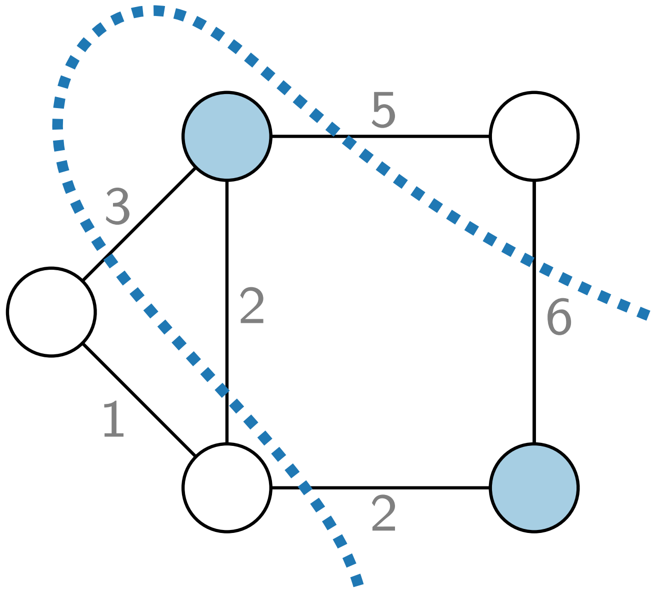
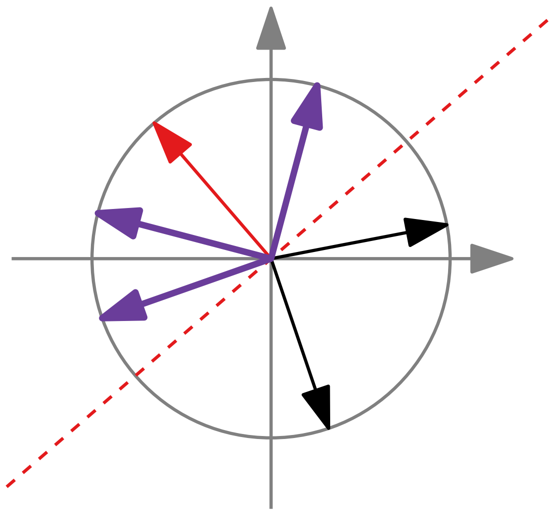
Advanced Algorithms

QP-Relaxation
for MaxCut

Johannes Zink · WS22

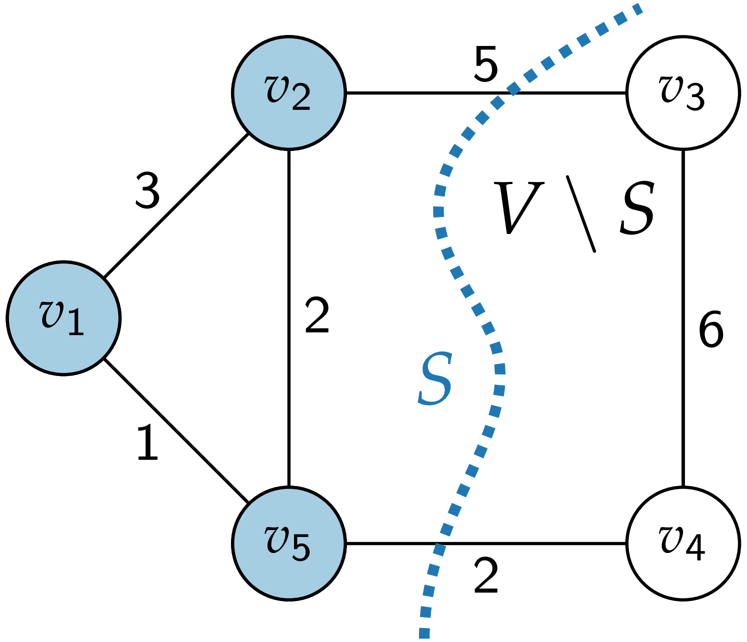




Cut

- Let $G = (V, E)$ be a graph with edge weights $c \colon E \to \mathbb{N}$.
- A **cut** of G is a partition $(S, V \setminus S)$ of V with $\emptyset \neq S \neq V$.
- The **weight** of a cut $(S, V \setminus S)$ is

$$c(S, V \setminus S) = \sum_{\substack{uv \in E, \\ u \in S, v \in V \setminus S}} c(uv)$$



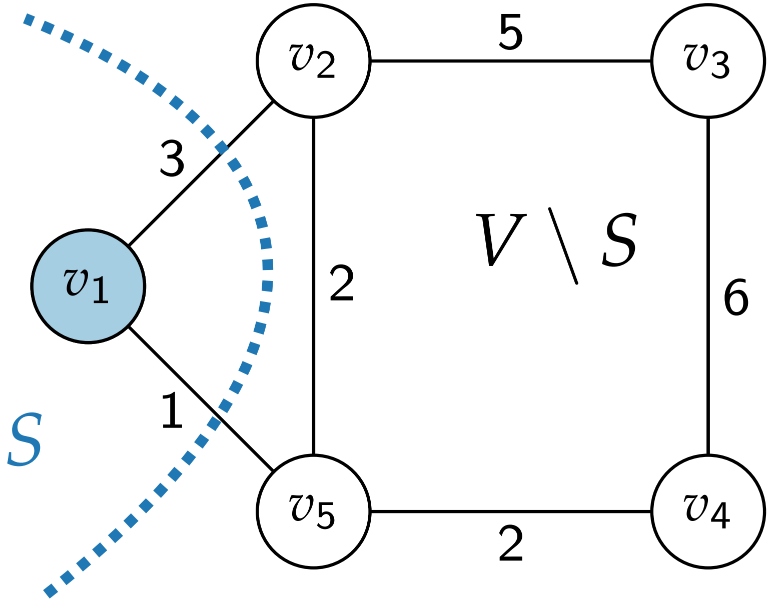
$$c(\{1, 2, 5\}, \{3, 4\}) = 7$$

The **MinCut** Problem

Input. Graph $G = (V, E)$, edge weights $c \colon E \to \mathbb{N}$.

Output. Cut $(S, V \setminus S)$ of G with **minimum** weight.

- Has applications in flow networks (*max-flow min-cut theorem*), finding a bottleneck in a network, graph partition problems, clustering, . . .
- Can be solved optimally in polynomial time, e.g. by the Stoer–Wagner algorithm.



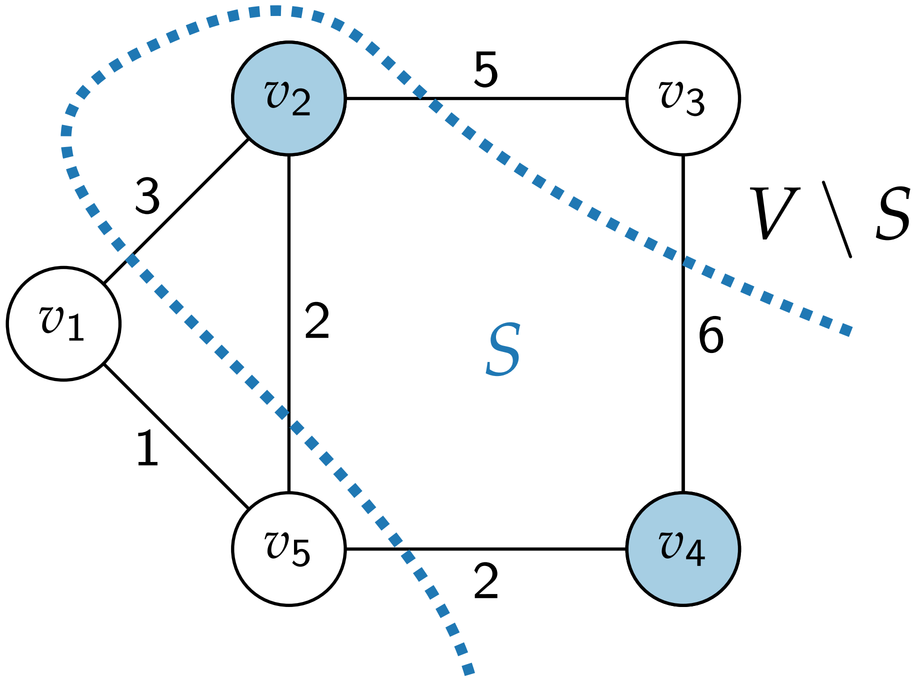
$$c(S, V \setminus S) = 4$$

The **MaxCut** Problem

Input. Graph $G = (V, E)$, edge weights $c \colon E \to \mathbb{N}$.

Output. Cut $(S, V \setminus S)$ of G with **maximum** weight.

- Has applications in statistical physics, where it is used for some models of magnetic spins in disordered systems, and in integrated circuit design for computer chips.
- NP-complete to find a cut with maximum weight.



$c(S, V \setminus S) = 18$

Randomized 0.5-Approximation for (Unweighted) MaxCut

Theorem 1.
CoinFlipMaxCut is a randomized 0.5-approximation algorithm for MaxCut.

CoinFlipMaxCut$(G, c\colon E \to 1)$

$S \leftarrow \emptyset$
foreach $v \in V$ **do**
  if coin flip shows Heads **then**
    $S \leftarrow S \cup \{v\}$
return $c(S, V \setminus S), S$

Proof.

- Runs in $O(n+m)$.
- Compute expected weight of cut:

$$\begin{aligned} \mathsf{E}[c(\textsc{CoinFlipMaxCut}(G))] &= \mathsf{E}\big[|E(S, V \setminus S)|\big] \\ &= \sum_{e \in E} \mathsf{P}[e \in E(S, V \setminus S)] \\ &= \sum_{e \in E} \frac{1}{2} = \frac{1}{2}|E| \geq \frac{1}{2}\mathrm{OPT}(G) \end{aligned}$$

- Can be "derandomized". Exercise.

LP-Relaxation

Integer Linear Program

$$\begin{array}{llcl} \textbf{maximize} & c^\intercal x & & \\ \textbf{subject to} & Ax & \leq & b \\ & x & \geq & 0 \\ & x & \in & \mathbb{Z} \end{array}$$

→ LP-Relaxation →

Linear Program

$$\begin{array}{llcl} \textbf{maximize} & c^\intercal x & & \\ \textbf{subject to} & Ax & \leq & b \\ & x & \geq & 0 \end{array}$$

↓ Solve in polynomial time ↓

Solution for LP

$x^\star$

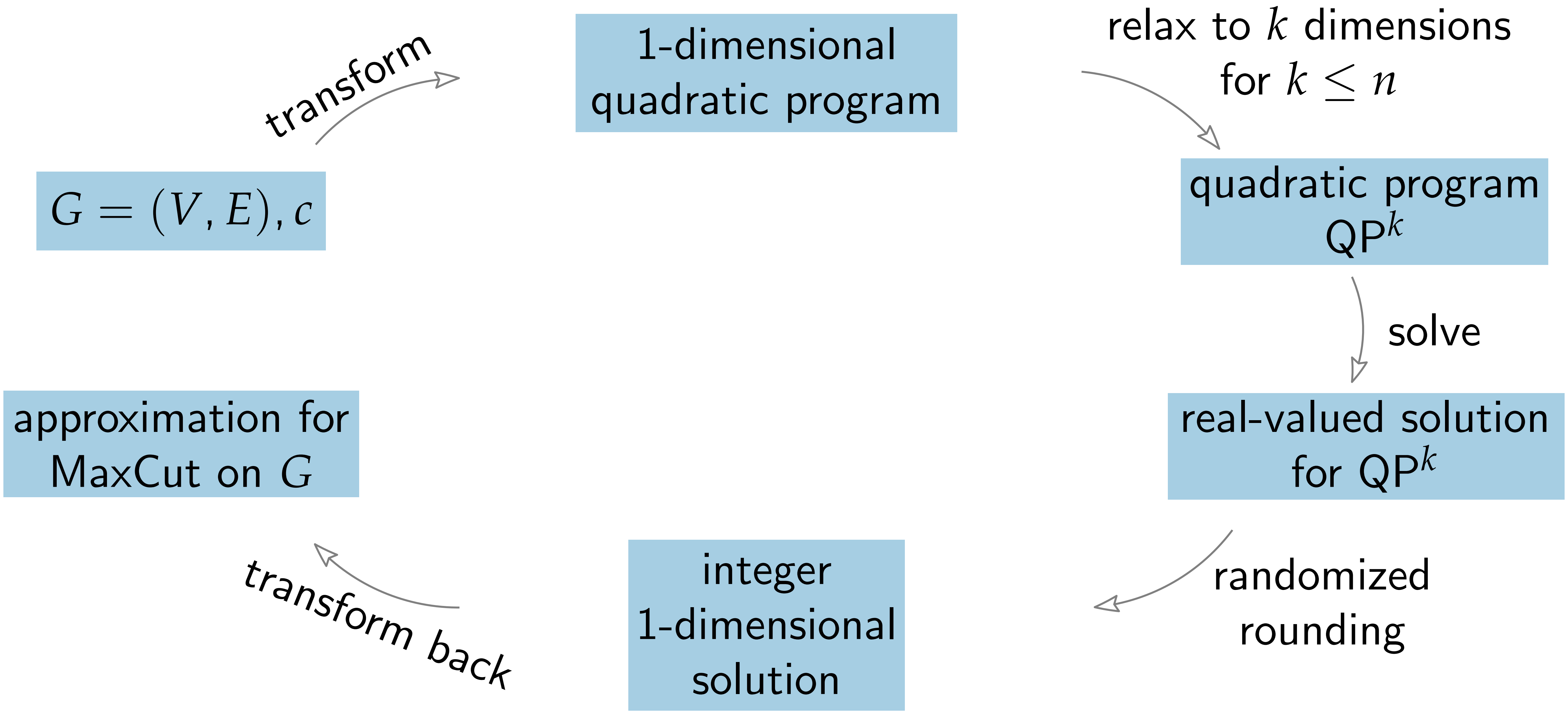
← e.g. rounding ←

Assignment for ILP

$x^\star$

↑ Solution, approximation, or bound ↑

Goemans-Williamson Algorithm for MaxCut



QP(G, c)

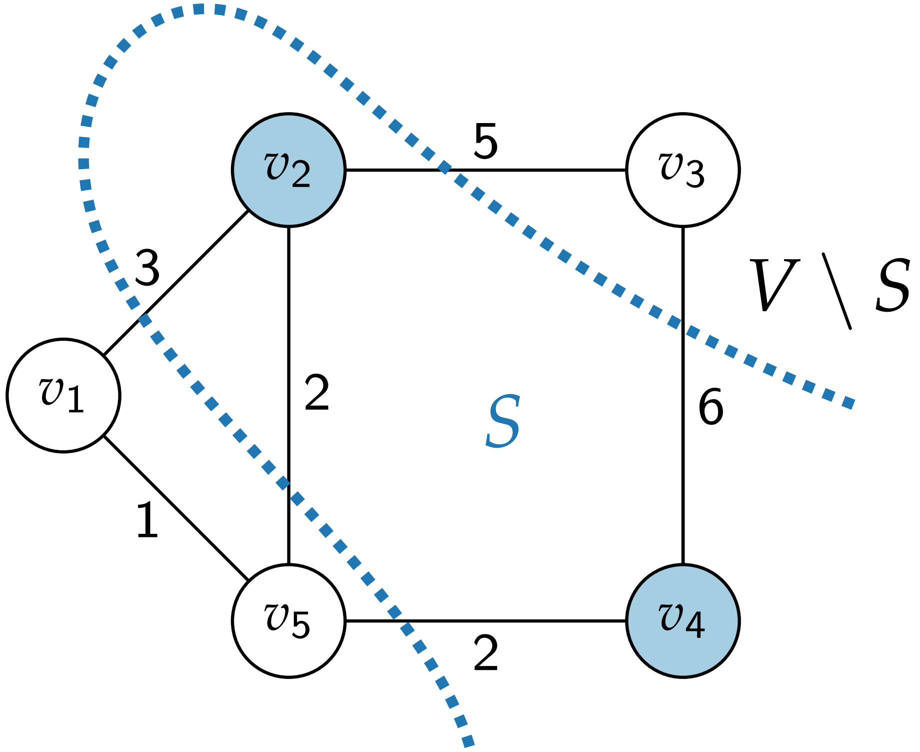
Idea.

- Indicator variable for each vertex v_i:
  $x_i \in \{1, -1\}$
- $x_i \cdot x_j = \begin{cases} 1 & \text{if } i, j \text{ in same partition} \\ -1 & \text{otherwise} \end{cases}$

QP(G, c)

$$\textbf{maximize} \quad \frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij}(1 - x_i x_j)$$

$$\textbf{subject to} \quad x_i^2 = 1$$



- Weight matrix c_{ij}

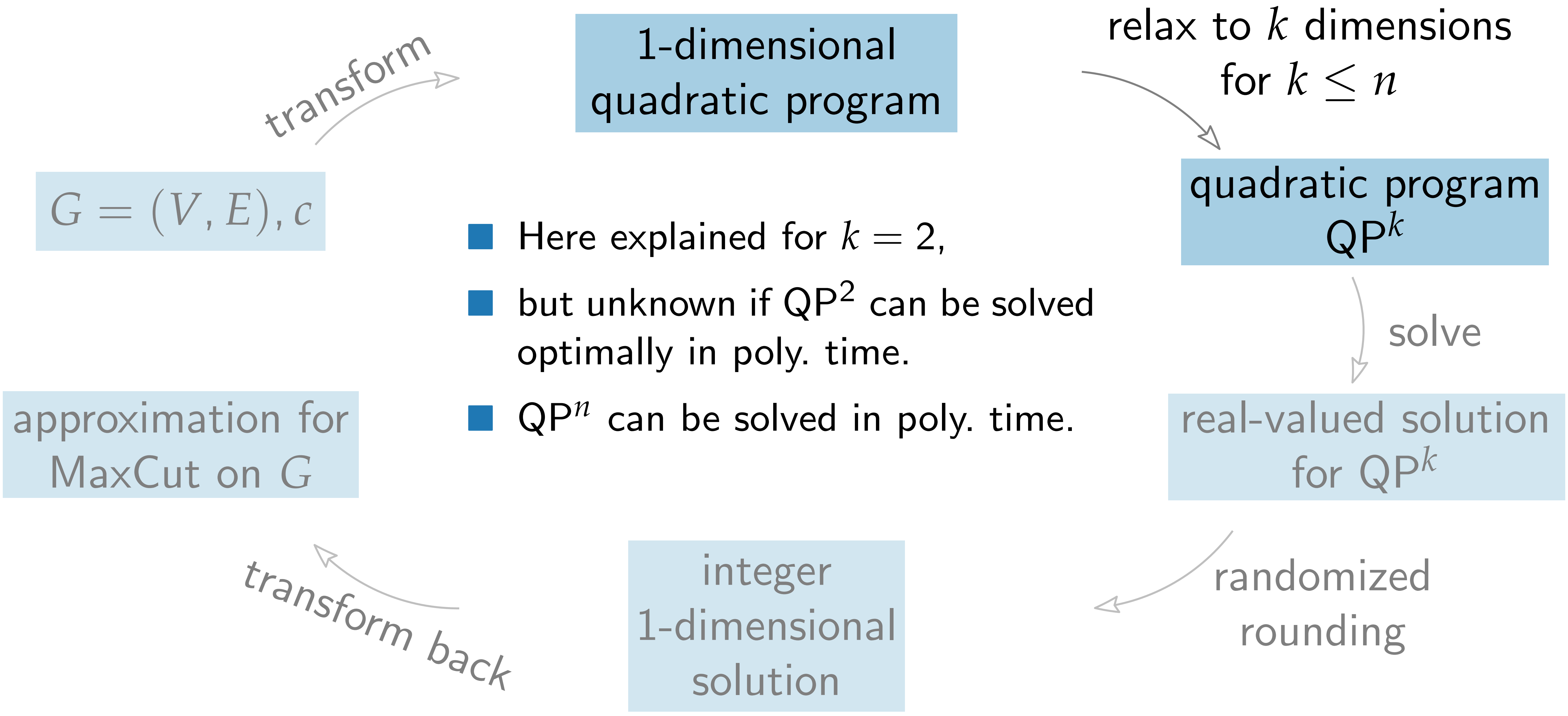
	1	2	3	4	5
1		3			1
2	3		5		2
3		5		6	
4			6		2
5	1	2		2	

- Solution
  $x_2 = x_4 = 1$
  $x_1 = x_3 = x_5 = -1$

Note.

- Solving QP(G, c) is NP-hard.
- Otherwise MaxCut wouldn't be NP-hard.

Goemans-Williamson Algorithm for MaxCut



- Here explained for $k = 2$,
- but unknown if QP^2 can be solved optimally in poly. time.
- QP^n can be solved in poly. time.

Relaxation of QP(G, c)

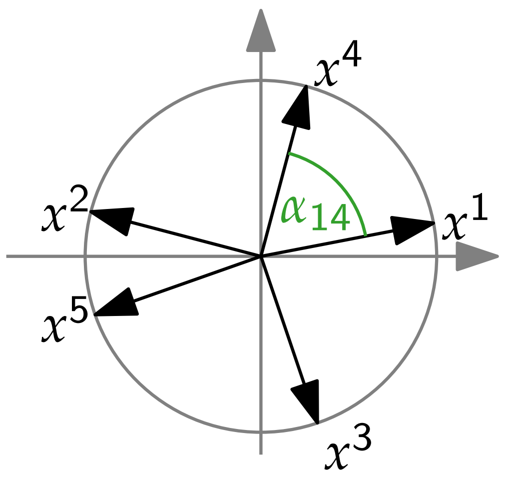
QP$^2(G, c)$

maximize $\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij}(1 - x^i \cdot x^j)$

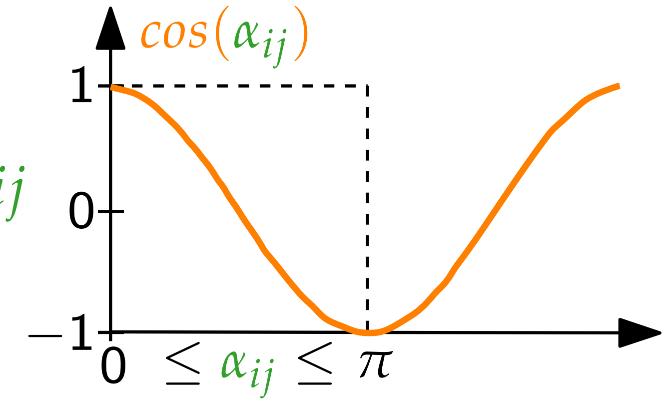
subject to $x^i \cdot x^i = 1$

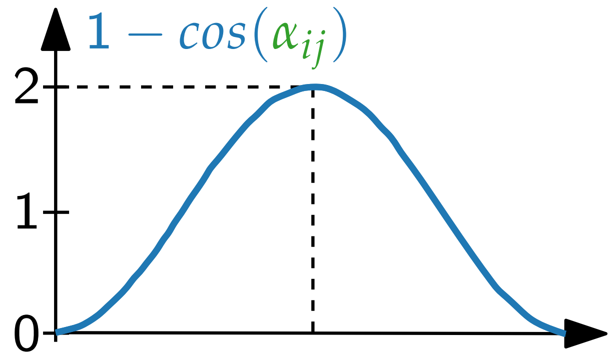
$x^i = (x^i_1, x^i_2) \in \mathbb{R}^2$

- " $\cdot$ " is scalar product.
- x^i lies on the unit circle.
- $x^i \cdot x^j = |x^i||x^j| \cos(\alpha_{ij})$ $= \cos(\alpha_{ij})$ with $0 \le \alpha_{ij} \le \pi$.

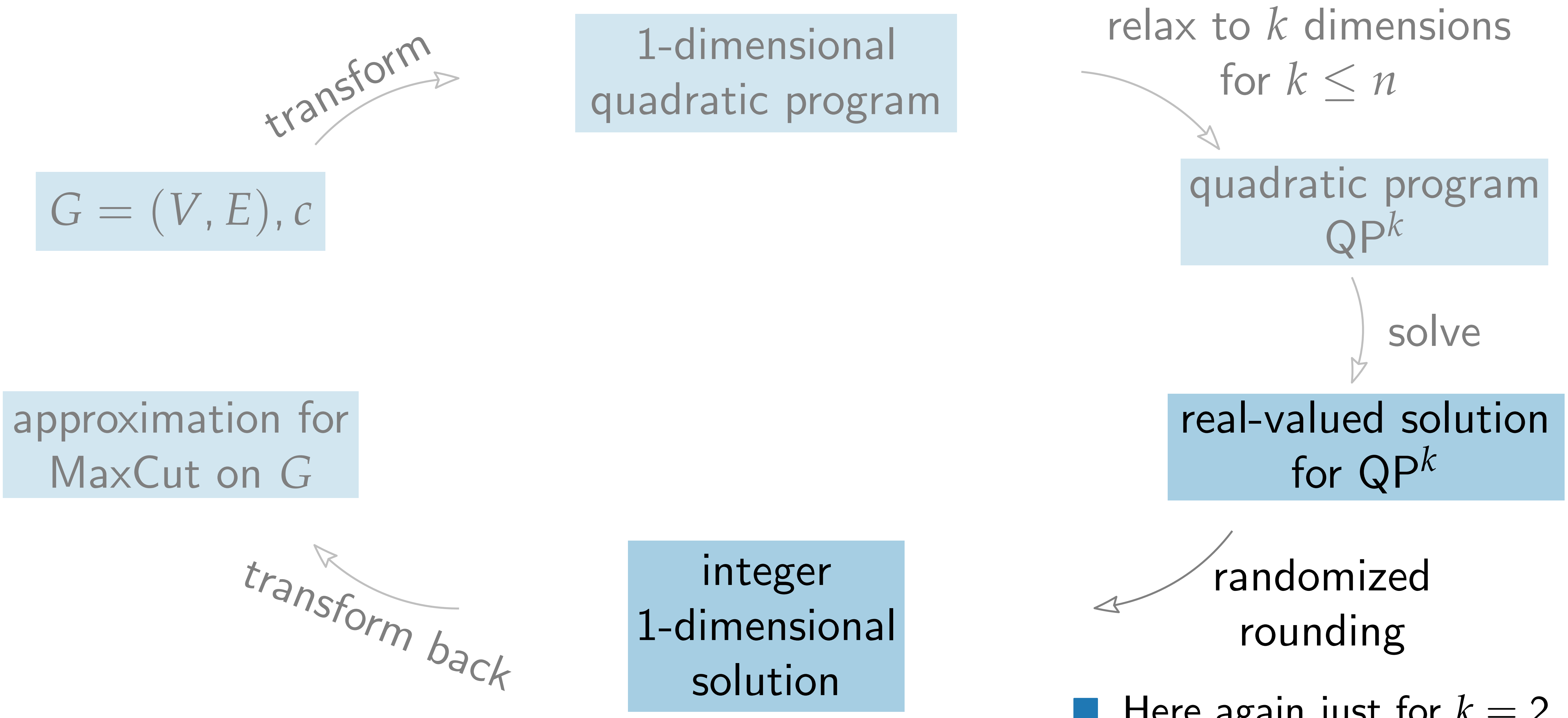


- We maximize angles α_{ij} since larger α_{ij} increase the contribution of c_{ij}.
- Hence, our objective is: $\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij}(1 - \cos(\alpha_{ij}))$





Goemans-Williamson Algorithm for MaxCut



- Here again just for $k = 2$.

Algorithm RandomizedMaxCut

RandomizedMaxCut(G, c)

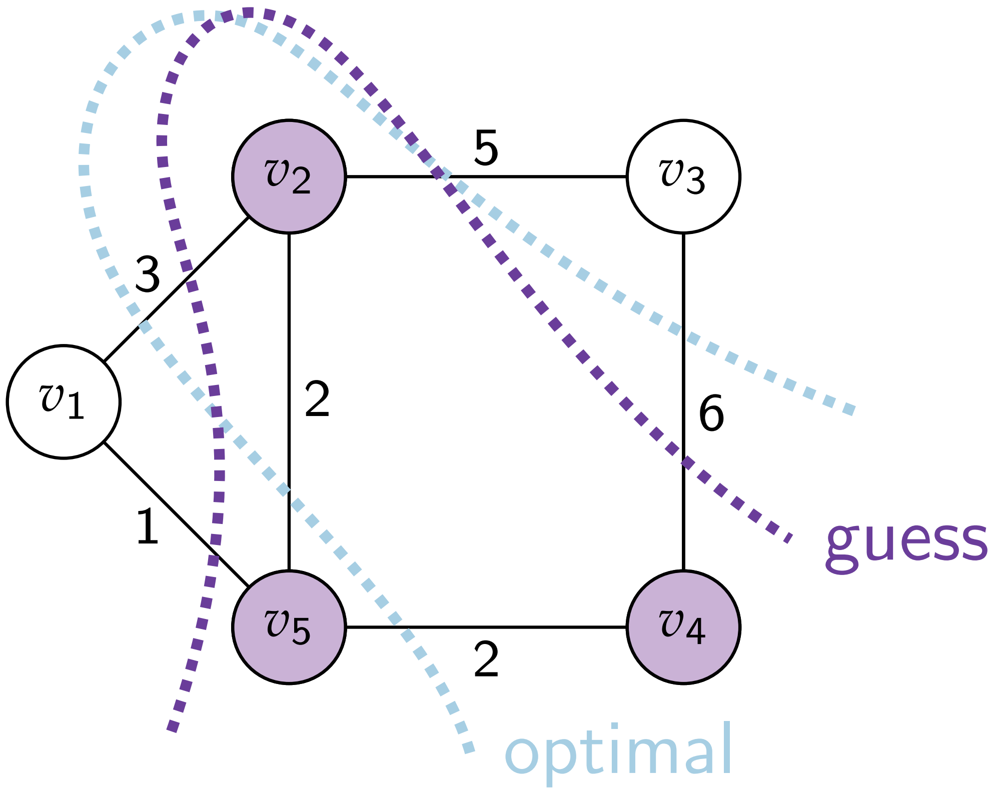
Compute optimal solution $(\tilde{x}^1, \ldots, \tilde{x}^n)$ for QP$^2(G, c)$

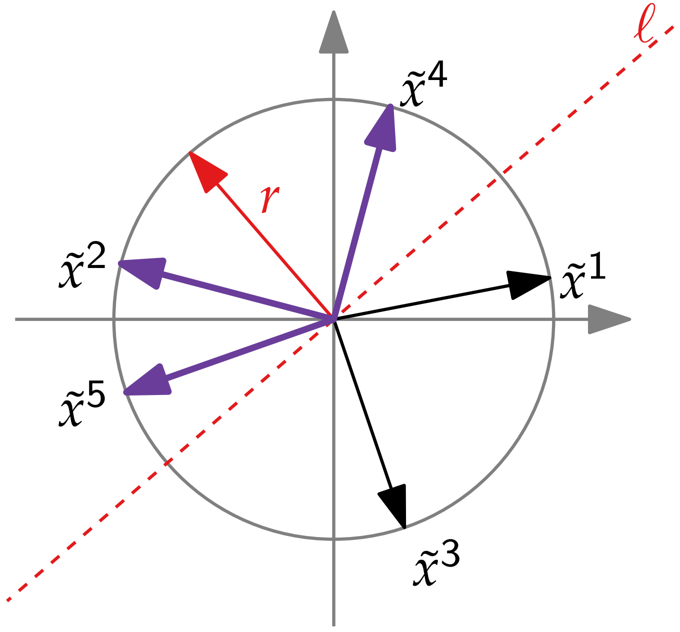
Pick random vector $r \in \mathbb{R}^2$

$S \leftarrow \{v_i \in V : \tilde{x}^i \cdot r \geq 0\}$

return $c(S, V \setminus S)$

- $\tilde{x}^i$ lies above line ℓ orthogonal to r



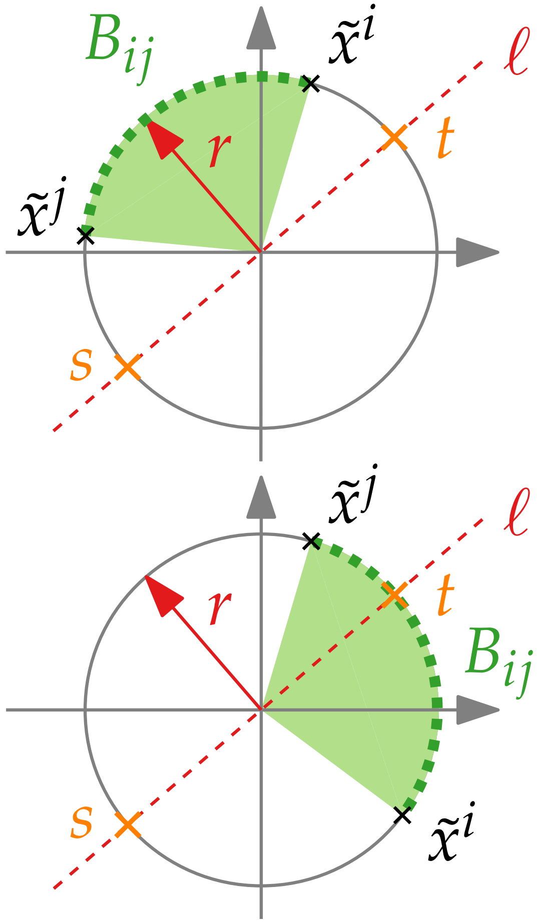


(Sketch)

RandomMaxCut – Expected Value

Lemma 2.
Let X be the solution of RandomizedMaxCut(G, c).
If r is picked uniformally at random, then

$$\mathrm{E}[X] = \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij} \frac{\alpha_{ij}}{\pi}.$$



Proof.

- $E[X] = \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij}\, \mathrm{P}[\ell \text{ separates } \tilde{x}^i, \tilde{x}^j] = \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij} \frac{\alpha_{ij}}{\pi}$
- $\mathrm{P}[\ell \text{ separates } \tilde{x}^i, \tilde{x}^j] = \mathrm{P}[s \text{ or } t \text{ lies on } B_{ij}] = \frac{\alpha_{ij}}{2\pi} + \frac{\alpha_{ij}}{2\pi} = \frac{\alpha_{ij}}{\pi}$
- B_{ij} has length α_{ij}.
- If $\tilde{x}^i$ (or $\tilde{x}^j$) lies $\leq \alpha_{ij}$ before s or t on the perimter of the unit disk, s or t lies on B_{ij}.

RandomMaxCut – Quality

Theorem 3.
Let X be the solution of RandomizedMaxCut(G, c). Then

$$\frac{E[X]}{OPT(G,c)} \geq 0.8785.$$

Proof.

- Lemma 2: $E[X] = \sum_{j=1}^{n}\sum_{i=1}^{j-1} c_{ij}\frac{\alpha_{ij}}{\pi}$
- Optimal solution for QP^2:

$$QP^2(G,c) = \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{j-1} c_{ij}(1 - x^i \cdot x^j) = \sum_{j=1}^{n}\sum_{i=1}^{j-1} c_{ij}\frac{1-\cos(\alpha_{ij})}{2}$$

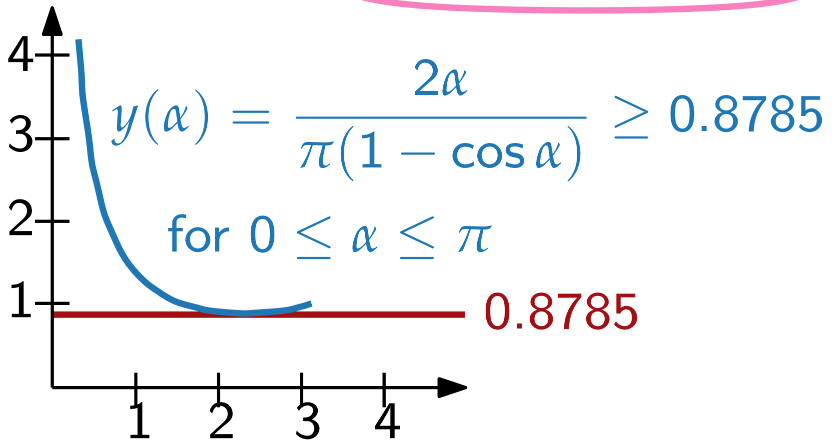
- $QP^2(G,c)$ is relaxation of $QP(G,c)$:

$$QP^2(G,c) \geq QP(G,c) = OPT(G,c)$$

- $$\frac{E[X]}{OPT(G,c)} \geq \frac{E[X]}{QP^2(G,c)} = \frac{\sum_{j=1}^{n}\sum_{i=1}^{j-1} c_{ij}\frac{\alpha_{ij}}{\pi}}{\sum_{j=1}^{n}\sum_{i=1}^{j-1} c_{ij}\frac{1-\cos(\alpha_{ij})}{2}} \geq 0.8785$$

- $$\frac{\frac{\alpha_{ij}}{\pi}}{\frac{1-\cos(\alpha_{ij})}{2}} \geq 0.8785$$

$$\Leftrightarrow \frac{\alpha_{ij}}{\pi} \geq 0.8785\frac{1-\cos(\alpha_{ij})}{2}$$



Example

1. Step: Build QP

$$\begin{aligned} \textbf{maximize} \quad & \frac{1}{2}\sum_{j=1}^{6}\sum_{i=1}^{j-1} c_{ij}(1 - x_i x_j) \\ \textbf{subject to} \quad & x_i^2 = 1 \end{aligned}$$

Weight matrix c_{ij}

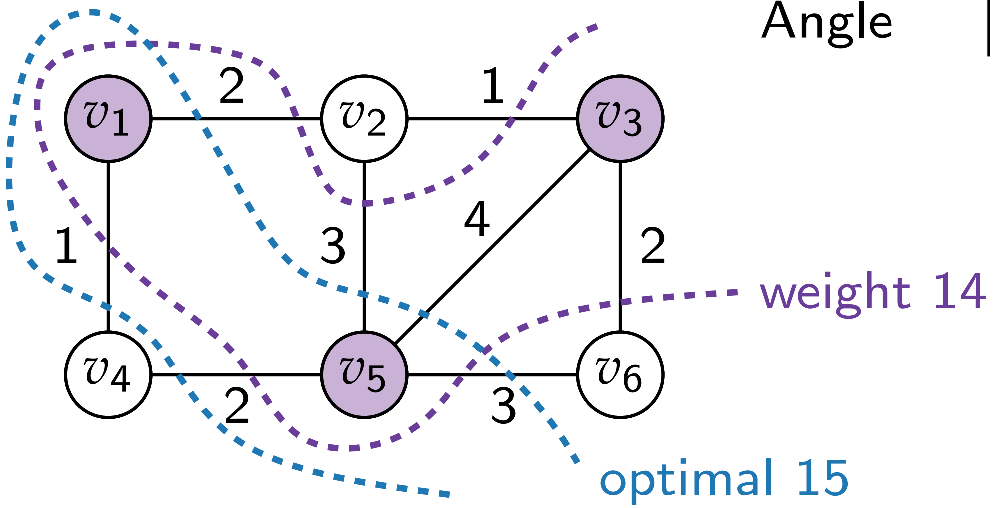
	1	2	3	4	5	6
1		2		1		
2	2		1		3	
3		1			4	2
4	1				2	
5		3	4	2		3
6			2		3	

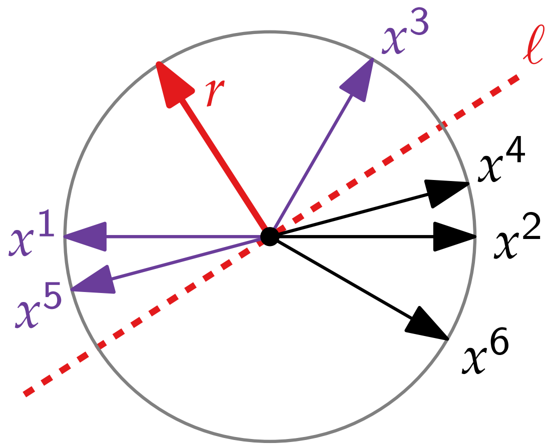
2. Step: Relax QP to QP^2

$$\begin{aligned} \textbf{maximize} \quad & \frac{1}{2}\sum_{j=1}^{6}\sum_{i=1}^{j-1} c_{ij}(1 - x^i \cdot x^j) \\ \textbf{subject to} \quad & x^i \cdot x^i = 1 \\ & x^i = (x_1^i, x_2^i) \in \mathbb{R}^2 \end{aligned}$$

3. Step: Solve QP^2

Variable	x^1	x^2	x^3	x^4	x^5	x^6
Angle	0	180	120	165	345	210





4. Step: Guess r

5. Step: Derive S

Goemans-Williamson Algorithm for MaxCut

$G = (V, E), c$

transform

1-dimensional quadratic program

relax to k dimensions for $k \leq n$

quadratic program QP^k

solve

real-valued solution for QP^k

randomized rounding

integer 1-dimensional solution

transform back

approximation for MaxCut on G

- So far, $k = 2$.
- QP^n can be solved in polynomial time.

$\mathrm{QP}^n(G, c)$

$\mathbf{QP}^2$ (G, c)

maximize $\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij}(1 - x^i \cdot x^j)$

subject to
$$x^i \cdot x^i = 1$$
$$x^i = (x^i_1, x^i_2) \in \mathbb{R}^2$$

$\mathbf{QP}^n$ (G, c)

maximize $\frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{j-1} c_{ij}(1 - x^i \cdot x^j)$

subject to
$$x^i \cdot x^i = 1$$
$$x^i \in \mathbb{R}^n$$

- A matrix M is called **positive semidefinite** if for any vector $v \in \mathbb{R}^n$:
$$v^\intercal \cdot M \cdot v \geq 0$$
- $M = (m_{ij}) = (x^i \cdot x^j)$ is positive semidefinite.
- $\mathrm{QP}^n(G, c)$ becomes problem $\textsc{SemiDefiniteCut}(G, c)$.
  - Can be approximated in time polynomial in (G, c) and $1/\varepsilon$ with additive guarantee ε.

Discussion

- If the *Unique Games Conjecture* is true, then the approximation ratio of ≈ 0.8785 achieved by SemiDefiniteCut (and RandomizedMaxCut) is best possible.
- Otherwise, no approximation ratio better than $\frac{16}{17} \approx 0.941$ is possible.
  In particular no polynomial-time approximation scheme (PTAS) exists.
- On planar graphs, the MaxCut problem can be solved optimally in polynomial time.

- Semidefinite programming is a powerful tool to develop approximation algorithms
- Whole book on this topic:
  - [Gärtner, Matoušek] "Approximation Algorithms and Semidefinite Progamming"

- Using randomness is another tool to design approximation algorithms.
- See future lectures.

Literature

Original paper:

- [GW '95] "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming"

Source:

- [Vazirani Ch26] "Approximation Algorithms"

Whole book on this topic:

- [Gärtner, Matoušek] "Approximation Algorithms and Semidefinite Progamming"