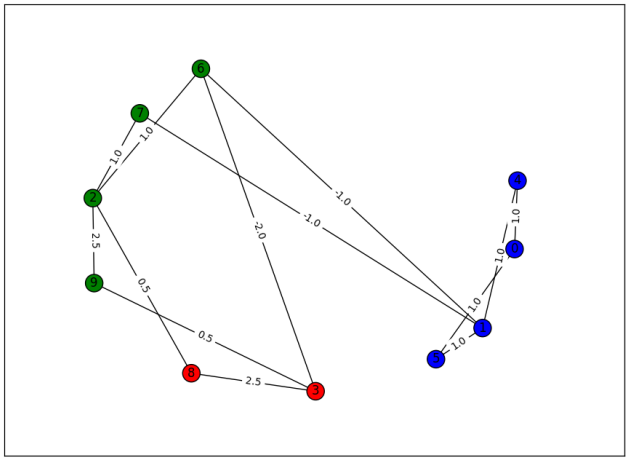

Figure 7: Solution of the Minimum Multiway Cut for G_6 having T=[1, 2, 3]

3.3.3 Multicway Correlation clustering on "real life" graphs

We extended the approach we had for Max Cut problem to the Minimum Multiway Cut. As a reminder, the goal is to create a meaningful clustering on real life graphs extracted from themoviedb.org. Theorem 3 ensure a good clustering if the graph is 4-stable, in other words if the clustering real matters. Indeed, if the graph is not 4-stable, the clustering that we could have exhibited is too much related to the actual instance of the graph that can't be perturbed without changing the clustering . One one hand, for this problem, the solution is weaker than the one for Max Cut as we need a 4-stable graph. One the other hand, the result is much more interesting because we can get as many clusters as we want.

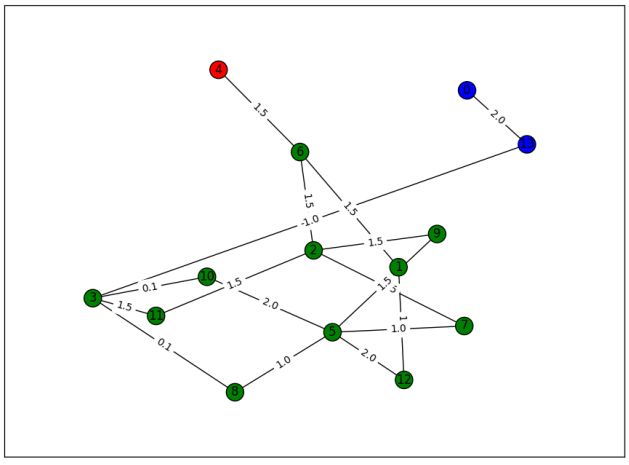

Figure 8: Solution of the Minimum Multiway Cut for G_7 having T=[0, 2, 4]

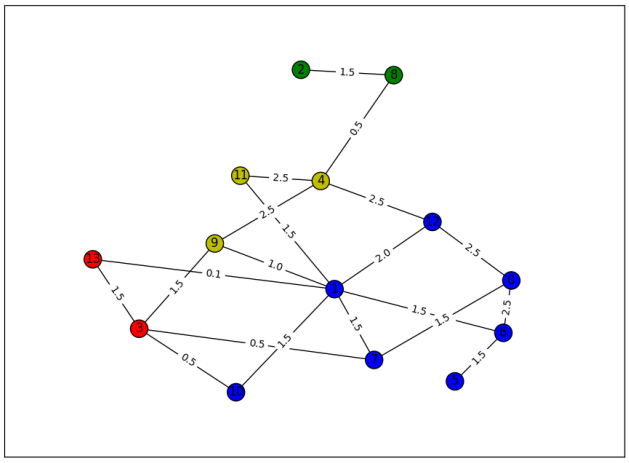

Figure 9: Solution of the Minimum Multiway Cut for G_8 having T=[0, 2, 3, 4]

Even if it is hard to get 4-stable graph, on these two examples we could extend theorem 3 to correlated graph and realize the desired clustering.

4 Conclusion & Future Directions

Before concluding our work, we will present further directions that could be handled.

4.1 Limitation in the number of clusters in Max Cut

The Max k-Cut problem is a generalized Max Cut problem to k clusters. The goal is to partition the graph in k clusters maximizing the weight of the cut edges. One can prove that for $k \geq 3$ there is no polynomial-time algorithm that solves ∞-stable instances. The proof given in [3], follows from the equivalent result for the Unique k-Coloring problem from Barbanchon [7]. One way to work around the problem is to iteratively handle both of the outputs of the algorithm and plug them in the Max Cut algorithm. That way, we could potentially have a similar theorem than theorem 1 for Max k-Cut like problems. Moreover, it could be an interesting way to compare this approach with the LP solver for Minimum Multiway Cut problem for which we know that the conditions on γ are harder to achieve.

4.2 Expert knowledge for the terminal points of Minimum Multiway Cut

In the Minimum Multiway Cut problem, the terminal nodes needs to be given as an input. It could be interesting to see if we couldn't do without this input by first, randomly pick k nodes and then, analyze the output clustering to find terminal nodes that give a lower objective.

4.3 Conclusion

In their paper, [3] presented properties on algorithms for stable instances of Max Cut and Minimum Multiway Cut. Indeed, authors provide a sufficient condition under which there is an algorithm for stable instances of a graph partitioning problem for positive graphs. In our work, we truly implemented those algorithms and constructed examples to experimentally approve their approach. Furthermore, we extended the class of tested graph and conjectured that the established properties are also valid for correlated graphs. Based on that we finally tested our algorithm as clustering algorithm on real life graphs.

References

[1] Shai Ben-David, "Clustering is Easy When ....What?", 2012.

[2] Yonatan Bilu and Nathan Linial," Are stable instances easy?", in *Innovations in Computer Science* pages 332–341, 2010.

[3] Konstantin Makarychev Yury Makarychev Aravindan and Vijayaraghavan, *Bilu–Linial Stable Instances of Max Cut and Minimum Multiway Cut*, 2013.

[4] Gruia Calinescu; Howard Karloff and Yuval Rabani, "An improved approximation algorithm for multiway cut.", In *Proceedings of the Symposium on Theory of Computing*, pages 48–52, 1998.

[5] Michel X. Goemans and David P. Williamson. .879-approximation algorithms for MAX CUT and MAX 2SAT. In *Proceedings of the Twenty-Sixth Annual Symposium on Theory of Computing*, pages 422–431, 1994.

[6] Christoph Helmberg, Franz Rendl, Robert J. Vanderbei, and Henry Wolkowicz. An Interior-Point Method for Semidefinite Programming. In *SIAM Journal on Optimization*, pages 342-361, 1996.

[7] Regis Barbanchon," On unique graph 3-colorability and parsimonious reductions in the plane", Theoretical computer science, 2004.