Mathematical Modelling of the Max 2-Cut Problem and Solving the Relaxed Model

January 26, 2022

Montaz Ali
School of Computer Science and Applied Mathematics
Witwatersrand University, Johannesburg

1 Max-Cut Problem

The maximum cut (Max-Cut) problem is one of the simplest graph partitioning problems, yet it is one of the most difficult combinatorial optimization problems to solve. The objective of Max-Cut is to partition the set of vertices of a (undirected) graph $G = (V, E)$ into two subsets, such that the sum of the weights (cut value) of the edges having one endpoint in each of the subsets is maximum. This problem is known to be NP-Complete.

Let $G = (V, E)$ be an undirected and connected graph, where $V = \{1, \cdots, n\}$, and E is the edge set. If the edge $(i, j) \in E$ connects vertices $i, j \in V$, we associate a weight $w_{ij} \geq 0$ with the edge, $w_{ij} = w_{ji}$. We denote the number of edges by m. If (i, j) is not edge in the given graph then we consider $w_{ij} = w_{ji} = 0$.

The more general problem is known as the Max-k-Cut problem (see Fig. 1) which reduces to Max-Cut (Max-2-Cut) for k=2. The max-k-cut problem is to partition V into subsets $\{V_1, V_2, \cdots, V_k\}$, $k \in [2, n]$, $V_i \neq \emptyset$, $V_i \cap V_j = \emptyset$, $\forall i \neq j$, such that the weight function

$$w(V_1, V_2, \cdots, V_k) = \sum_{1 \leq r < s \leq k} \sum_{i \in V_r, j \in V_s} w_{ij}, \tag{1}$$

is maximized, where the edge weights $w_{ij} = w_{ji}$ are such that $w_{ij} = 0$ for $[i, j] \notin E$.

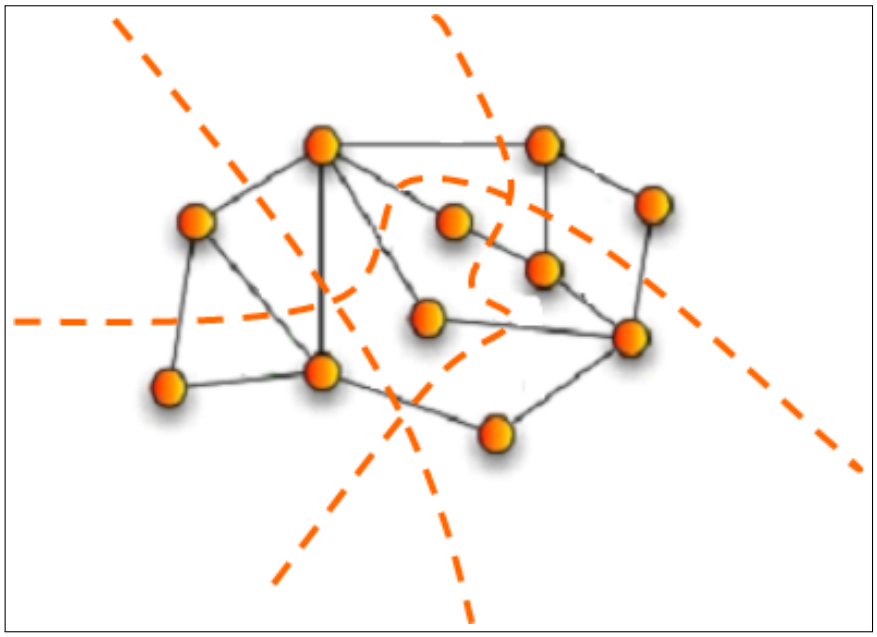

Figure 1: Example of max k-cut

We now present the Max-Cut problem as a mathematical optimization problem. In the Max-Cut problem (see Fig. 2) a cut as a partition of V into two disjoint subsets (not necessarily equal) S and $\bar{S}$ (=$V \setminus S$).

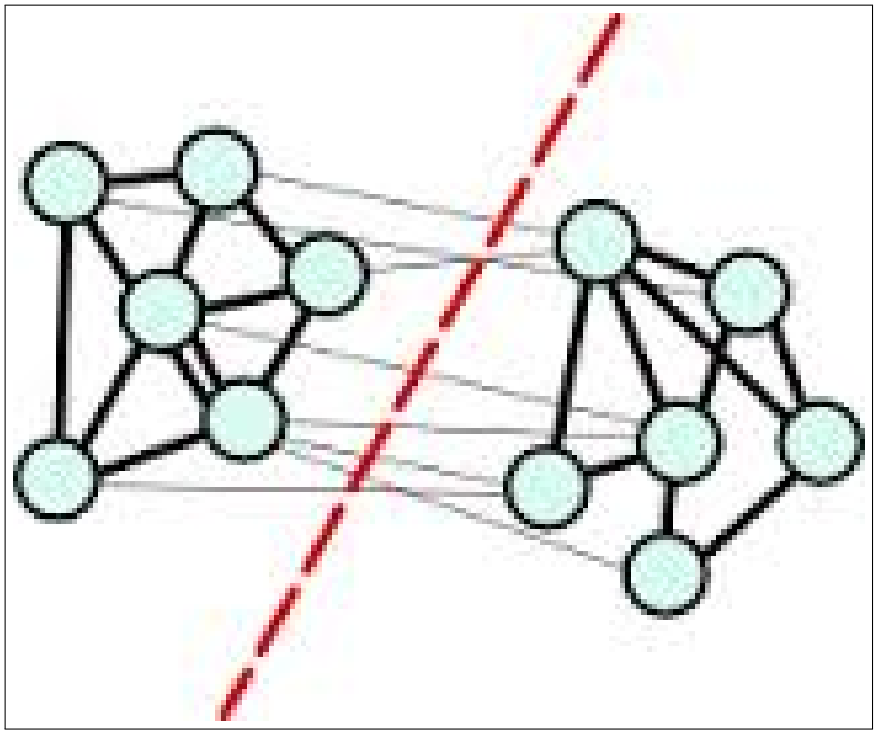

Figure 2: Example of max 2-cut

The weight (cut value) of the cut $(S, \bar{S})$ is given by the function $W : S \times \bar{S} \to R$ and is

defined as

$$W(S, \bar{S}) = \sum_{i \in S,\, j \in \bar{S}} w_{ij}.$$

Clearly, a maximum cut is defined by the following optimization problem:

$$\text{Max-Cut} = \max_{\forall S \subseteq V} W(S, V \setminus S).$$

We can formulate Max-Cut as the following integer quadratic programming problem:

$$\begin{cases} \max_x & \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}(1 - x_i x_j) \\ s.t. & x_i \in \{1, -1\}, i \in \{1, \cdots, n\}. \end{cases}$$

The problem can be written equivalently as

$$\begin{cases} \max & f(x) = \frac{1}{2} \sum_{1 \le i < j \le n} w_{ij}(1 - x_i x_j) \\ s.t. & x_i \in \{1, -1\}, i \in V. \end{cases} \tag{2}$$

To check the above formulation (2) is correct, we define the subset $S \subseteq V$ such that $S = \{i \big| x_i = 1\}$. It can be easily seen that S induces a cut $(S, \bar{S})$ with corresponding weight (cut value) equal to:

$$W(S, V \setminus S) = \frac{1}{2} \sum_{1 \le i < j \le n} w_{ij}(1 - x_i x_j).$$

Consider the following example graph in Fig. 3 and write down the mathematical model the max 2-cut problem for the graph where you may want to take some some $w_{ij} < 0$ when testing your examples. Notice that $x_i x_j = -1$ exactly when the edge (i, j) crosses the cut.

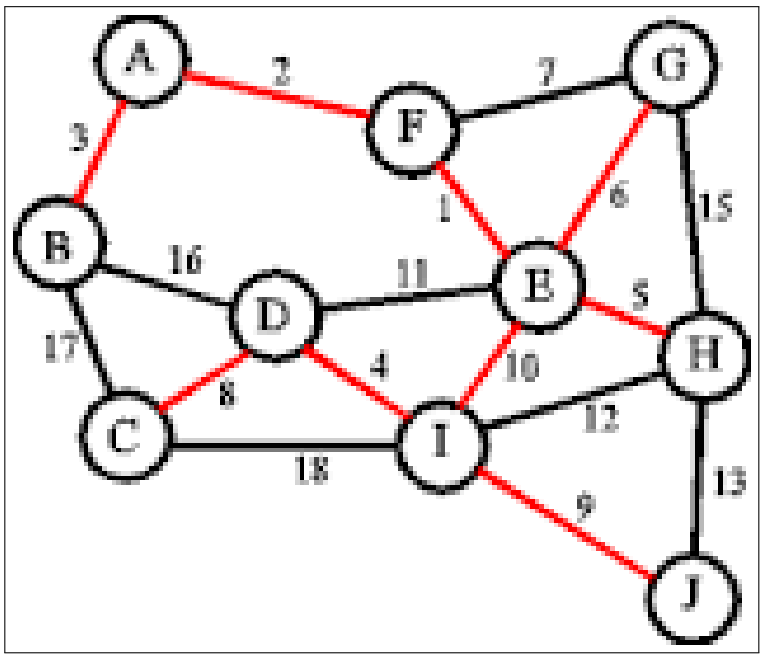

Figure 3: Example graph for max 2-cut

1.1 The SDP Formulation

The general form of the SDP is given as follows:

$$\begin{array}{ll} \max C \bullet X, & s.t \\ A_i \bullet X = b_i & \forall i \\ X \succcurlyeq 0 & \end{array} \tag{3}$$

where $X \succcurlyeq 0$ means the matrix X is positive semi-definite; $A \bullet B = \langle A, B \rangle = Trace(A^T B) = \sum_i \sum_j a_{ij} b_{ij}$ is the matrix inner product of matrices A and B. It is easy to prove that $trace(AB) = Trace(BA) = Trace(B^T A^T)$.

The first step of converting the integer program into an SDP is known as relaxation. A relaxation of an optimization program is another optimization program which, ideally, is easier to solve and and every solution of original program is also a solution of the new program with the same (or related) objective value. Our scheme is as follows:

- convert the integer program into a semi-definite program (SDP),
- solve the semi-definite program,
- and then convert the solution of SDP into an integer solution $\{1, -1\}$ again.

In the case of Eqn (2), we change the domain of x_i's to be unit vectors instead of integers in $\{1, -1\}$. In particular, we write problem as

$$\begin{array}{l} \max_{y_i} \frac{1}{2} \sum_{(i,j) \in E} w_{ij}(1 - y_i^T y_j) \\ s.t. \quad ||y_i|| = 1, i \in V, y_i \in R^n. \end{array} \tag{4}$$

The above problem has an equivalent SDP formulation as follows:

$$\begin{array}{l} \max \frac{1}{2} \sum_{(i,j) \in E} w_{ij}(1 - y_i^T y_j) \\ s.t. \quad ||y_i|| = 1, i \in V, y_i \in R^n. \end{array} \tag{5}$$

where we have written $y_i = (0, 0, \cdots, x_i)^T$. We now introduce $X_{ij} = y_i^T y_j$ and write the above problem as

$$\begin{array}{l} \max_{y_i, X} \frac{1}{4} \sum_{i,j \in V} w_{ij}(1 - X_{ij}) \\ s.t. \quad X_{ii} = 1, i \in V, y_i \in R^n. \\ X \succcurlyeq 0 \end{array} \tag{6}$$

(The extra factor of $1/2$ is because we count each edge (i, j) twice now.) Consider the following notation

$$L_{ij} = \begin{cases} \sum_k w_{ik} & \text{if } i = j \\ -w_{ij} & \text{if } i \neq j \end{cases}$$

It now follows that

$$\begin{aligned}
\tfrac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(1-X_{ij}) &= \tfrac{1}{4}\left(\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}X_{ij}\right) \\
&= \tfrac{1}{4}\left(\sum_{i}\left(\sum_{j} w_{ij}\right) + \sum_{i,i\neq j}\sum_{j,j\neq i} L_{ij}X_{ij}\right) \\
&= \tfrac{1}{4}\left(\sum_{i} L_{ii}X_{ii} + \sum_{i\neq j} L_{ij}X_{ij}\right) \\
&= \tfrac{1}{4}\langle L, X\rangle \\
&= \tfrac{1}{4} L \bullet X
\end{aligned}$$

Hence the resulting SDP is as follows:

$$\begin{aligned}
&\max \tfrac{1}{4} L \bullet X, \qquad s.t \\
&X \bullet e_i e_i^T = 1 \quad \forall i \\
&X \succcurlyeq 0
\end{aligned} \tag{7}$$

where $X_{ii} = X \bullet e_i e_i^T$ and e_i is the i-th coordinate vector.