Approximation Algorithms

Chapter 26

Semidefinite Programming

Zacharias Pitouras

Introduction

- LP place a good lower bound on OPT for NP-hard problems
- Are there other ways of doing this?
- Vector programs provide another class of relaxations
- For problems expressed as strict quadratic programs
- Vector programs are equivalent to semidefinite programs
- Semidefinite programs can be solved in time polynomial in n and $\log(1/\varepsilon)$
- A 0.87856 factor algorithm for MAX-CUT

Contents

The maximum cut problem

- MAX-CUT
- Given an undirected graph $G=(V,E)$, with edge weights $w\colon E \longrightarrow \mathbf{Q}^{+}$, find a partition $(S,\bar{S})$ of V so as to maximize the total weight of edges in this cut, i.e., edges that have one endpoint in S and one endpoint in $\bar{S}$.

Strict quadratic programs and vector programs (1)

- A quadratic program is the problem of optimizing a quadratic function of integer valued variables, subject to quadratic constraints on these variables.
- *Strict quadratic program*: monomials of degree 0 or 2.
- Strict quadratic program for MAX-CUT:

y_i an indicator variable for vertex υ_i with values +1 or -1.

Partition, $S = \{\upsilon_i \mid y_i = 1\}, \overline{S} = \{\upsilon_i \mid y_i = -1\}$

If υ_i and υ_j on opposite sides, then $y_i y_j = -1$

and edge contributes w_{ij} to objective function

On the other hand, edge makes no contribution.

Strict quadratic programs and vector programs (2)

- An optimal solution to this program is a maximum cut in G.

$$
\begin{aligned}
\text{maximize} \quad & \frac{1}{2}\sum_{1\le i<j\le n} w_{ij}\left(1-y_i y_j\right) \\
\text{subject to} \quad & y_i^2 = 1, \qquad \upsilon_i \in V \\
& y_i \in \mathbf{Z}, \qquad \upsilon_i \in V
\end{aligned}
$$

Strict quadratic programs and vector programs (3)

- This program relaxes to a vector program
- A vector program is defined over n vector variables in $\mathbf{R}^n$, say $v_1, v_2, \ldots, v_n$, and is the problem of optimizing a linear function of the inner products $v_i \cdot v_j$ for $1 \le i \le j \le n$, subject to linear constraints on these inner products
- A vector program is obtained from a linear program by replacing each variable with an inner product of a pair of these vectors

Strict quadratic programs and vector programs (4)

- A strict quadratic program over n integer variables defines a vector program over n vector variables in $\mathbf{R}^n$
- Establish a correspondence between the n integer variables and the n vector variables, and replace each degree 2 term with the corresponding inner product
- $y_i \,.\, y_j$ is replaced with $v_i \,.\, v_j$

Strict quadratic programs and vector programs (5)

- Vector program for MAX-CUT

$$\text{maximize} \quad \frac{1}{2}\sum_{1\le i<j\le n} w_{ij}\left(1-\upsilon_i\upsilon_j\right)$$

$$\text{subject to} \quad \upsilon_i\cdot\upsilon_i=1, \qquad \upsilon_i\in V$$

$$\upsilon_i\in \mathbf{R}^n, \qquad \upsilon_i\in V$$

Strict quadratic programs and vector programs (6)

- Because of the constraint $\upsilon_i \cdot \upsilon_j = 1$, the vectors $\upsilon_1, \upsilon_2, \ldots, \upsilon_n$ are constrained to lie on the n-dimensional sphere S_{n-1}
- Any feasible solution to the strict quadratic program of MAX-CUT yields a solution to the vector program

  $(y_i, 0, \ldots, 0)$ is assigned to υ_i

- The vector program corresponding to a strict quadratic program is a relaxation of the quadratic program providing an upper bound on OPT
- Vector programs are approximable to any desired degree of accuracy in polynomial time

Properties of positive semidefinite matrices (1)

- Let ***A*** be a real, symmetric *n*   *n* matrix
- Then ***A*** has real eigenvalues and has *n* linearly independent eigenvectors
- ***A*** is positive semidefinite if
  $\forall x \in \mathbf{R}^n, x^T \boldsymbol{A} x \geq 0$
- **Theorem 26.3** *Let* ***A*** *be a real symmetric n   n matrix. Then the following are equivalent:*
  1. $\forall x \in \mathbf{R}^n, x^T \boldsymbol{A} x \geq 0$
  2. *All eigenvalues of* ***A*** *are nonnegative*
  3. *There is an n   n real matrix* ***W***, *such that*
     $\boldsymbol{A} = \boldsymbol{W}^T \boldsymbol{W}$

Properties of positive semidefinite matrices (2)

Proof: $(1 \Rightarrow 2)$: Let λ be an eigenvalue of $\boldsymbol{A}$, and let $\boldsymbol{v}$ be a corresponding eigenvector. Therefore, $\boldsymbol{Av} = \lambda \boldsymbol{v}$. Pre-multiplying by $\boldsymbol{v}^T$ we get $\boldsymbol{v}^T \boldsymbol{Av} = \lambda \boldsymbol{v}^T \boldsymbol{v}$. Now, by (1), $\boldsymbol{v}^T \boldsymbol{Av} \geq 0$. Therefore, $\lambda \boldsymbol{v}^T \boldsymbol{v} \geq 0$. Since $\boldsymbol{v}^T \boldsymbol{v} > 0$, $\lambda \geq 0$.

$(2 \Rightarrow 3)$: Let $\lambda_1, \ldots, \lambda_n$ be the n eigenvalues of $\boldsymbol{A}$, and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be the corresponding complete collection of orthonormal eigenvectors. Let $\mathbf{Q}$ be the matrix whose columns are $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$, and $\boldsymbol{\Lambda}$ be the diagonal matrix with entries $\lambda_1, \ldots, \lambda_n$. Since for each i, $\boldsymbol{Av}_i = \lambda_i \boldsymbol{v}_i$, we have $\boldsymbol{A}\mathbf{Q} = \mathbf{Q}\boldsymbol{\Lambda}$. Since $\mathbf{Q}$ is orthogonal, i.e., $\mathbf{Q}\mathbf{Q}^T = I$, we get that $\mathbf{Q}^T = \mathbf{Q}^{-1}$. Therefore,

$$\boldsymbol{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^T.$$

Let $\boldsymbol{D}$ be the diagonal matrix whose diagonal entries are the positive square roots of $\lambda_1, \ldots, \lambda_n$ (by (2), $\lambda_1, \ldots, \lambda_n$ are nonnegative, and thus their square roots are real). Then, $\boldsymbol{\Lambda} = \boldsymbol{D}\boldsymbol{D}^T$. Substituting, we get

$$\boldsymbol{A} = \mathbf{Q}\boldsymbol{D}\boldsymbol{D}^T\mathbf{Q}^T = (\mathbf{Q}\boldsymbol{D})(\mathbf{Q}\boldsymbol{D})^T.$$

Now, (3) follows by letting $\boldsymbol{W} = (\mathbf{Q}\boldsymbol{D})^T$.

$(3 \Rightarrow 1)$: For any

$$\boldsymbol{x} \in \mathbb{R}^n, \ \boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{W}^T \boldsymbol{W} \boldsymbol{x} = (\boldsymbol{W}\boldsymbol{x})^T (\boldsymbol{W}\boldsymbol{x}) \geq 0. \qquad \square$$

Properties of positive semidefinite matrices (3)

- With Cholesky decomposition a real symmetric matrix can be decomposed in polynomial time as $\boldsymbol{A} = \boldsymbol{U\Lambda U^T}$, where $\boldsymbol{\Lambda}$ is a diagonal matrix whose diagonal entries are the eigenvalues of $\boldsymbol{A}$
- $\boldsymbol{A}$ is positive semidefinite if all the entries of $\boldsymbol{\Lambda}$ are nonnegative giving a polynomial time test for positive semidefiniteness
- The decomposition $\boldsymbol{WW^T}$ is not polynomial time computable, but can be approcimated
- The sum of two positive semidefinite matrices is also positive semidefinite
- Convex combination is also positive semidefinite

The semidefinite programming problem (1)

- Let $\boldsymbol{Y}$ be an $n \ \ n$ matrix of real valued variables with y_{ij} entry
- The *semidefinite programming problem* is the problem of maximizing a linear function of the y_{ij} ’s, subject to linear constraints on them, and the additional constraint that $\boldsymbol{Y}$ be symmetric and positive semidefinite

The semidefinite programming problem (2)

- Denote by $\mathbf{R}^{n \quad n}$ the space of $n \quad n$ real matrices
- The trace of a matrix $\boldsymbol{A}$ is the sum of its diagonal entries and is denoted by tr($\boldsymbol{A}$)
- The Frobenius inner product of matrices $\boldsymbol{A}$, $\boldsymbol{B}$ is defined to be

$$\boldsymbol{A} \bullet \boldsymbol{B} = \mathrm{tr}(\boldsymbol{A}^T \boldsymbol{B}) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{ij}$$

- M_n denotes the cone of symmetric $n \quad n$ real matrices
- For $\boldsymbol{A} \in \boldsymbol{M}_n, \boldsymbol{A} \succeq 0$ denotes that $\boldsymbol{A}$ is pos. sem.

The semidefinite programming problem (3)

- The general semidefinite programming problem, S:

$$\begin{aligned} &\text{maximize} && \boldsymbol{C} \bullet \boldsymbol{Y} \\ &\text{subject to} && \boldsymbol{D}_i \bullet \boldsymbol{Y} = d_i, \quad 1 \le i \le k \\ & && \boldsymbol{Y} \succeq 0, \\ & && \boldsymbol{Y} \in M_n \end{aligned}$$

The semidefinite programming problem (4)

- A matrix satisfying all the constraints is a feasible solution and so is any convex combination of these solutions
- Let $\boldsymbol{A}$ be an infeasible point. Let $\boldsymbol{C}$ be another point. A hyperplane $\boldsymbol{C} \bullet \boldsymbol{Y} \leq b$ is called a seperating hyperplane for $\boldsymbol{A}$ if all feasible points satisfy it and point $\boldsymbol{A}$ does not
- **Theorem 26.4** *Let* S *be a semidefinite programming problem, and* **A** *be a point in* $\mathbf{R}^{n \times n}$*. We can determine in polynomial time, whether* **A** *is feasible for* S *and, if it is not, find a separating hyperplane.*

The semidefinite programming problem (4)

- **Proof:** Testing for feasibility involves ensuring that $\boldsymbol{A}$ is symmetric and positive semidefinite and satisfies all the constraints. This can be done in polynomial time. If $\boldsymbol{A}$ is infeasible, a separating hyperplane is obtained as follows.
  - If $\boldsymbol{A}$ is not symmetric, $\alpha_{ij} > \alpha_{ji}$ for some i,j. Then $y_{ij} \leq y_{ji}$ is a separating hyperplane
  - If $\boldsymbol{A}$ is not positive semidefinite, then it has a negative eigenvalue, say λ, and υ the corresponding eigenvector. Then, $\left(\upsilon\upsilon^T\right)\bullet\boldsymbol{Y} = \upsilon^T\boldsymbol{Y}\upsilon \geq 0$ is a separating hyperplane.
  - If any of the linear constraints is violated, it directly yields a separating hyperplane

The semidefinite programming problem (5)

- Let V be a vector program on n n-dimensional vector variables $v_1, v_2, \ldots, v_n$.
- Define the corresponding semidefinite program, S, over n^2 variables y_{ij}, for $1 \leq i,j \leq n$ as follows:
  - Replace each inner product $v_i \cdot v_j$ by the variable y_{ij}.
  - Require that matrix $\boldsymbol{Y}$ is symmetric and positive semidefinite
- **Lemma 26.5** *Vector program* V *is equivalent to semidefinite program* S

The semidefinite programming problem (6)

- **Proof:** One must show that corresponding to each feasible solution to V, there is a feasible solution to S of the same objective function value and vice versa
  - Let $\alpha_1 ,\ldots, \alpha_n$ be a feasible solution to V. Let $\boldsymbol{W}$ be the matrix whose columns are $\alpha_1 ,\ldots, \alpha_n$

    Then it is easy to see that $\boldsymbol{A}=\boldsymbol{W}^T \boldsymbol{W}$ is a feasible solution to S having the same objective function value
  - Let $\boldsymbol{A}$ be a feasible solution to S. By theorem 26.3 there is a n n matrix $\boldsymbol{W}$ such that $\boldsymbol{A}=\boldsymbol{W}^T \boldsymbol{W}$. Let $\alpha_1 ,\ldots, \alpha_n$ be the columns of $\boldsymbol{W}$. Then it is easy to see that $\alpha_1 ,\ldots, \alpha_n$ is a feasible solution to V having the same objective function value

The semidefinite programming problem (7)

- The semidefinite programming relaxation to MAX-CUT is:

$$\begin{aligned}
&\text{maximize} && \frac{1}{2}\sum_{1\le i<j\le n} w_{ij}\left(1-y_i y_j\right) && \\
&\text{subject to} && y_i^2 = 1, && \upsilon_i \in V \\
& && \boldsymbol{Y} \succeq 0, && \\
& && \boldsymbol{Y} \in M_n &&
\end{aligned}$$

Randomized rounding algorithm (1)

- Assume we have an optimal solution to the vector program
- Let $\alpha_1 ,\ldots, \alpha_n$ be an optimal solution and let OPT_v denote its objective function value
- These vectors lie on the n-dimensional unit sphere S_{n-1}
- We need a cut $(S,\bar{S})$ whose weight is a large fraction of OPT_v

Randomized rounding algorithm (2)

- Let θ_{ij} denote the angle between vectors α_i and α_j . The contribution of this pair of vectors to OPT_{υ} is,

$$\frac{w_{ij}}{2}(1-\cos\theta_{ij})$$

- The closer θ_{ij} is to π, the larger the contribution will be
- Pick $\boldsymbol{r}$ to be a uniformly distributed vector on the unit sphere and let $S=\{\upsilon_i \mid a_i \cdot \boldsymbol{r} \geq 0\}$

Randomized rounding algorithm (3)

- **Lemma 26.6**
  - **Pr**[v_i and v_j are separated]= θ_{ij}/π
- **Proof:**
  - Project $\boldsymbol{r}$ onto the plane containing α_i and α_j
  - Now, vertices v_i and v_j will be separated iff the projection lies in one of the two arcs of angle θ_{ij} . Since $\boldsymbol{r}$ has been picked from a spherically symmetric distribution, its projection will be a random direction in the plane. The lemma follows.

Randomized rounding algorithm (4)

- **Lemma 26.7** *Let $x_1, \ldots, x_n$ be picked independently from the normal distribution with mean 0 and unit standard deviation. Let* $d = \sqrt{x_1^2 + \ldots + x_n^2}$.
  Then, $\left(x_1/d, \ldots, x_n/d\right)$ *is a random vector on the unit sphere.*

- **Algorithm 26.8 (MAX-CUT)**

  1. Solve vector program V. Let $\alpha_1, \ldots, \alpha_n$ be an optimal solution.

  2. Pick $\boldsymbol{r}$ to be a uniformly distributed vector on the unit sphere.

  3. Let $S = \{\upsilon_i \mid a_i \cdot \boldsymbol{r} \geq 0\}$

Randomized rounding algorithm (5)

- **Lemma 26.9** $\mathbf{E}[W] \geq \alpha * \mathrm{OPT}_{\upsilon}$

- **Corollary 26.10** *The integrality gap for vector relaxation is at least a*>0.87856

- **Theorem 26.11** *There is a randomized approximation algorithm for MAX-CUT achieving an approximation factor of* 0.87856

Improving the guarantee for MAX-2SAT (1)

- MAX-2SAT is the restriction of MAX-SAT to formulae in which each clause contains at most two literals
- Already obtained a ¾ algorithm for that
- Semidefinite programming gives an improved algorithm
- Idea: convert the obvious quadratic program into a strict quadratic program

Improving the guarantee for MAX-2SAT (2)

- To each Boolean variable x_i introduce variable y_i which is constrained to be either +1 or -1, for $1 \le i \le n$. In addition introduce another variable y_0 which is also constrained to be either +1 or -1.
- x_i is true if $y_i = y_0$ and false otherwise
- The value $\upsilon(C)$ of clause C is defined to be 1 if C is satisfied and 0 otherwise
- For clauses containing only one literal

$$\upsilon(x_i) = \frac{1 + y_0 y_i}{2} \text{ and } \upsilon(\overline{x}_i) = \frac{1 - y_0 y_i}{2}$$

Improving the guarantee for MAX-2SAT (3)

- For a clause with 2 literals

$$\upsilon(x_i \vee x_j) = 1 - \upsilon(\overline{x}_i)\upsilon(\overline{x}_j) = 1 - \frac{1 - y_0 y_i}{2} \frac{1 - y_0 y_j}{2}$$

$$= \frac{1}{4}\left(3 + y_0 y_i + y_0 y_j - y_0^2 y_i y_j\right)$$

$$= \frac{1 + y_0 y_i}{4} + \frac{1 + y_0 y_j}{4} + \frac{1 - y_i y_j}{4}$$

Improving the guarantee for MAX-2SAT (4)

- MAX-2SAT can be written as a strict strict quadratic program

$$\text{maximize} \quad \sum_{0 \le i < j \le n} \alpha_{ij}\left(1 + y_i y_j\right) + b_{ij}\left(1 - y_i y_j\right)$$

$$\text{subject to} \quad y_i^2 = 1, \qquad 0 \le i \le n$$

$$y_i \in \mathbf{Z}, \qquad 0 \le i \le n$$

Improving the guarantee for MAX-2SAT (5)

- Corresponding vector program relaxation where vector variable υ_i corresponds to y_i

$$\text{maximize} \quad \sum_{0 \le i < j \le n} \alpha_{ij}\left(1+\upsilon_i \cdot \upsilon_j\right) + b_{ij}\left(1-\upsilon_i \cdot \upsilon_j\right)$$

$$\text{subject to} \quad \upsilon_i \cdot \upsilon_j = 1, \qquad 0 \le i \le n$$

$$\upsilon_i \in \mathbf{R}^{n+1}, \qquad 0 \le i \le n$$

Improving the guarantee for MAX-2SAT (6)

- The algorithm is similar to that for MAX-CUT
- Let $\alpha_0, \ldots, \alpha_n$ be an optimal solution. Pick a vector $\boldsymbol{r}$ uniformly distributed on the unit sphere in $(n+1)$ dimensions and let $y_i = 1$ iff $r \cdot a_i \geq 0$ for $0 \leq i \leq n$
- This gives a truth assignment for the Boolean variables
- Let W be the random variable denoting the weight of this truth assignment

Improving the guarantee for MAX-2SAT (7)

- **Lemma 26.13** $\mathbf{E}[W] \geq \alpha * \mathrm{OPT}_{\upsilon}$
- **Proof:**

$$\mathbf{E}[W] = 2 \sum_{0 \leq i < j \leq n} a_{ij} \Pr[y_i = y_j] + b_{ij} \Pr[y_i \neq y_j]$$

let θ_{ij} denote the angle between a_i and a_j

$$\Pr[y_i \neq y_j] = \frac{\theta_{ij}}{\pi} \geq \frac{a}{2}\left(1 - \cos\theta_{ij}\right)$$

and

$$\Pr[y_i = y_j] = 1 - \frac{\theta_{ij}}{\pi} \geq \frac{a}{2}\left(1 + \cos\theta_{ij}\right)$$

Therefore,

$$\mathbf{E}[W] \geq a \cdot \sum_{0 \leq i < j \leq n} a_{ij}\left(1 + \cos\theta_{ij}\right) + b_{ij}\left(1 - \cos\theta_{ij}\right) = a \cdot \mathrm{OPT}_{\upsilon}$$

The END