Integer and Combinatorial Optimization: Expressing MAXCUT as a Semidefinite Program

John E. Mitchell

Department of Mathematical Sciences
RPI, Troy, NY 12180 USA

March 2019
The **MaxCut** Problem

Given a graph $G = (V, E)$, a cut partitions $V$ into two sets $V_1$ and $V_2$. An edge $e = (u, v)$ is in the cut if exactly one of its endpoints is in $V_1$. We want to choose a partition so as to maximize the weight of the edges with exactly one endpoint in each set.

In the SDP approach, we formulate the problem using variables for the vertices, and construct a quadratic integer formulation. Relaxing the integrality requirement leads to a semidefinite program.
With \( V_1 = \{1, 4, 5, 7\} \) and \( V_2 = \{2, 3, 6\} \), the cut has value
\[
8 + 4 + 5 + 2 + 6 + 6 + 3 + 5 + 2 = 41.
\]
Reformulating as an SDP

Let $G = (V, E)$ be a graph with edge weights $w_e$.

Let $n = |V|$.

For any subset $U \subseteq V$, let $\delta(U)$ denote the edges in $E$ with exactly one endpoint in $U$ and let $E(U)$ denote the set of edges with both endpoints in $U$.

Given a partition of $V$ into $V_1$ and $V_2$, the value of the corresponding cut can be expressed in two ways as

$$z(V_1, V_2) = \sum_{e \in \delta(V_1)} w_e = \sum_{e \in E} w_e - \sum_{e \in E(V_1)} w_e - \sum_{e \in E(V_2)} w_e.$$

For notational convenience, for any missing edge $(u, v) \in (V \times V) \setminus E$, we define $w_{uv} = 0$. 

Another representation \( e \in E(V_1): e = (j,k) \). \( x_j = 1, x_k = 1 \).
\( \omega_{jk} = \omega_{jk} x_j x_k \).

Combining the two formulations for MAXCUT, we also have

\[
z(V_1, V_2) = \sum_{e \in \delta(V_1)} w_e = \sum_{e \in E} w_e - \sum_{e \in E(V_1)} w_e - \sum_{e \in E(V_2)} w_e
\]

\[
= 0.5 \left( \sum_{e \in E} w_e - \sum_{e \in E(V_1)} w_e - \sum_{e \in E(V_2)} w_e + \sum_{e \in \delta(V_1)} w_e \right)
\]

\[
= 0.25 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} - 0.25 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_i x_j
\]

with \( x_i = 1 \) if \( i \in V_1 \) and \( x_i = -1 \) if \( i \in V_2 \).
The Laplacian Matrix

This can be expressed more concisely in terms of the Laplacian matrix of the weighted graph:

\[ L_G = D_G - W_G, \]

where the entries of \( W_G \) are the edge weights \( w_e \) and where \( D_G \) is a diagonal matrix with

\[ D_G(i, i) = \sum_{j \in V} w_{ij}. \]
Laplacian matrix for the example graph

\[
L_G = \begin{bmatrix}
14 & -5 & -2 & -7 & 0 & 0 & 0 \\
-5 & 12 & -4 & 0 & 0 & 0 & -3 \\
-2 & -4 & 25 & -2 & -6 & -5 & -6 \\
-7 & 0 & -2 & 20 & -3 & -8 & 0 \\
0 & 0 & -6 & -3 & 13 & -4 & 0 \\
0 & 0 & -5 & -8 & -4 & 22 & -5 \\
0 & -3 & -6 & 0 & 0 & -5 & 14 \\
\end{bmatrix}
\]
Writing the value using the Laplacian matrix

\[
    z(V_1, V_2) = 0.25 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} - 0.25 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_i x_j \\
    = 0.25x^T L_G x
\]

with \( x_i = 1 \) if \( i \in V_1 \) and \( x_i = -1 \) if \( i \in V_2 \).

Hence, the \textsc{MaxCut} problem can be written as the \textit{quadratic binary} problem

\[
    \begin{align*}
    \max_x & \quad 0.25x^T L_G x \\
    \text{subject to} & \quad x_i = \pm 1 \quad \forall i \in V.
\end{align*}
\]  

(1)

\[\text{MaxCut as an SDP}\]
\[0.25 \sum_{i} \sum_{j} w_{ij} x_i x_j - 0.25 \sum_{i} \left( \sum_{j} w_{ij} \right) x_i^2 = 0.25 \sum_{i} \sum_{j} w_{ij} x_i x_j - 0.25 \sum_{i} \sum_{j} w_{ij} x_i^2\]

\[= 0.25 \sum_{i} \sum_{j} L_{ii} x_i^2 + 0.25 \sum_{i} \sum_{j} L_{ij} x_i x_j\]

diagonal entries of Laplacian

\[= 0.25 \sum_{i} L_{ii} x_i^2 + 0.25 \sum_{i} \sum_{j} L_{ij} x_i x_j\]

\[= 0.25 x^T L g x\]
Rewrite the problem

This problem can then be relaxed to a semidefinite program.

First, note from properties of the trace function that

\[ x^T L_G x = \text{trace}(x^T L_G x) = \text{trace}(L_G(xx^T)) \]

Now, introduce an \( n \times n \) matrix \( X \).

We can express problem (1) equivalently as

\[
\begin{align*}
\max_{x, X} & \quad 0.25\text{trace}(L_G X) \\
\text{subject to} & \quad x_i = \pm 1 \quad \forall i \in V \\
& \quad X = xx^T
\end{align*}
\]

which in turn is equivalent to the problem

\[
\begin{align*}
\max_{x, X} & \quad 0.25\text{trace}(L_G X) \\
\text{subject to} & \quad X_{ii} = 1 \quad \text{for } i = 1, \ldots, n \\
& \quad X = xx^T
\end{align*}
\]
\[ x^T L_g x = \text{trace} \left( x^T L_g x \right) = \text{trace} \left( L_g x x^T \right) \]

For any matrices \( A, B \) with \( AB, BA \) both defined, have \( \text{trace} \left( AB \right) = \text{trace} \left( BA \right) \)

\[ = \sum_i \left( \sum_j A_{ij} B_{ji} \right) \]
Relax the conditions on $X$

$$d^T X d = d^T x x^T d = (d^T x)^2$$

Since $X = xx^T$, we must have that $X$ is symmetric and positive semidefinite.

Furthermore, it must have rank equal to 1.

Relaxing the restriction on the rank, we get the following **semidefinite programming relaxation of MaxCut**:

$$\begin{align*}
\max_X & \quad 0.25 \text{trace} (L_G X) \\
\text{subject to} & \quad X_{ii} = 1 \quad \text{for } i = 1, \ldots, n \\
& \quad X \succeq 0,
\end{align*}$$

(2)

where the notation $X \succeq 0$ is equivalent to the requirement that $X$ be symmetric and positive semidefinite.
Example

In the example, the optimal value of the SDP relaxation is 47.

The optimal solution $X$ does have rank equal to one, so factorizing $X$ gives the optimal solution $x$ to the MAXCUT instance:

$$
x_i = 1 \implies i \in V_1 = \{1, 3, 6\}, \quad x_i = -1 \implies i \in V_2 = \{2, 4, 5, 7\}
$$
A bad case for the SDP relaxation

The SDP relaxation doesn’t always give the optimal solution.

For example, for a cycle of length 5 with every edge length equal to one, the MaxCut has value 4, but the value of the SDP relaxation is 4.5224 and the optimal matrix $X$ has rank 2.

The ratio $4/4.5224 = 0.8845$. 