# Repulsive springs and approximating the maximum cut

A *cut* in a graph $G = (V, E)$ is the set of edges connecting a set $S \subseteq V$ to $V \setminus S$, where $\emptyset \subset S \subset V$. The *Maximum Cut Problem* is to find a cut with maximum cardinality. We denote by $\mathsf{maxcut}(G)$ this maximum. (More generally, we can be given a weighting $w : V \to \mathbb{R}_+$, and we could be looking for a cut with maximum total weight. To keep things simple, however, we restrict our introductory discussions to the unweighted case.)

The Maximum Cut Problem is NP-hard; one natural approach is to find an "approximately" maximum cut. Formulated in different terms, Erdős in 1967 described the following simple heuristic algorithm for the Maximum Cut Problem: for an arbitrary ordering $(v_1, \ldots, v_n)$ of the nodes, we color $v_1, v_2, \ldots, v_n$ successively red or blue. For each $i$, $v_i$ is colored blue iff the number of edges connecting $v_i$ to blue nodes among $v_1, \ldots, v_{i-1}$ is less than the number of edges connecting $v_i$ to red nodes in this set. Then the cut formed by the edges between red and blue nodes contains at least half of all edges. In particular, we get a cut that is at least half as large as the maximum cut.

There is an even easier randomized algorithm to achieve this approximation, at least in expected value. Let us 2-color the nodes of $G$ randomly, so that each node is colored red or blue independently, with probability 1/2. Then the probability that an edge belongs to the cut between red and blue is 1/2, and expected number of edges in this cut is $|E|/2$.

Both of these algorithms show that the maximum cut can be approximated from below in polynomial time with a multiplicative error of at most 1/2. Can we do better? The following strong negative result of Hastad [4] shows that we cannot get arbitrarily close to the optimum:

**Proposition 2.0.1** *It is NP-hard to find a cut with more than* $(16/17)\mathsf{maxcut}(G) \approx .94\,\mathsf{maxcut}(G)$ *edges.*

Building on results of Delorme, Poljak and Rendl [2, 6], Goemans and Williamson [3] give a polynomial time algorithm that approximates the maximum cut in a graph with a relative error of about 13%:

**Theorem 2.0.2** *One can find in polynomial time a cut with at least* $.878\,\mathsf{maxcut}(G)$ *edges.*

What is this strange constant? From the proof below we will see that it can be defined as $2c/\pi$, where $c$ is the largest positive number for which

$$\arccos t \geq c(1-t) \tag{2.1}$$

for $-1 \leq t \leq 1$. This would seem like a random biproduct of a particular proof; however, it turns out that if we use the complexity theoretic hypothesis called the "Unique Games Conjecture" (stronger than $P \neq NP$), then this constant is optimal (Khot, Kindler, Mossel,

O'Donnell [5]): no polynomial time algorithms can approximate the max cut with a better approximation ratio for all graphs.

The algorithm of Goemans and Williamson makes use of the following geometric construction. We want to find a representation $i \mapsto \mathbf{u}_i$ $(i \in V)$ of the nodes of the graph in the unit sphere in $\mathbb{R}^d$ so that the following "energy" is maximized:

$$\mathcal{E}(\mathbf{u}) = \sum_{ij \in E} \frac{1}{4}(\mathbf{u}_i - \mathbf{u}_j)^2 = \frac{1}{2} \sum_{ij \in E} (1 - \mathbf{u}_i^{\mathsf{T}} \mathbf{u}_j).$$

We can think of replacing the rubber bands by (very peculiar) repulsive strings, which push their endpoints apart with a force that increases proportionally with the length.

If we work in $\mathbb{R}^1$, then the problem is equivalent to the Maximum Cut problem: each node is represented by either 1 or $-1$, and the edges between differently labeled nodes contribute 1 to the energy, the other edges contribute 0. Hence the maximum energy $\mathcal{E}_{\max}$ is an upper bound on the maximum size $\mathsf{maxcut}(G)$ of any cut.

Unfortunately, the argument above also implies that for $d = 1$, the optimal embedding is NP-hard to find. While I am not aware of a proof of this, it is probably NP-hard for $d = 2$ and more generally, for any fixed $d$. The surprising fact is that for $d \geq n$, such an embedding can be found in polynomial time using semidefinite optimization.

Let $X$ denote the $V \times V$ matrix defined by $X_{ij} = \mathbf{u}_i^{\mathsf{T}} \mathbf{u}_j$. Then $X$ satisfies the constraints:

$$X \succeq 0, \tag{2.2}$$

$$X_{ii} = 1, \tag{2.3}$$

and the energy $\mathcal{E}(u)$ can be expressed as

$$\frac{1}{2} \sum_{ij \in E} (1 - X_{ij}). \tag{2.4}$$

Conversely, if $X$ is a $V \times V$ matrix satisfying (2.2) and (2.3), then we can write it as a Gram matrix of vectors in $\mathbb{R}^n$, these vectors will have unit length, and (2.4) gives the energy.

The semidefinite optimization problem of maximizing (2.4), subject to (2.2) and (2.3), can be solved in polynomial time (with an arbitrarily small relative error). So $\mathcal{E}_{\max}$ is a polynomial time computable upper bound on the size of the maximum cut.

How good is this bound? And how to construct an approximately optimum cut from this representation? Here is the simple but powerful trick: *take a random hyperplane $H$ through the origin in $\mathbb{R}^n$*. The partition of $R^d$ given by $H$ yields a cut in our graph. Since the construction pushes adjacent points apart, one expects that the random cut will intersect many edges.

To be more precise, let $ij \in E$ and let $\mathbf{u}_i, \mathbf{u}_j \in S^{n-1}$ be the corresponding vectors in the representation constructed above. It is easy to see that the probability that a random

hyperplane $H$ through 0 separates $\mathbf{u}_i$ and $\mathbf{u}_j$ is $(\arccos \mathbf{u}_i^\mathsf{T}\mathbf{u}_j)/\pi$. Thus the expected number of edges intersected by $H$ is

$$\sum_{ij\in E} \frac{\arccos \mathbf{u}_i^\mathsf{T}\mathbf{u}_j}{\pi} \geq \sum_{ij\in E} c\,\frac{1-\mathbf{u}_i^\mathsf{T}\mathbf{u}_j}{\pi} = \frac{2c}{\pi}\,\mathcal{E}_{\max} \geq \frac{2c}{\pi}\,\mathsf{maxcut}(G).$$

This completes the analysis of the algorithm.

One objection to the above algorithm could be that it uses random numbers. In fact, the algorithm can be *derandomized* by well established but non-trivial techniques. We do not discuss this issue here; see e.g. [1], Chapter 15 for a survey of derandomization methods.

# Bibliography


[1] N. Alon and J.H. Spencer: *The Probabilistic Method*, Wiley, New York, 1992.

[2] C. Delorme and S. Poljak: Laplacian eigenvalues and the maximum cut problem, *Math. Programming* **62** (1993) 557–574.

[3] M. X. Goemans and D. P. Williamson: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *J. Assoc. Comput. Mach.* **42** (1995), 1115–1145.

[4] J. Håstad: Some optimal in-approximability results, *Proc. 29th ACM Symp. on Theory of Comp.*, 1997, 1–10.

[5] S. Khot, G. Kindler, E. Mossel and R. O'Donnell: Optimal inapproximability results for MAX-CUT and other two-variable CSP's, *SIAM Journal on Computing* **37**, 319–357.

[6] S. Poljak and F. Rendl: Nonpolyhedral relaxations of graph-bisection problems, DIMACS Tech. Report 92-55 (1992).