Maximum-Weight Matching Problem

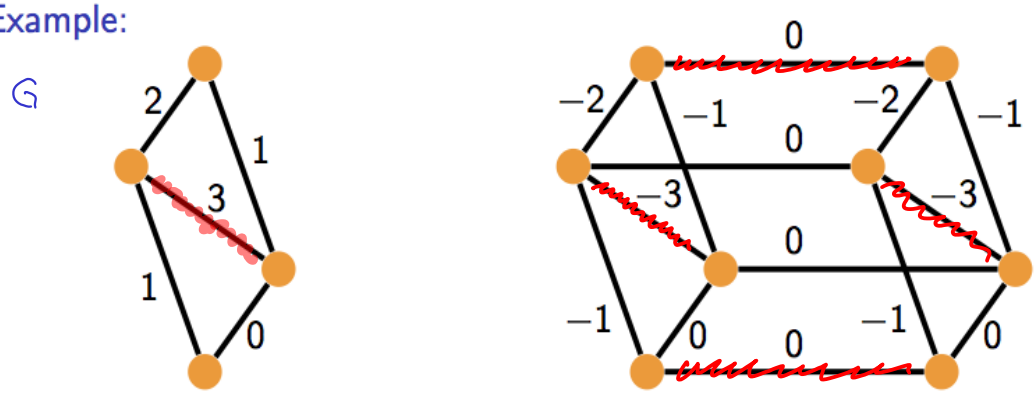
Given: A graph $G = (V, E)$ and $c \in \mathbb{R}^E$.

Task: Find a matching M of G such that $c(M)$ is maximum.

Observation: the maximum-weight matching problem can be reduced to the minimum-weight perfect matching problem as follows:

- multiply all edge weights by -1, i. e., replace c with $-c$;
- make a copy G^* of G; give all edges of G^* the same weights as in G;
- join each node in G to its copy in G^* by an edge of weight 0.

Example:



matchings in both copies must be max weight in G
add 0-edges for isolated vertices

LP Formulation for Maximum-Weight Matching Problem

$$\begin{aligned} \max \quad & \sum_{e \in E} c_e \cdot x_e \\ \text{s.t.} \quad & x(\delta(v)) \leq 1 && \text{for all } v \in V \\ & x(\gamma(S)) \leq (|S| - 1)/2 && \text{for all } S \subseteq V,\ |S| \geq 3,\ |S| \text{ odd} \\ & x \geq 0 \end{aligned}$$

Theorem 13.29.

The maximum weight of a matching of G is equal to the optimal value of the LP above. □

Remarks:

- Theorem can be proved by applying the Matching Polytope Theorem to the graph constructed on the last slide.
- Alternatively, it can be proved directly by a modified version of the Blossom Algorithm. (See book of Cook et al.)

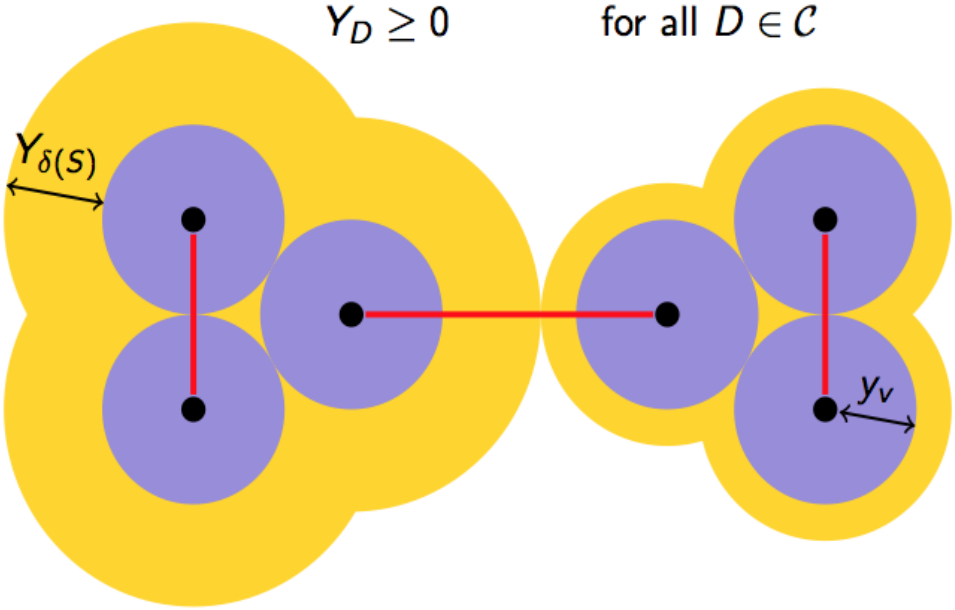
Geometric Interpretation of the Dual LP

Dual perfect matching LP:

$$\max \sum_{v \in V} y_v + \sum_{D \in \mathcal{C}} Y_D$$

$$\text{s.t. } y_u + y_v + \sum_{\substack{D \in \mathcal{C}: \\ e \in D}} Y_D \leq c_e \qquad \text{for all } e = \{u, v\} \in E$$

$$Y_D \geq 0 \qquad \text{for all } D \in \mathcal{C}$$



Geometric Interpretation of the Dual LP (cont.)

- Consider complete graph with even number of nodes corresponding to points in plane and edge weights given by Euclidean distances.
- By Theorems 13.24 and 13.28, there is a perfect matching and a nested dual solution $(y, Y) \geq 0$ satisfying complementary slackness.
- For all nodes $v \in V$, construct (non-overlapping) circular disks D_v centered at v, having radius $y_v \geq 0$. We call D_v the control zone of v.

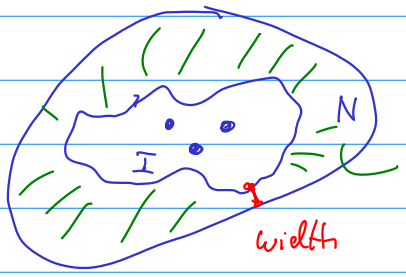
Definition 13.30 (Moat).

A moat $N \setminus \text{interior}(I)$ is given by a pair of compact sets (N, I) with $I \subseteq N$, $|I \cap V|$ odd, and $V \cap \big(N \setminus \text{interior}(I)\big) = \emptyset$. The width of a moat is

$$\min_{x \in I, y \notin \text{interior}(N)} \|x - y\|_2 \ .$$

- Since (y, Y) is feasible, there is room between each pair of nodes $\{u, v\}$ to pack control zones of radius y_u and y_v plus moats of width $Y_{\delta(S)}$ surrounding each odd set $S \subseteq V$ separating u and v.

Definition of a moat in general



$N \setminus$ interior(I) , the moat

How to Construct Moats

- Consider odd subsets $S \subset V$ in order of increasing size.
- For each $v \in S$, construct a disk B_v centered at v of radius

$$\rho_v := y_v + \sum_{v \in T \subsetneq S} Y_{\delta(T)}$$

total value of all $Y_{\delta(T)}$ for $T \subsetneq S$

and a disk B'_v of radius $\rho_v + Y_{\delta(S)}$.

↖ add $Y_{\delta(S)}$ to ρ_v

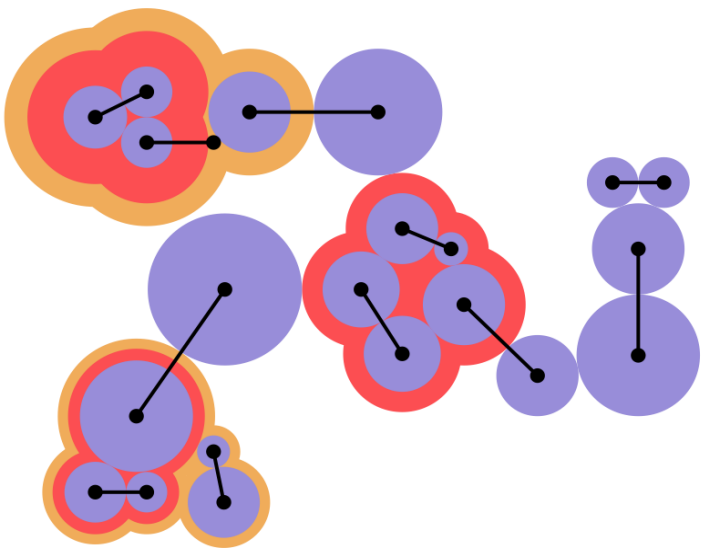
- The moat corresponding to S is given by

$$N := \bigcup_{v \in S} B'_v \quad \text{and} \quad I := \bigcup_{v \in S} B_v \ .$$

- By the properties of (y, Y) and by construction, the moats and control zones do not overlap.

are nested

Example



Goemans-Williamson Algorithm for Minimum Weight Perfect Matchings

Given: Complete graph $G = (V, E)$ with $|V|$ even and edge weights $c \in \mathbb{R}^E_{\geq 0}$ that satisfy the triangle inequality.

Notation:

- An even spanning forest of G is a spanning forest where all trees have an even number of nodes.
- An edge e of an even spanning forest is called even if removing e results in another even forest. Otherwise, e is called odd.

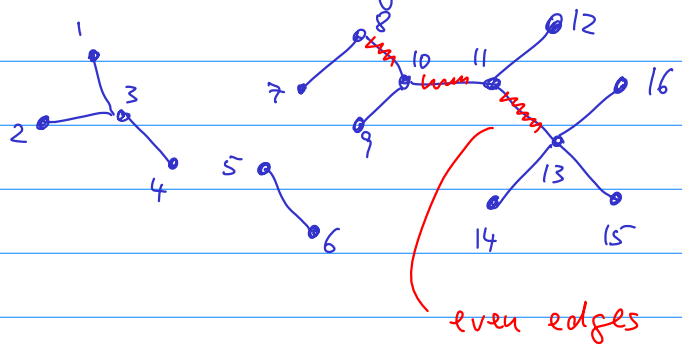
The algorithm of Goemans and Williamson computes an even spanning forest instead of a perfect matching.

Lemma 13.31.

Given an even spanning forest of G, one can construct a perfect matching whose weight is no greater than the forest's weight.

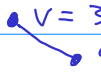
Proof:. . . □

1) Delete all even edges

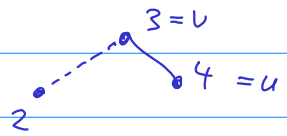


2) Consider one of the remaining trees T with ≥ 3 nodes
delete all leaves in $T \rightarrow T^*$

3) Consider leaf v in T^*
$\Rightarrow$ $\exists$ leaf $u \in T$ that was deleted in the construction of T^*.



Claim: $\exists$ other leaf $w \in T$ adjacent to v



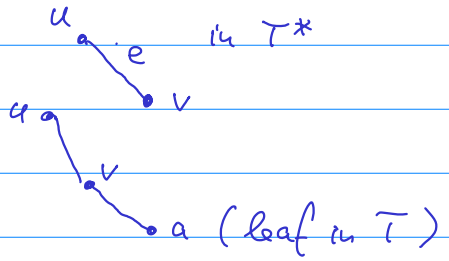
↳ 2 cases

(i) v is isolated vertex in T^*

$\Rightarrow$ all neighbors in T are leaves in T

and ≥ 2 neighbors

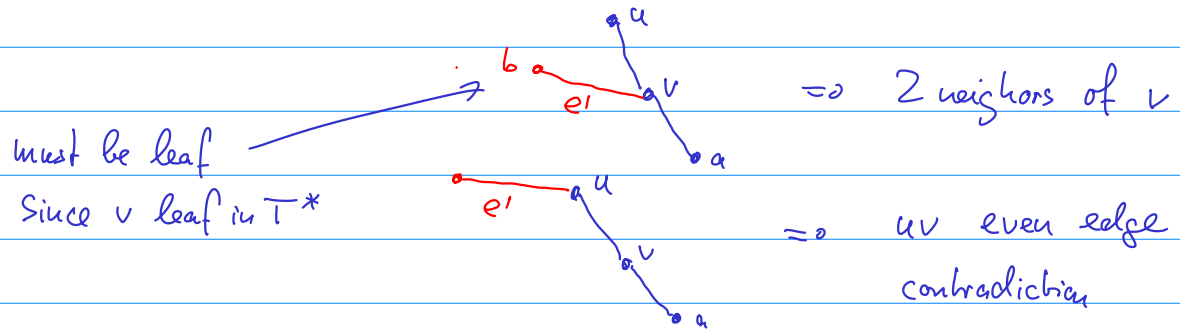
(ii) v endpoint of an edge $e = uv$ in T^*



$\Rightarrow v$ in T

$|T|$ even $\Rightarrow$ T has more vertices

$\Rightarrow$ $\exists$ edge e' to another vertex

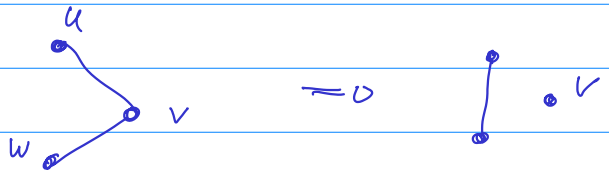


$\Rightarrow$ 2 neighbors of v

$\Rightarrow$ uv even edge

contradiction

4) replace



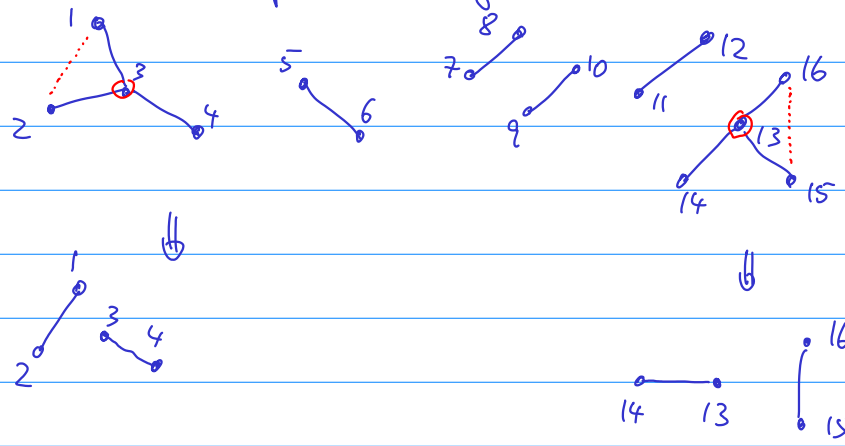
replace this constellation in T by matching edge + v

Δ-inequality $\Rightarrow$ weight gets smaller!

5) iterate this process with remaining trees (the modified T and all other trees)

keep isolated edges

T after deletion of even edges



Goemans-Williamson Algorithm (cont.)

Idea: At each stage, the algorithm has a forest F (not even until the end) and a dual feasible solution $(y, Y) \geq 0$.

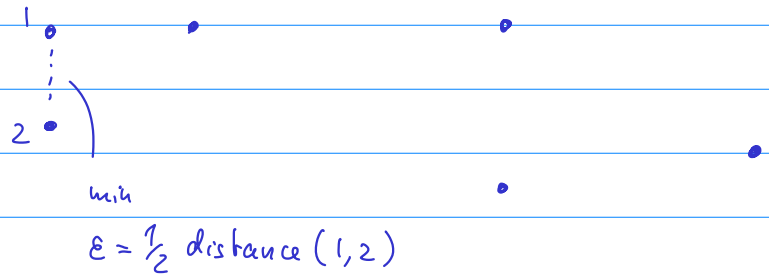
Notation: Let $\mathcal{T}$ denote the node sets of the trees in F. For each set $T \in \mathcal{T}$, let

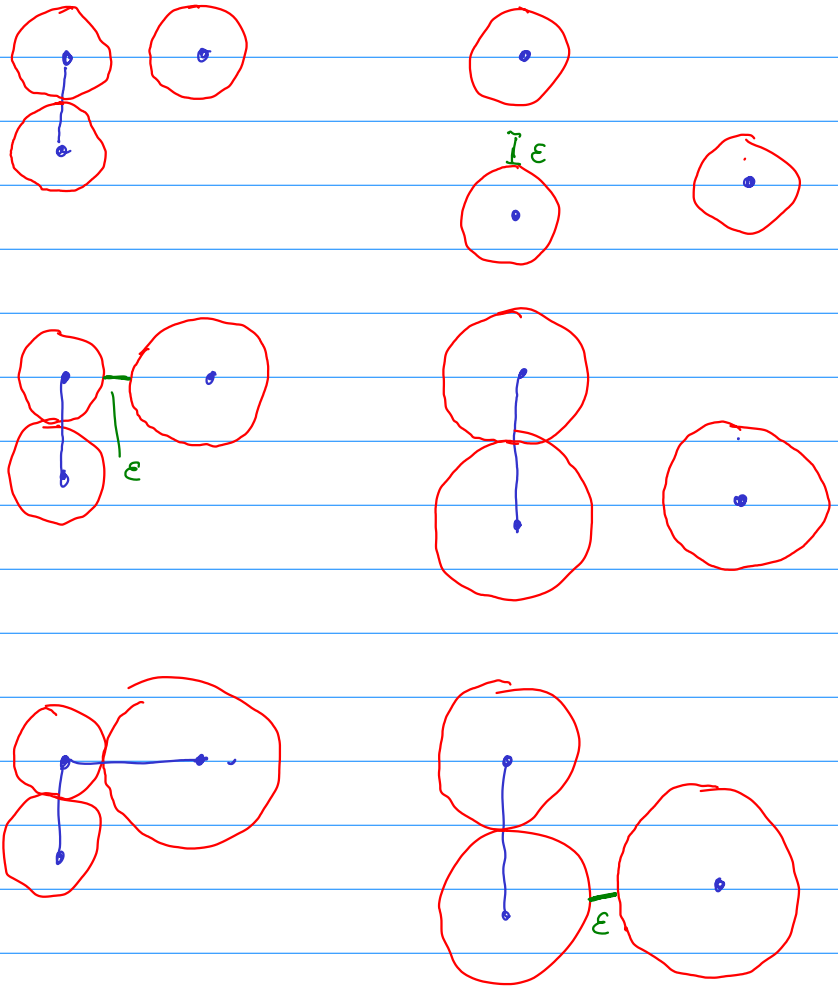
$$\text{parity}(T) := \begin{cases} 0 & \text{if } |T| \text{ is even,} \\ 1 & \text{if } |T| \text{ is odd.} \end{cases}$$

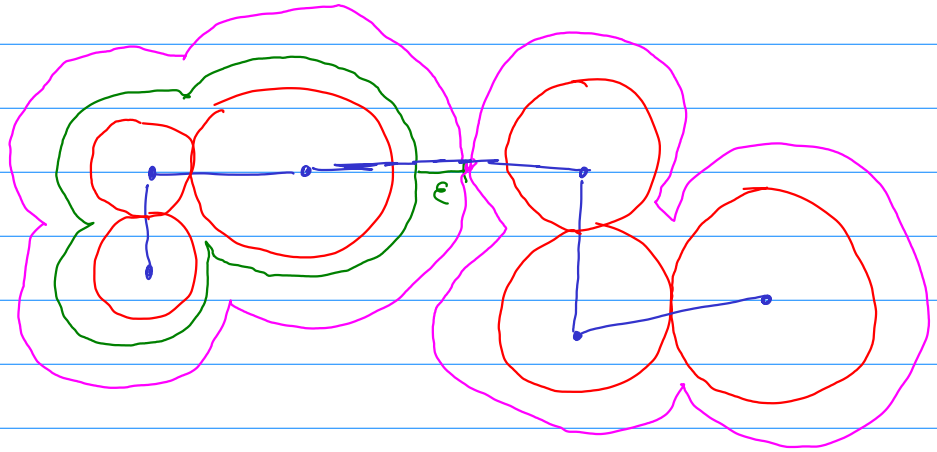
Goemans-Williamson Algorithm:

1. Let $F := (V, \emptyset)$, $\mathcal{T} := \{\{v\} \mid v \in V\}$, $y := 0$, and $Y := 0$;
2. If $|T|$ is even for all $T \in \mathcal{T}$, remove any even edges from F and stop;
3. Find an edge $e = \{v, w\}$ with $v \in T_i \in \mathcal{T}$, $w \in T_j \in \mathcal{T}$, $T_i \neq T_j$, that minimizes $\varepsilon := \bar{c}_e / (\text{parity}(T_i) + \text{parity}(T_j))$; $\Rightarrow$ one must be odd
4. For each $T \in \mathcal{T}$ such that $T = \{v\}$ for some $v \in V$, add ε to y_v;
5. For each $T \in \mathcal{T}$ with $|T| > 1$ odd, add ε to $Y_{\delta(T)}$;
6. Add e to F and update $\mathcal{T}$ by removing T_i and T_j and adding $T_i \cup T_j$;
7. Go to 2;

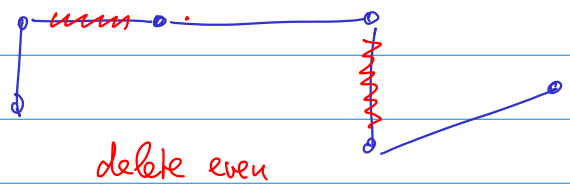
Example







even forest



delete even edges, gives matching directly in this case

Analysis of Goemans-Williamson Algorithm

Theorem 13.32.

The algorithm terminates with an even forest F^* and a feasible dual solution (y^*, Y^*) such that the weight of F^* is at most twice the value of the dual solution (y^*, Y^*).

Proof:... □

Remarks:

- Notice that the Goemans-Williamson Algorithm is considerably more efficient than the Blossom Algorithm.
- It is a primal-dual approximation algorithm with performance ratio 2.
- The algorithm is an example of a general class of primal-dual approximation algorithms for problems in combinatorial optimization.

Thm 13.32 $\Rightarrow$

$c(M) \leq c(\overline{T}^*) \leq$ 2 cost of dual solution (y^*, Y^*)

$c(M) \uparrow$ constructed from forest $\overline{T}^*$

$c(\overline{T}^*) \uparrow$ Δ-inequality

$\leq$ Thm 13.32

≤ 2 (cost of opt. solution of LP relaxation)

$\uparrow$ weak duality

LP-relaxation $\rightarrow \leq 2 \cdot$ (cost of optimal perfect matching)

$\Rightarrow$ algorithm is a 2-approximation algorithm □

$\uparrow$ performance ratio

3 different ways to evaluate approx algorithms

$A(I)$ ← value of solution

↑

↑ instance

$OPT(I)$ optimal value for I

also

consider minimization problem

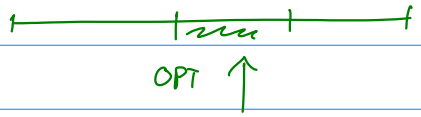
if $A(I) \leq \rho \cdot OPT(I) \quad \forall I \qquad (\rho \geq 1)$

then A is a ρ-approx algo

ρ is the relative performance guarantee

$A(I) \leq OPT(I) + \varepsilon$

additive performance



$A(I) \leq \rho \cdot OPT(I) + \varepsilon$

asymptotic performance guarantee