

# Approximation Schemes for Multi-Budgeted Independence Systems

Rico Zenklusen

MIT

Joint work with Fabrizio Grandoni

# Outline

## ① Motivation and previous results

- Motivation
- Our results

## ② Feasibilization

## ③ A PTAS for 2-BUDGETED MATCHINGS

## ④ A PTAS for K-BUDGETED MATROID INDEPENDENT SET

## ⑤ Conclusion

# General problem setting

- ▶ Ground set:  $S$ .
- ▶ Solution set:  $\mathcal{I} \subseteq 2^S$ .
- ▶ Weight:  $w : S \rightarrow \mathbb{Z}_+$ .
- ▶ Lengths:  $\ell_i : S \rightarrow \mathbb{Z}_+$  with budget  $B_i \in \mathbb{Z}_+ \quad \forall i \in [k] := \{1, \dots, k\}$ .

$$\max\{w(I) \mid I \in \mathcal{I}, \ell_i(I) \leq B_i \quad \forall i \in [k]\}$$

## Typical examples

- ▶  $k$ -budgeted maximum spanning tree (basis in a matroid)
- ▶  $k$ -budgeted maximum forest (independent set in a matroid)
- ▶  $k$ -budgeted maximum matching
- ▶  $k$ -budgeted shortest path
- ▶ ...

Which problems admit good approximation algorithms?

→ we are interested in PTASs.

# General problem setting

- ▶ Ground set:  $S$ .
- ▶ Solution set:  $\mathcal{I} \subseteq 2^S$ .
- ▶ Weight:  $w : S \rightarrow \mathbb{Z}_+$ .
- ▶ Lengths:  $\ell_i : S \rightarrow \mathbb{Z}_+$  with budget  $B_i \in \mathbb{Z}_+ \quad \forall i \in [k] := \{1, \dots, k\}$ .

$$\max\{w(I) \mid I \in \mathcal{I}, \ell_i(I) \leq B_i \quad \forall i \in [k]\}$$

## Typical examples

- ▶  $k$ -budgeted maximum spanning tree (basis in a matroid)
- ▶  $k$ -budgeted maximum forest (independent set in a matroid)
- ▶  $k$ -budgeted maximum matching
- ▶  $k$ -budgeted shortest path
- ▶ ...

Which problems admit good approximation algorithms?

→ we are interested in PTASs.

# General problem setting

- ▶ Ground set:  $S$ .
- ▶ Solution set:  $\mathcal{I} \subseteq 2^S$ .
- ▶ Weight:  $w : S \rightarrow \mathbb{Z}_+$ .
- ▶ Lengths:  $\ell_i : S \rightarrow \mathbb{Z}_+$  with budget  $B_i \in \mathbb{Z}_+ \quad \forall i \in [k] := \{1, \dots, k\}$ .

$$\max\{w(I) \mid I \in \mathcal{I}, \ell_i(I) \leq B_i \quad \forall i \in [k]\}$$

## Typical examples

- ▶  $k$ -budgeted maximum spanning tree (basis in a matroid)
- ▶  $k$ -budgeted maximum forest (independent set in a matroid)
- ▶  $k$ -budgeted maximum matching
- ▶  $k$ -budgeted shortest path
- ▶ ...

Which problems admit good approximation algorithms?

→ we are interested in PTASs.

# General problem setting

- ▶ Ground set:  $S$ .
- ▶ Solution set:  $\mathcal{I} \subseteq 2^S$ .
- ▶ Weight:  $w : S \rightarrow \mathbb{Z}_+$ .
- ▶ Lengths:  $\ell_i : S \rightarrow \mathbb{Z}_+$  with budget  $B_i \in \mathbb{Z}_+ \quad \forall i \in [k] := \{1, \dots, k\}$ .

$$\max\{w(I) \mid I \in \mathcal{I}, \ell_i(I) \leq B_i \quad \forall i \in [k]\}$$

## Typical examples

- ▶  $k$ -budgeted maximum spanning tree (basis in a matroid)
- ▶  $k$ -budgeted maximum forest (independent set in a matroid)
- ▶  $k$ -budgeted maximum matching
- ▶  $k$ -budgeted shortest path
- ▶ ...

Which problems admit good approximation algorithms?

→ we are interested in **PTASs**.

# Motivation

## Practical perspective: interesting modeling tool

- ▶ Natural way to deal with multiple partially conflicting objectives.
- ▶ Budgets can model hard resource constraints.

## Theoretical perspective: not well understood

- ▶ Many results are known for a single budget, but not much is known for 2 or more budgets.
- ▶ In some settings only multi-criteria PTASs are known:

$$\max\{w(I) \mid I \in \mathcal{I}, \ell_i(I) \leq (1 + \epsilon)B_i \ \forall i \in [k]\}$$

# Motivation

## Practical perspective: interesting modeling tool

- ▶ Natural way to deal with multiple partially conflicting objectives.
- ▶ Budgets can model hard resource constraints.

## Theoretical perspective: not well understood

- ▶ Many results are known for a single budget, but not much is known for 2 or more budgets.
- ▶ In some settings only multi-criteria PTASs are known:

$$\max\{w(I) \mid I \in \mathcal{I}, \ell_i(I) \leq (1 + \epsilon)B_i \ \forall i \in [k]\}$$

# Restriction to independence systems

## First observation

If all solutions  $I \in \mathcal{I}$  have same cardinality then even checking feasibility with 2 budgets is typically NP-complete.

## Proof

- ▶ Idea: reduce to exact weight problem:  $\max\{I \in \mathcal{I} \mid w(I) = W\}$ , where  $w : S \rightarrow \mathbb{Z}_+$ ,  $W \in \mathbb{Z}_+$ .

→ Checking feasibility is NP-complete for SPANNING TREES, PERFECT MATCHINGS, ...

- ▶ Let  $r = |I| = \mathbf{1}(I)$  for  $I \in \mathcal{I}$ .
- ▶ Write weight constraints as budget constraints and render coefficients positive:

$$\{I \in \mathcal{I} \mid w(I) \leq W, -w(I) \leq -W\} = \\ \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) + C \cdot \mathbf{1}(I) \leq -W + C \cdot r\}$$

□

- ▶ We consider problems whose solutions form an independence system  $(S, \mathcal{I})$ :  $I \in \mathcal{I}, J \subseteq I \Rightarrow J \in \mathcal{I}$ .

# Restriction to independence systems

## First observation

If all solutions  $I \in \mathcal{I}$  have same cardinality then even checking feasibility with 2 budgets is typically NP-complete.

## Proof

- ▶ Idea: reduce to exact weight problem:  $\max\{I \in \mathcal{I} \mid w(I) = W\}$ , where  $w : S \rightarrow \mathbb{Z}_+$ ,  $W \in \mathbb{Z}_+$ .

→ **Checking feasibility is NP-complete** for SPANNING TREES, PERFECT MATCHINGS, ...

- ▶ Let  $r = |I| = \mathbf{1}(I)$  for  $I \in \mathcal{I}$ .
- ▶ Write weight constraints as budget constraints and render coefficients positive:

$$\begin{aligned} \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) \leq -W\} = \\ \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) + C \cdot \mathbf{1}(I) \leq -W + C \cdot r\} \end{aligned}$$



- ▶ We consider problems whose solutions form an **independence system**  $(S, \mathcal{I})$ :  $I \in \mathcal{I}, J \subseteq I \Rightarrow J \in \mathcal{I}$ .

# Restriction to independence systems

## First observation

If all solutions  $I \in \mathcal{I}$  have same cardinality then even checking feasibility with 2 budgets is typically NP-complete.

## Proof

- ▶ Idea: reduce to exact weight problem:  $\max\{I \in \mathcal{I} \mid w(I) = W\}$ , where  $w : S \rightarrow \mathbb{Z}_+$ ,  $W \in \mathbb{Z}_+$ .

→ **Checking feasibility is NP-complete** for SPANNING TREES, PERFECT MATCHINGS, ...

- ▶ Let  $r = |I| = \mathbf{1}(I)$  for  $I \in \mathcal{I}$ .
- ▶ Write weight constraints as budget constraints and render coefficients positive:

$$\begin{aligned} \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) \leq -W\} = \\ \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) + C \cdot \mathbf{1}(I) \leq -W + C \cdot r\} \end{aligned}$$



- ▶ We consider problems whose solutions form an **independence system**  $(S, \mathcal{I})$ :  $I \in \mathcal{I}, J \subseteq I \Rightarrow J \in \mathcal{I}$ .

# Restriction to independence systems

## First observation

If all solutions  $I \in \mathcal{I}$  have same cardinality then even checking feasibility with 2 budgets is typically NP-complete.

## Proof

- ▶ Idea: reduce to exact weight problem:  $\max\{I \in \mathcal{I} \mid w(I) = W\}$ , where  $w : S \rightarrow \mathbb{Z}_+$ ,  $W \in \mathbb{Z}_+$ .

→ **Checking feasibility is NP-complete** for SPANNING TREES, PERFECT MATCHINGS, ...

- ▶ Let  $r = |I| = \mathbf{1}(I)$  for  $I \in \mathcal{I}$ .
- ▶ Write weight constraints as budget constraints and render coefficients positive:

$$\begin{aligned} \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) \leq -W\} = \\ \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) + C \cdot \mathbf{1}(I) \leq -W + C \cdot r\} \end{aligned}$$



- ▶ We consider problems whose solutions form an **independence system**  $(S, \mathcal{I})$ :  $I \in \mathcal{I}, J \subseteq I \Rightarrow J \in \mathcal{I}$ .

# Restriction to independence systems

## First observation

If all solutions  $I \in \mathcal{I}$  have same cardinality then even checking feasibility with 2 budgets is typically NP-complete.

## Proof

- ▶ Idea: reduce to exact weight problem:  $\max\{I \in \mathcal{I} \mid w(I) = W\}$ , where  $w : S \rightarrow \mathbb{Z}_+$ ,  $W \in \mathbb{Z}_+$ .

→ **Checking feasibility is NP-complete** for SPANNING TREES, PERFECT MATCHINGS, ...

- ▶ Let  $r = |I| = \mathbf{1}(I)$  for  $I \in \mathcal{I}$ .
- ▶ Write weight constraints as budget constraints and render coefficients positive:

$$\begin{aligned} \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) \leq -W\} = \\ \{I \in \mathcal{I} \mid w(I) \leq W, -w(I) + C \cdot \mathbf{1}(I) \leq -W + C \cdot r\} \end{aligned}$$

- ▶ We consider problems whose solutions form an **independence system**  $(S, \mathcal{I})$ :  $I \in \mathcal{I}, J \subseteq I \Rightarrow J \in \mathcal{I}$ . □

# Some relevant previous results

## Results for 1-budgeted problems

- ▶ FPTAS for **1-BUDGETED SHORTEST PATH** (Warburton [1987], Hassin [1992], Lorenz and Raz [2001]).
    - ➔ dynamic programming and rounding & scaling
  - ▶ PTAS for **1-BUDGETED SPANNING TREE** (works on any matroid) (Ravi and Goemans [1996]).
    - ➔ exploits that edges on base polytope are short
  - ▶ PTAS for **1-BUDGETED MATCHING** and **1-BUDGETED MATROID INTERSECTION INDEPENDENT SET** (Berger et al. [2009]).
    - ➔ careful patching of two matchings/independent sets
- 

## Multi-criteria algorithms (based on exact weight algorithms)

Combining results from Papadimitriou and Yannakakis [2000], Mulmuley et al. [1987], and Camerini et al. [1992], multi-criteria PRASs (with very high running times) can be obtained for

- ▶ **MATCHING**,
- ▶ **MATROID INTERSECTION INDEPENDENT SET** for representable matroids.

# Some relevant previous results

## Results for 1-budgeted problems

- ▶ FPTAS for **1-BUDGETED SHORTEST PATH** (Warburton [1987], Hassin [1992], Lorenz and Raz [2001]).
    - ➔ dynamic programming and rounding & scaling
  - ▶ PTAS for **1-BUDGETED SPANNING TREE** (works on any matroid) (Ravi and Goemans [1996]).
    - ➔ exploits that edges on base polytope are short
  - ▶ PTAS for **1-BUDGETED MATCHING** and **1-BUDGETED MATROID INTERSECTION INDEPENDENT SET** (Berger et al. [2009]).
    - ➔ careful patching of two matchings/independent sets
- 

## Multi-criteria algorithms (based on exact weight algorithms)

Combining results from Papadimitriou and Yannakakis [2000], Mulmuley et al. [1987], and Camerini et al. [1992], multi-criteria PRASs (with very high running times) can be obtained for

- ▶ **MATCHING**,
- ▶ **MATROID INTERSECTION INDEPENDENT SET** for representable matroids.

# Our results

## Feasibilization (multi-criteria $\rightarrow$ multi-budgeted)

Technique to transform multi-criteria algorithms for independence systems into multi-budgeted algorithms (PTAS  $\rightarrow$  PTAS, PRAS  $\rightarrow$  PRAS).

- $\rightarrow$  Filtering and scaling of budget constraints.

## PTAS for 2-BUDGETED MATCHING

- $\rightarrow$  General framework that might be useful in other contexts.
- $\rightarrow$  Based on a topological property about curves in  $\mathbb{R}^2$ .
- $\rightarrow$  Proving a generalization of the curve property allows for applying the same technique to obtain a PTAS for  $k$ -BUDGETED MATCHING.
  - $\rightarrow$  Generalized Necklace Splitting Problem.

## PTAS for $k$ -BUDGETED MATROID INDEPENDENT SET

- $\rightarrow$  Low dimensional faces on matroid polytope have few fractional values.

# Outline

## ① Motivation and previous results

- Motivation
- Our results

## ② Feasibilization

## ③ A PTAS for 2-BUDGETED MATCHINGS

## ④ A PTAS for K-BUDGETED MATROID INDEPENDENT SET

## ⑤ Conclusion

# Feasibilization

## A nice observation with several consequences

$k$ -criteria algorithms can be transformed into  $k$ -budgeted algorithms:

- ▶ PTAS  $\rightarrow$  PTAS, PRAS  $\rightarrow$  PRAS

### Algorithm

1. **Filtering:** Guess the  $k/\epsilon$  heaviest elements  $E_G$  in opt.
2. **Scaling:** Round each budget down by a factor of  $1 - \frac{\epsilon}{k+1}$  and apply multi-criteria algorithm with precision  $\frac{\epsilon}{k+1}$  to obtain  $E_H$ .
3. **Return:**  $E_G \cup E_H$ .

- ▶ In our context: mainly interesting to apply transformation:

pseudo-poly. randomized exact weight algo  $\rightarrow$  PRAS for  $k$ -multi-criteria  $\rightarrow$  PRAS for  $k$ -budgeted problem.

to algos of Mulmuley et al. [1987] and Camerini et al. [1992].

- ▶ PRAS for  $k$ -BUDGETED MATCHINGS and  $k$ -BUDGETED MATROID INTERSECTION INDEPENDENT SET for representable matroids.
- ▶ However: these algorithms are **very slow** (and randomized).

Derandomization of above exact weight algos is longstanding open problem.

# Feasibilization

## A nice observation with several consequences

$k$ -criteria algorithms can be transformed into  $k$ -budgeted algorithms:

- ▶ PTAS  $\rightarrow$  PTAS, PRAS  $\rightarrow$  PRAS

### Algorithm

1. **Filtering:** Guess the  $k/\epsilon$  heaviest elements  $E_G$  in opt.
2. **Scaling:** Round each budget down by a factor of  $1 - \frac{\epsilon}{k+1}$  and apply multi-criteria algorithm with precision  $\frac{\epsilon}{k+1}$  to obtain  $E_H$ .
3. **Return:**  $E_G \cup E_H$ .

- ▶ In our context: mainly interesting to apply transformation:

pseudo-poly. randomized exact weight algo  $\rightarrow$  PRAS for  $k$ -multi-criteria  $\rightarrow$  PRAS for  $k$ -budgeted problem.

to algos of [Mulmuley et al. \[1987\]](#) and [Camerini et al. \[1992\]](#).

- ▶ PRAS for  $k$ -BUDGETED MATCHINGS and  $k$ -BUDGETED MATROID INTERSECTION INDEPENDENT SET for representable matroids.
- ▶ However: these algorithms are **very slow** (and randomized).

Derandomization of above exact weight algos is longstanding open problem.

# Feasibilization

## A nice observation with several consequences

$k$ -criteria algorithms can be transformed into  $k$ -budgeted algorithms:

- ▶ PTAS  $\rightarrow$  PTAS, PRAS  $\rightarrow$  PRAS

### Algorithm

1. **Filtering:** Guess the  $k/\epsilon$  heaviest elements  $E_G$  in opt.
2. **Scaling:** Round each budget down by a factor of  $1 - \frac{\epsilon}{k+1}$  and apply multi-criteria algorithm with precision  $\frac{\epsilon}{k+1}$  to obtain  $E_H$ .
3. **Return:**  $E_G \cup E_H$ .

- ▶ In our context: mainly interesting to apply transformation:

pseudo-poly. randomized exact weight algo  $\rightarrow$  PRAS for  $k$ -multi-criteria  $\rightarrow$  PRAS for  $k$ -budgeted problem.

to algos of [Mulmuley et al. \[1987\]](#) and [Camerini et al. \[1992\]](#).

- ▶ PRAS for  $k$ -BUDGETED MATCHINGS and  $k$ -BUDGETED MATROID INTERSECTION INDEPENDENT SET for representable matroids.
- ▶ However: these algorithms are **very slow** (and randomized).

Derandomization of above exact weight algos is longstanding open problem.

# Outline

## ① Motivation and previous results

- Motivation
- Our results

## ② Feasibilization

## ③ A PTAS for 2-BUDGETED MATCHINGS

## ④ A PTAS for $k$ -BUDGETED MATROID INDEPENDENT SET

## ⑤ Conclusion

## Filtering (a classical idea to be used later)

- ▶ To obtain a PTAS it suffices to provide an efficient algo returning a solution of  $\text{value} \geq \text{opt} - c \cdot w_{\max}$ , where
  - ▶  $w_{\max} := \max\{w(s) \mid s \in \mathcal{S}\}$ ,
  - ▶  $c$ : constant.

### Filtering

$\mathcal{A}$  can be transformed into a PTAS for the  $k$ -budgeted problem:

- ▶ Guess the  $c/\epsilon$  heaviest elements of an optimal solution.
  - ▶ Prune the problem by removing all elements heavier than the minimum weight guessed element.
  - ▶ For each guess run algorithm  $\mathcal{A}$ .
  - ▶ Return best output.
- 
- ▶ We present a polynomial algo for 2-BUDGETED MATCHING returning a solution of weight  $\geq \text{opt} - 6w_{\max}$ .

## Filtering (a classical idea to be used later)

- ▶ To obtain a PTAS it suffices to provide an efficient algo returning a solution of  $\text{value} \geq \text{opt} - c \cdot w_{\max}$ , where
  - ▶  $w_{\max} := \max\{w(s) \mid s \in S\}$ ,
  - ▶  $c$ : constant.

### Filtering

$\mathcal{A}$  can be transformed into a PTAS for the  $k$ -budgeted problem:

- ▶ **Guess the  $c/\epsilon$  heaviest elements** of an optimal solution.
  - ▶ Prune the problem by removing all elements heavier than the minimum weight guessed element.
  - ▶ For each guess run algorithm  $\mathcal{A}$ .
  - ▶ Return best output.
- 
- ▶ We present a polynomial algo for 2-BUDGETED MATCHING returning a solution of weight  $\geq \text{opt} - 6w_{\max}$ .

## Filtering (a classical idea to be used later)

- ▶ To obtain a PTAS it suffices to provide an efficient algo returning a solution of  $\text{value} \geq \text{opt} - c \cdot w_{\max}$ , where
  - ▶  $w_{\max} := \max\{w(s) \mid s \in S\}$ ,
  - ▶  $c$ : constant.

### Filtering

$\mathcal{A}$  can be transformed into a PTAS for the  $k$ -budgeted problem:

- ▶ **Guess the  $c/\epsilon$  heaviest elements** of an optimal solution.
  - ▶ Prune the problem by removing all elements heavier than the minimum weight guessed element.
  - ▶ For each guess run algorithm  $\mathcal{A}$ .
  - ▶ Return best output.
- 
- ▶ We present a polynomial algo for 2-BUDGETED MATCHING returning a solution of weight  $\geq \text{opt} - 6w_{\max}$ .

# General framework

Consider a graph  $G = (V, E)$  with matchings  $\mathcal{M} \subseteq 2^E$  and matching polytope  $P_{\mathcal{M}} = \text{conv}(\{\mathbf{1}_M \mid M \in \mathcal{M}\})$ .

1. Get optimal basic solution  $x^*$  to

$$\begin{aligned} \max_{x \in P_{\mathcal{M}}} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in \{1, 2\} \end{aligned}$$

2. Compute convex decomposition

$$x^* = \alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2} + \alpha_3 \mathbf{1}_{M_3},$$

where  $M_1, M_2, M_3 \in \mathcal{M}$ .

3. Merge  $M_1$  and  $M_2$  to get

$$M_{1,2} \in \mathcal{M} \text{ "close" to}$$
$$x_{1,2} := \frac{1}{\alpha_1 + \alpha_2} (\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2}).$$

4. Then merge  $M_{1,2}$  and  $M_3$  to get

$$M \in \mathcal{M} \text{ "close" to}$$
$$x^* = (\alpha_1 + \alpha_2) \mathbf{1}_{M_{1,2}} + \alpha_3 \mathbf{1}_{M_3}.$$

# General framework

Consider a graph  $G = (V, E)$  with matchings  $\mathcal{M} \subseteq 2^E$  and matching polytope  $P_{\mathcal{M}} = \text{conv}(\{\mathbf{1}_M \mid M \in \mathcal{M}\})$ .

1. Get optimal basic solution  $x^*$  to

$$\begin{aligned} \max_{x \in P_{\mathcal{M}}} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in \{1, 2\} \end{aligned}$$

2. Compute convex decomposition

$$x^* = \alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2} + \alpha_3 \mathbf{1}_{M_3},$$

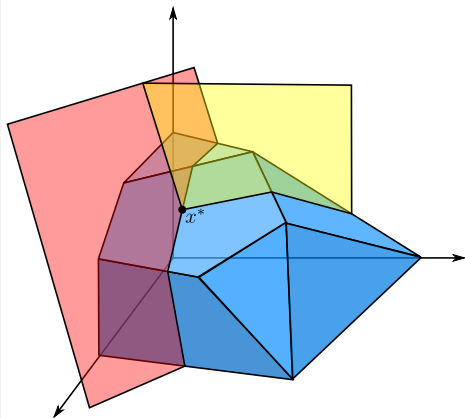
where  $M_1, M_2, M_3 \in \mathcal{M}$ .

3. Merge  $M_1$  and  $M_2$  to get

$$M_{1,2} \in \mathcal{M} \text{ "close" to } x_{1,2} := \frac{1}{\alpha_1 + \alpha_2} (\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2}).$$

4. Then merge  $M_{1,2}$  and  $M_3$  to get

$$M \in \mathcal{M} \text{ "close" to } x^* = (\alpha_1 + \alpha_2) \mathbf{1}_{M_{1,2}} + \alpha_3 \mathbf{1}_{M_3}.$$



# General framework

Consider a graph  $G = (V, E)$  with matchings  $\mathcal{M} \subseteq 2^E$  and matching polytope  $P_{\mathcal{M}} = \text{conv}(\{\mathbf{1}_M \mid M \in \mathcal{M}\})$ .

1. Get optimal basic solution  $x^*$  to

$$\begin{aligned} \max_{x \in P_{\mathcal{M}}} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in \{1, 2\} \end{aligned}$$

2. Compute convex decomposition

$$x^* = \alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2} + \alpha_3 \mathbf{1}_{M_3},$$

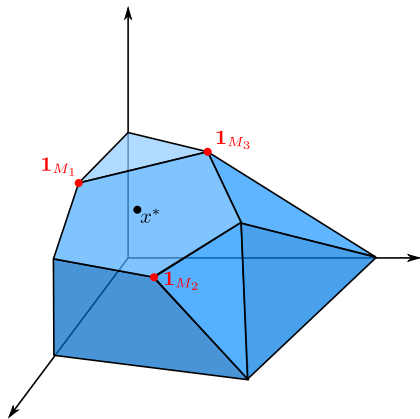
where  $M_1, M_2, M_3 \in \mathcal{M}$ .

3. Merge  $M_1$  and  $M_2$  to get

$$M_{1,2} \in \mathcal{M} \text{ "close" to } x_{1,2} := \frac{1}{\alpha_1 + \alpha_2} (\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2}).$$

4. Then merge  $M_{1,2}$  and  $M_3$  to get

$$M \in \mathcal{M} \text{ "close" to } x^* = (\alpha_1 + \alpha_2) \mathbf{1}_{M_{1,2}} + \alpha_3 \mathbf{1}_{M_3}.$$



# General framework

Consider a graph  $G = (V, E)$  with matchings  $\mathcal{M} \subseteq 2^E$  and matching polytope  $P_{\mathcal{M}} = \text{conv}(\{\mathbf{1}_M \mid M \in \mathcal{M}\})$ .

1. Get optimal basic solution  $x^*$  to

$$\begin{aligned} \max_{x \in P_{\mathcal{M}}} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in \{1, 2\} \end{aligned}$$

2. Compute convex decomposition

$$x^* = \alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2} + \alpha_3 \mathbf{1}_{M_3},$$

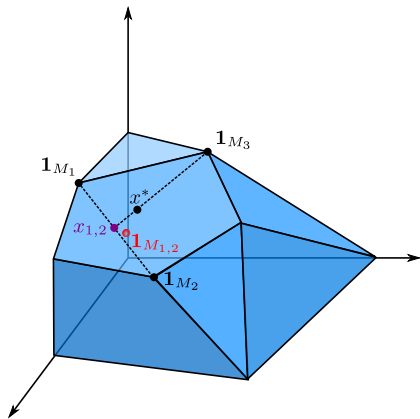
where  $M_1, M_2, M_3 \in \mathcal{M}$ .

3. Merge  $M_1$  and  $M_2$  to get

$$M_{1,2} \in \mathcal{M} \text{ "close" to}$$
$$x_{1,2} := \frac{1}{\alpha_1 + \alpha_2} (\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2}).$$

4. Then merge  $M_{1,2}$  and  $M_3$  to get

$$M \in \mathcal{M} \text{ "close" to}$$
$$x^* = (\alpha_1 + \alpha_2) \mathbf{1}_{M_{1,2}} + \alpha_3 \mathbf{1}_{M_3}.$$



# General framework

Consider a graph  $G = (V, E)$  with matchings  $\mathcal{M} \subseteq 2^E$  and matching polytope  $P_{\mathcal{M}} = \text{conv}(\{\mathbf{1}_M \mid M \in \mathcal{M}\})$ .

1. Get optimal basic solution  $x^*$  to

$$\begin{aligned} \max_{x \in P_{\mathcal{M}}} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in \{1, 2\} \end{aligned}$$

2. Compute convex decomposition

$$x^* = \alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2} + \alpha_3 \mathbf{1}_{M_3},$$

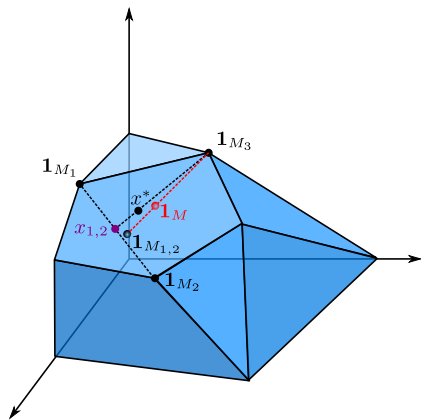
where  $M_1, M_2, M_3 \in \mathcal{M}$ .

3. Merge  $M_1$  and  $M_2$  to get

$$M_{1,2} \in \mathcal{M} \text{ "close" to}$$
$$x_{1,2} := \frac{1}{\alpha_1 + \alpha_2} (\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2}).$$

4. Then merge  $M_{1,2}$  and  $M_3$  to get

$$M \in \mathcal{M} \text{ "close" to}$$
$$x^* = (\alpha_1 + \alpha_2) \mathbf{1}_{M_{1,2}} + \alpha_3 \mathbf{1}_{M_3}.$$



# Merging two matchings I

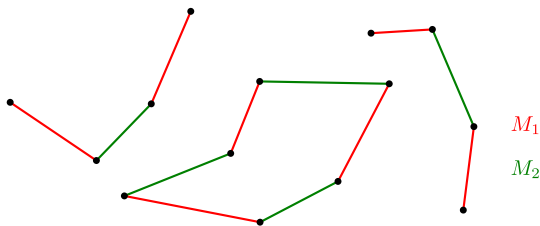
**Goal:** get  $M_{1,2} \in \mathcal{M}$  “close” to  $x_{1,2} := \frac{1}{\alpha_1 + \alpha_2}(\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2})$ :

- ▶  $\ell_i(M_{1,2}) \leq \ell_i(x_{1,2})$  for  $i \in \{1, 2\}$ ,
- ▶  $w(M_{1,2}) \geq w^T x_{1,2} - 2w_{\max}$ .

# Merging two matchings I

**Goal:** get  $M_{1,2} \in \mathcal{M}$  "close" to  $x_{1,2} := \frac{1}{\alpha_1 + \alpha_2}(\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2})$ :

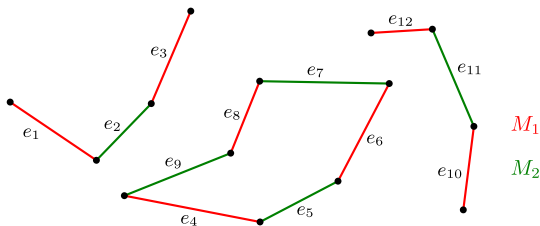
- ▶  $l_i(M_{1,2}) \leq l_i(x_{1,2})$  for  $i \in \{1, 2\}$ ,
- ▶  $w(M_{1,2}) \geq w^T x_{1,2} - 2w_{\max}$ .



# Merging two matchings I

**Goal:** get  $M_{1,2} \in \mathcal{M}$  "close" to  $x_{1,2} := \frac{1}{\alpha_1 + \alpha_2}(\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2})$ :

- ▶  $l_i(M_{1,2}) \leq l_i(x_{1,2})$  for  $i \in \{1, 2\}$ ,
- ▶  $w(M_{1,2}) \geq w^T x_{1,2} - 2w_{\max}$ .



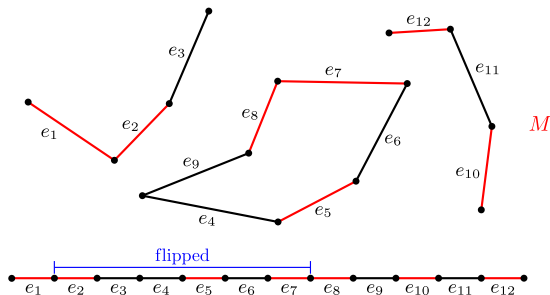
- ▶ **Order edges:** adjacent edges in same comp. are numbered consecutively.
- ▶ By **flipping** a (circular) subinterval of edges of  $M_1$ , an *almost matching*  $y \in [0, 1]^E$  is obtained: **removing at most 2 edges**,  $y$  can be transformed into a matching.



# Merging two matchings I

**Goal:** get  $M_{1,2} \in \mathcal{M}$  "close" to  $x_{1,2} := \frac{1}{\alpha_1 + \alpha_2}(\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2})$ :

- ▶  $l_i(M_{1,2}) \leq l_i(x_{1,2})$  for  $i \in \{1, 2\}$ ,
- ▶  $w(M_{1,2}) \geq w^T x_{1,2} - 2w_{\max}$ .

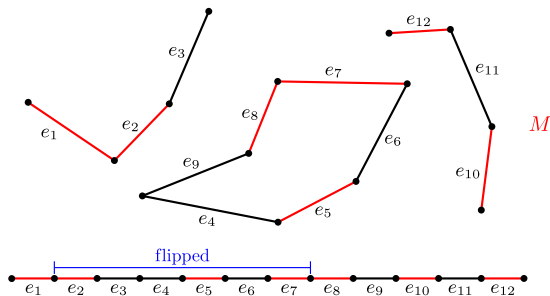


- ▶ **Order edges:** adjacent edges in same comp. are numbered consecutively.
- ▶ By **flipping** a (circular) subinterval of edges of  $M_1$ , an *almost matching*  $y \in [0, 1]^E$  is obtained: **removing at most 2 edges,  $y$  can be transformed into a matching.**

# Merging two matchings I

**Goal:** get  $M_{1,2} \in \mathcal{M}$  "close" to  $x_{1,2} := \frac{1}{\alpha_1 + \alpha_2} (\alpha_1 \mathbf{1}_{M_1} + \alpha_2 \mathbf{1}_{M_2})$ :

- ▶  $l_i(M_{1,2}) \leq l_i(x_{1,2})$  for  $i \in \{1, 2\}$ ,
- ▶  $w(M_{1,2}) \geq w^T x_{1,2} - 2w_{\max}$ .

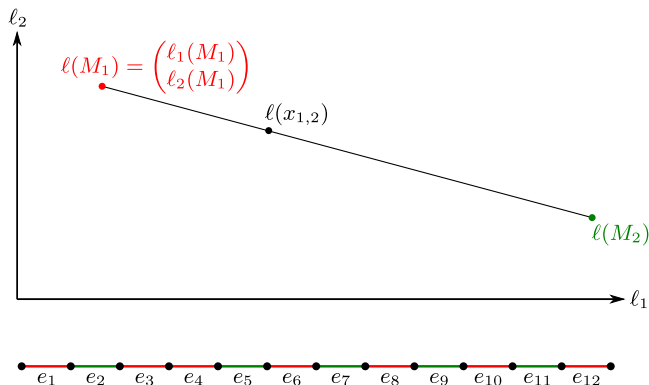


## Approach to obtain $M_{1,2}$

1. Find almost matching  $y \in [0, 1]^E$  "very close" to  $x_{1,2}$ .
2. Remove at most 2 edges from  $y$  to obtain  $M_{1,2}$ .

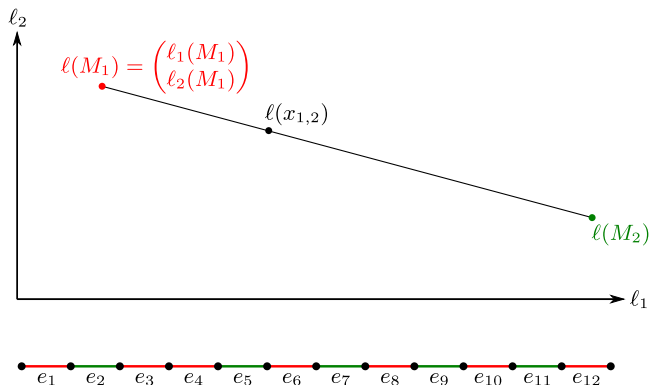
# Merging two matchings II

Consider the length functions (objective function to be considered later).



# Merging two matchings II

Consider the length functions (objective function to be considered later).

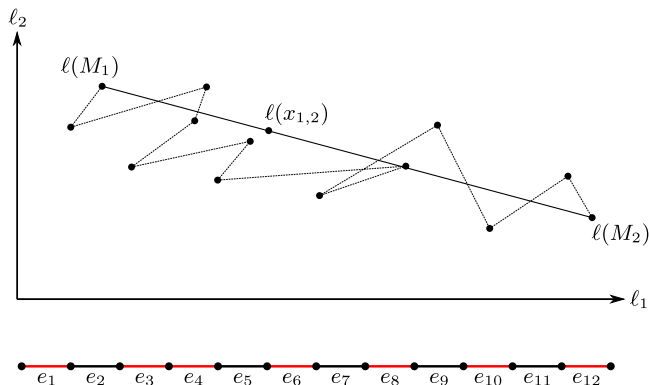


**Goal:** get almost matching  $y$  through edge flips with  $l(y) = l(x_{1,2})$ .

- ▶ To show later: the weight of such a  $y$  must also be close to  $w(x_{1,2})$ .

# Merging two matchings II

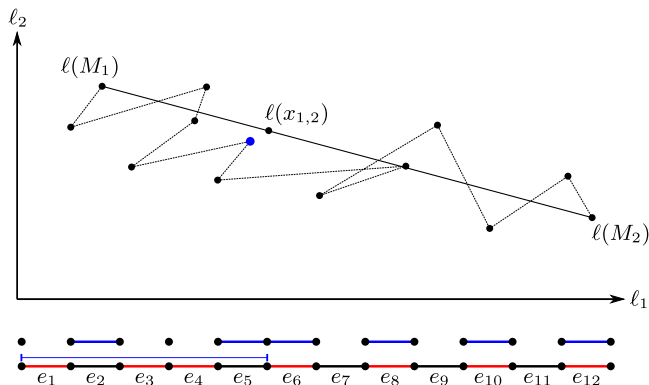
Consider the length functions (objective function to be considered later).



- ▶ Almost matchings “between”  $M_1$  and  $M_2$  can be obtained by starting with  $M_1$  and **flipping**  $e_1, \dots, e_i$  for  $i \in \{1, 2, \dots\}$ .

# Merging two matchings II

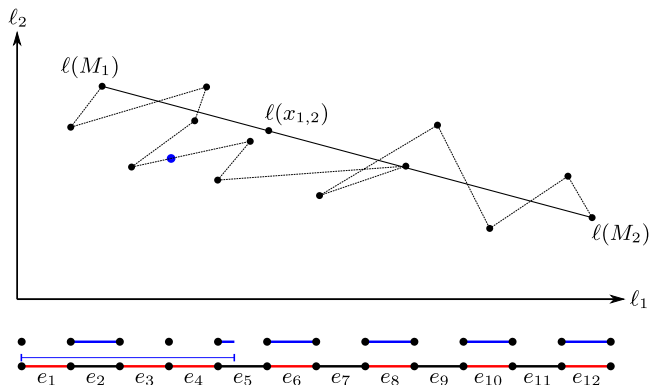
Consider the length functions (objective function to be considered later).



- ▶ Almost matchings “between”  $M_1$  and  $M_2$  can be obtained by starting with  $M_1$  and flipping  $e_1, \dots, e_i$  for  $i \in \{1, 2, \dots\}$ .

# Merging two matchings II

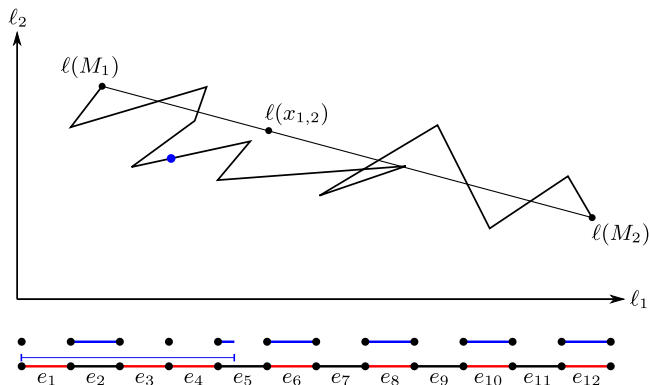
Consider the length functions (objective function to be considered later).



- ▶ Flips can also be performed with a **fractional start and/or endpoint**.

# Merging two matchings II

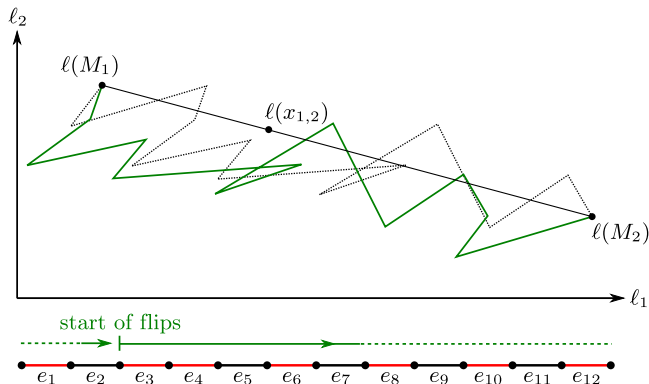
Consider the length functions (objective function to be considered later).



- ▶ Flips can also be performed with a **fractional start and/or endpoint**.
  - ➔ For any point on the line there is a corresponding almost matching.

# Merging two matchings II

Consider the length functions (objective function to be considered later).

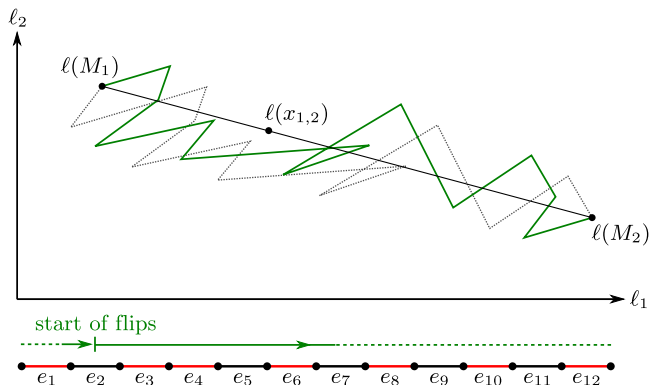


- ▶ Various curves can be obtained by **changing the start of the flip interval**

...

# Merging two matchings II

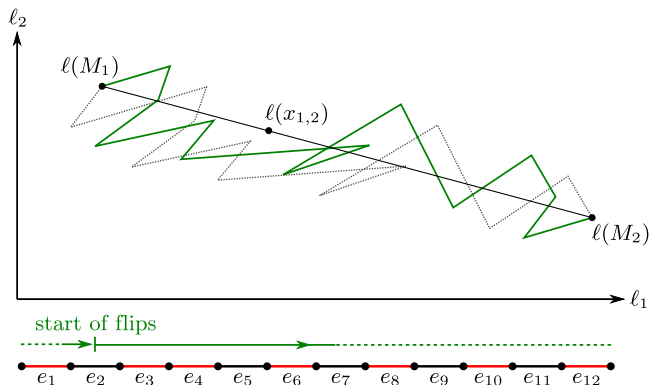
Consider the length functions (objective function to be considered later).



- ▶ Various curves can be obtained by **changing the start of the flip interval** ... also **fractionally**.

# Merging two matchings II

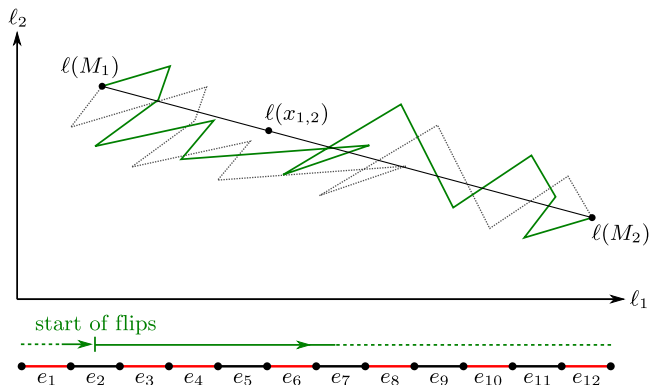
Consider the length functions (objective function to be considered later).



Is there a starting point such that the **curve contains**  $l(x_{1,2})$ ?

# Merging two matchings II

Consider the length functions (objective function to be considered later).

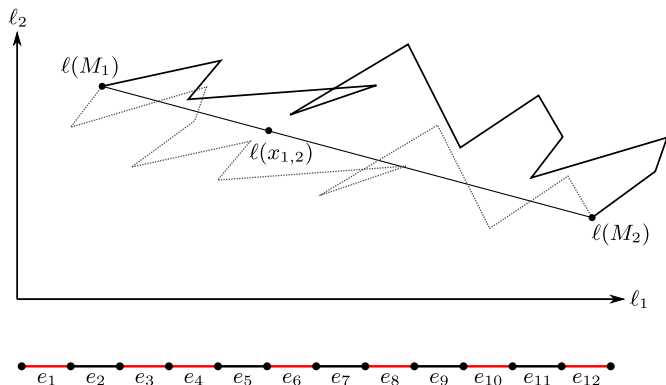


Is there a starting point such that the **curve contains**  $l(x_{1,2})$ ?

→ Yes, such a starting point exists.

# Merging two matchings II

Consider the length functions (objective function to be considered later).

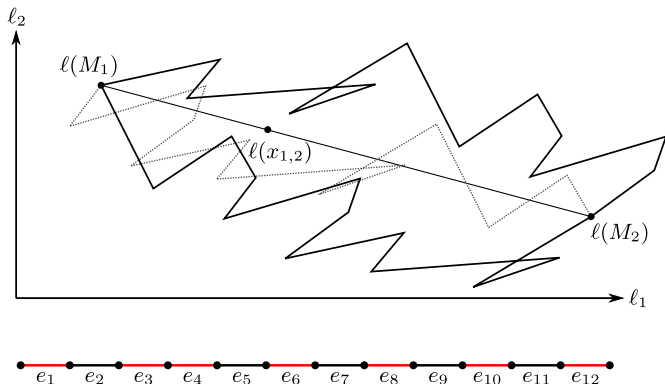


## Intuitive explanation:

- ▶ Consider the line  $L$  connecting  $\ell(M_1)$  and  $\ell(M_2)$ .
- ▶ Starting point = point furthest below  $L \Rightarrow$  curve lies above  $L$ .
- ▶ Starting point = point furthest above  $L \Rightarrow$  curve lies below  $L$ .
- ▶  $\exists$  continuous transformation between curves  $\Rightarrow$  it is possible to hit  $\ell(x_{1,2})$ .

# Merging two matchings II

Consider the length functions (objective function to be considered later).

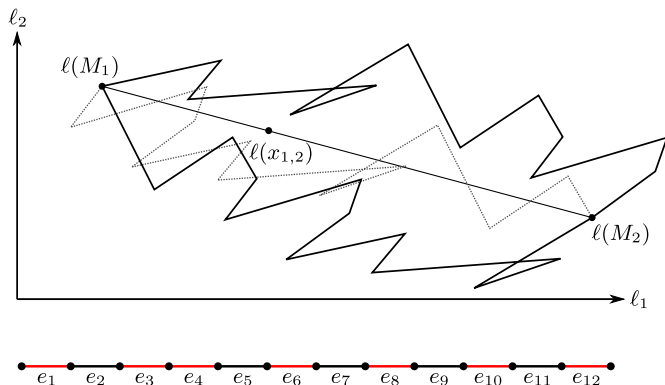


## Intuitive explanation:

- ▶ Consider the line  $L$  connecting  $l(M_1)$  and  $l(M_2)$ .
- ▶ Starting point = point furthest below  $L \Rightarrow$  curve lies above  $L$ .
- ▶ Starting point = point furthest above  $L \Rightarrow$  curve lies below  $L$ .
- ▶  $\exists$  continuous transformation between curves  $\Rightarrow$  it is possible to hit  $l(x_{1,2})$ .

# Merging two matchings II

Consider the length functions (objective function to be considered later).

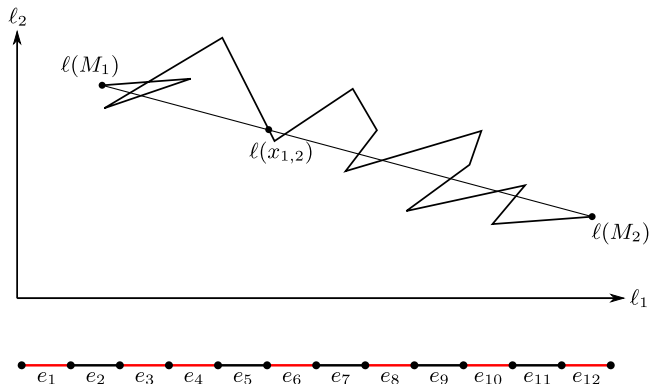


## Intuitive explanation:

- ▶ Consider the line  $L$  connecting  $l(M_1)$  and  $l(M_2)$ .
- ▶ Starting point = point furthest below  $L \Rightarrow$  curve lies above  $L$ .
- ▶ Starting point = point furthest above  $L \Rightarrow$  curve lies below  $L$ .
- ▶  $\exists$  continuous transformation between curves  $\Rightarrow$  it is possible to hit  $l(x_{1,2})$ .

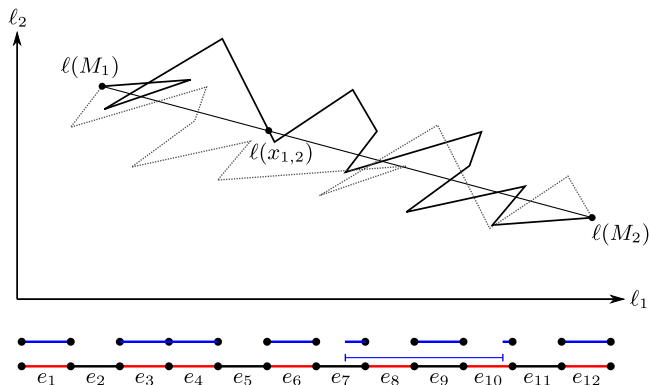
# Merging two matchings II

Consider the length functions (objective function to be considered later).



# Merging two matchings II

Consider the length functions (objective function to be considered later).



## Theorem

Starting with  $M_1$ , it is possible to flip a (possibly fractional) subinterval of edges to obtain an almost matching  $y$  with  $l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .

# What about the objective value? (sketch of reasoning)

Almost matching  $y$  obtained by merging  $M_1$  and  $M_2$  has weight close to  $x_{1,2}$ .

- ▶ Recall that by construction:  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2$ .
- ▶ Lagrangian weight:  $\mathcal{L}(x) := w(x) + \lambda_1^* \ell_1(x) + \lambda_2^* \ell_2(x) \quad \forall x \in [0, 1]^E$ .
  - Interpreted as a vector,  $\mathcal{L} \perp F$ , where  $F$  is the 2-dimensional face of  $P_{\mathcal{M}}$  containing  $\mathbf{1}_{M_1}, \mathbf{1}_{M_2}, \mathbf{1}_{M_3}$ .

## Idea:

- ▶ Show that  $y$  is almost on the face  $F$ :
  - Let  $\bar{y} := \mathbf{1}_{M_1} + \mathbf{1}_{M_2} - y \Rightarrow \bar{y}$  is also an almost matching.
  - Hence,  $\frac{1}{2}(y + \bar{y}) = \frac{1}{2}(\mathbf{1}_{M_1} + \mathbf{1}_{M_2}) \in F$ .
  - $y$  and  $\bar{y}$  are almost in  $P_{\mathcal{M}} \rightarrow y, \bar{y}$  almost on  $F$ .
- ▶  $\mathcal{L}(y) \approx \mathcal{L}(x_{1,2})$ .
- ▶ Since  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2 \rightarrow w(y) \approx w(x_{1,2})$ .

# What about the objective value? (sketch of reasoning)

Almost matching  $y$  obtained by merging  $M_1$  and  $M_2$  has weight close to  $x_{1,2}$ .

- ▶ Recall that by construction:  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2$ .
- ▶ Lagrangian weight:  $\mathcal{L}(x) := w(x) + \lambda_1^* \ell_1(x) + \lambda_2^* \ell_2(x) \quad \forall x \in [0, 1]^E$ .
  - Interpreted as a vector,  $\mathcal{L} \perp F$ , where  $F$  is the 2-dimensional face of  $P_{\mathcal{M}}$  containing  $\mathbf{1}_{M_1}, \mathbf{1}_{M_2}, \mathbf{1}_{M_3}$ .

## Idea:

- ▶ Show that  $y$  is almost on the face  $F$ :
  - Let  $\bar{y} := \mathbf{1}_{M_1} + \mathbf{1}_{M_2} - y \Rightarrow \bar{y}$  is also an almost matching.
  - Hence,  $\frac{1}{2}(y + \bar{y}) = \frac{1}{2}(\mathbf{1}_{M_1} + \mathbf{1}_{M_2}) \in F$ .
  - $y$  and  $\bar{y}$  are almost in  $P_{\mathcal{M}} \rightarrow y, \bar{y}$  almost on  $F$ .
- ▶  $\mathcal{L}(y) \approx \mathcal{L}(x_{1,2})$ .
- ▶ Since  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2 \rightarrow w(y) \approx w(x_{1,2})$ .

# What about the objective value? (sketch of reasoning)

Almost matching  $y$  obtained by merging  $M_1$  and  $M_2$  has weight close to  $x_{1,2}$ .

- ▶ Recall that by construction:  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2$ .
- ▶ Lagrangian weight:  $\mathcal{L}(x) := w(x) + \lambda_1^* \ell_1(x) + \lambda_2^* \ell_2(x) \quad \forall x \in [0, 1]^E$ .
  - Interpreted as a vector,  $\mathcal{L} \perp F$ , where  $F$  is the 2-dimensional face of  $P_{\mathcal{M}}$  containing  $\mathbf{1}_{M_1}, \mathbf{1}_{M_2}, \mathbf{1}_{M_3}$ .

## Idea:

- ▶ Show that  $y$  is almost on the face  $F$ :
  - Let  $\bar{y} := \mathbf{1}_{M_1} + \mathbf{1}_{M_2} - y \Rightarrow \bar{y}$  is also an almost matching.
  - Hence,  $\frac{1}{2}(y + \bar{y}) = \frac{1}{2}(\mathbf{1}_{M_1} + \mathbf{1}_{M_2}) \in F$ .
  - $y$  and  $\bar{y}$  are almost in  $P_{\mathcal{M}} \rightarrow y, \bar{y}$  almost on  $F$ .
- ▶  $\mathcal{L}(y) \approx \mathcal{L}(x_{1,2})$ .
- ▶ Since  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2 \rightarrow w(y) \approx w(x_{1,2})$ .

# What about the objective value? (sketch of reasoning)

Almost matching  $y$  obtained by merging  $M_1$  and  $M_2$  has weight close to  $x_{1,2}$ .

- ▶ Recall that by construction:  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2$ .
- ▶ Lagrangian weight:  $\mathcal{L}(x) := w(x) + \lambda_1^* \ell_1(x) + \lambda_2^* \ell_2(x) \quad \forall x \in [0, 1]^E$ .
  - Interpreted as a vector,  $\mathcal{L} \perp F$ , where  $F$  is the 2-dimensional face of  $P_{\mathcal{M}}$  containing  $\mathbf{1}_{M_1}, \mathbf{1}_{M_2}, \mathbf{1}_{M_3}$ .

## Idea:

- ▶ Show that  $y$  is almost on the face  $F$ :
  - Let  $\bar{y} := \mathbf{1}_{M_1} + \mathbf{1}_{M_2} - y \Rightarrow \bar{y}$  is also an almost matching.
  - Hence,  $\frac{1}{2}(y + \bar{y}) = \frac{1}{2}(\mathbf{1}_{M_1} + \mathbf{1}_{M_2}) \in F$ .
  - $y$  and  $\bar{y}$  are almost in  $P_{\mathcal{M}} \rightarrow y, \bar{y}$  almost on  $F$ .
- ▶  $\mathcal{L}(y) \approx \mathcal{L}(x_{1,2})$ .
- ▶ Since  $\ell_i(y) = \ell_i(x_{1,2})$  for  $i = 1, 2 \rightarrow w(y) \approx w(x_{1,2})$ .

## Nice . . . , and how to prove it properly? (the real proof)

### Claim

The matching  $M_{1,2}$  obtained from the almost matching  $y$  (merge of  $M_1$  and  $M_2$ ) satisfies:  $w(M_{1,2}) \geq w(x_{1,2}) - 2w_{\max}$ .

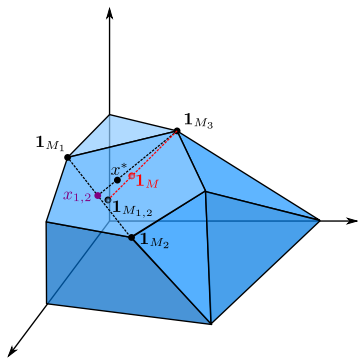
### Proof.

- ▶  $y + \bar{y} = \mathbf{1}_{M_1} + \mathbf{1}_{M_2} \rightarrow \mathcal{L}(y) + \mathcal{L}(\bar{y}) = 2\mathcal{L}(x^*)$
- ▶ Removing  $\bar{r} \leq 2$  units from  $\bar{y}$  leads to a feasible solution.  
 $\Rightarrow \mathcal{L}(\bar{y}) - \bar{r}w_{\max} \leq \mathcal{L}(x^*)$
- ▶ Hence  $\mathcal{L}(y) \geq \mathcal{L}(x_{1,2}) - \bar{r}w_{\max}$
- ▶ Since  $l_i(y) = l_i(x_{1,2}) \rightarrow w(y) \geq w(x_{1,2}) - \bar{r}w_{\max}$
- ▶ Matching  $M_{1,2}$  is obtained by removing at most  $r \leq 2$  units from  $y$ .  
 $\Rightarrow w(M_{1,2}) \geq w(y) - rw_{\max} \geq w(x_{1,2}) - (r + \bar{r})w_{\max}$
- ▶ Result follows by observing that  $r + \bar{r} \leq 2$ .



## In summary

1. Merging  $M_1$  and  $M_2$  by edge flips we get an almost matching  $y$  satisfying:
  - ▶  $l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .
2. Removing at most 2 units of  $y$  we get a matching  $M_{1,2}$  satisfying:
  - ▶  $l_i(M_{1,2}) \leq l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .
  - ▶  $w(M_{1,2}) \geq w(x_{1,2}) - 2w_{\max}$ .
3. Applying the same merging procedure to  $M_{1,2}$  and  $M_3$  we obtain a matching  $M$  satisfying:
  - ▶  $l_i(M) \leq l_i(x^*) \leq B_i$  for  $i = 1, 2$ .
  - ▶  $w(M) \geq w(x^*) - 6w_{\max}$ .

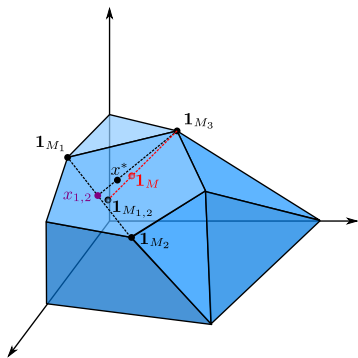


→ The “ $-6w_{\max}$ ” after the second merging is due to a slight error-amplification.

→ The algo is transformed into a PTAS by a preprocessing phase using filtering.

## In summary

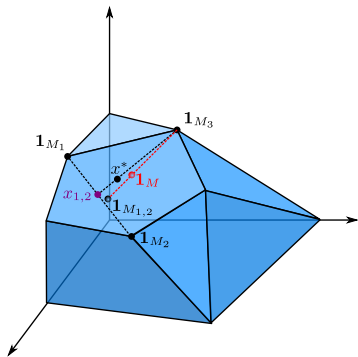
1. Merging  $M_1$  and  $M_2$  by edge flips we get an almost matching  $y$  satisfying:
  - ▶  $l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .
2. Removing at most 2 units of  $y$  we get a matching  $M_{1,2}$  satisfying:
  - ▶  $l_i(M_{1,2}) \leq l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .
  - ▶  $w(M_{1,2}) \geq w(x_{1,2}) - 2w_{\max}$ .
3. Applying the same merging procedure to  $M_{1,2}$  and  $M_3$  we obtain a matching  $M$  satisfying:
  - ▶  $l_i(M) \leq l_i(x^*) \leq B_i$  for  $i = 1, 2$ .
  - ▶  $w(M) \geq w(x^*) - 6w_{\max}$ .



- The “ $-6w_{\max}$ ” after the second merging is due to a **slight error-amplification**.
- The algo is transformed into a PTAS by a **preprocessing phase using filtering**.

## In summary

1. Merging  $M_1$  and  $M_2$  by edge flips we get an almost matching  $y$  satisfying:
  - ▶  $l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .
2. Removing at most 2 units of  $y$  we get a matching  $M_{1,2}$  satisfying:
  - ▶  $l_i(M_{1,2}) \leq l_i(y) = l_i(x_{1,2})$  for  $i = 1, 2$ .
  - ▶  $w(M_{1,2}) \geq w(x_{1,2}) - 2w_{\max}$ .
3. Applying the same merging procedure to  $M_{1,2}$  and  $M_3$  we obtain a matching  $M$  satisfying:
  - ▶  $l_i(M) \leq l_i(x^*) \leq B_i$  for  $i = 1, 2$ .
  - ▶  $w(M) \geq w(x^*) - 6w_{\max}$ .



- The “ $-6w_{\max}$ ” after the second merging is due to a **slight error-amplification**.
- The algo is transformed into a PTAS by a **preprocessing phase using filtering**.

# How to extend to any constant number of budgets?

- ▶ Only steps that needs to be generalized: **merging of matchings**.
- ▶ It is ok to **flip** any **constant # of subintervals** of the ordered edges.

## Conjecture: Generalized Necklace Splitting Problem

For  $k \in \mathbb{N}$ ,  $\exists r(k) \in \mathbb{Z}_+$  such that for every curve  $f : [0, 1] \rightarrow \mathbb{R}^k$  and  $\mu \in [0, 1]$ ,  $\exists r(k)$  disjoint intervals  $[a_1, b_1], \dots, [a_{r(k)}, b_{r(k)}] \subseteq [0, 1]$  satisfying

$$f(0) + \sum_{i=1}^{r(k)} (f(b_i) - f(a_i)) = \mu f(0) + (1 - \mu)f(1).$$

- ▶ This conjecture can be seen to fit into the context of the necklace splitting problem with two thieves, who want to divide the necklace such that one thief gets exactly a  $\mu$ -fraction of every type of bead (instead of the classical 1/2:1/2 division).

# How to extend to any constant number of budgets?

- ▶ Only steps that needs to be generalized: **merging of matchings**.
- ▶ It is ok to **flip** any **constant # of subintervals** of the ordered edges.

## Conjecture: Generalized Necklace Splitting Problem

For  $k \in \mathbb{N}$ ,  $\exists r(k) \in \mathbb{Z}_+$  such that for every curve  $f : [0, 1] \rightarrow \mathbb{R}^k$  and  $\mu \in [0, 1]$ ,  
 $\exists r(k)$  disjoint intervals  $[a_1, b_1], \dots, [a_{r(k)}, b_{r(k)}] \subseteq [0, 1]$  satisfying

$$f(0) + \sum_{i=1}^{r(k)} (f(b_i) - f(a_i)) = \mu f(0) + (1 - \mu) f(1).$$

- ▶ This conjecture can be seen to fit into the context of the necklace splitting problem with two thieves, who want to divide the necklace such that one thief gets exactly a  $\mu$ -fraction of every type of bead (instead of the classical 1/2:1/2 division).

# How to extend to any constant number of budgets?

- ▶ Only steps that needs to be generalized: **merging of matchings**.
- ▶ It is ok to **flip** any **constant # of subintervals** of the ordered edges.

## Conjecture: Generalized Necklace Splitting Problem

For  $k \in \mathbb{N}$ ,  $\exists r(k) \in \mathbb{Z}_+$  such that for every curve  $f : [0, 1] \rightarrow \mathbb{R}^k$  and  $\mu \in [0, 1]$ ,  
 $\exists r(k)$  disjoint intervals  $[a_1, b_1], \dots, [a_{r(k)}, b_{r(k)}] \subseteq [0, 1]$  satisfying

$$f(0) + \sum_{i=1}^{r(k)} (f(b_i) - f(a_i)) = \mu f(0) + (1 - \mu) f(1).$$

- ▶ This conjecture can be seen to fit into the context of the necklace splitting problem with two thieves, who want to divide the necklace such that one thief gets exactly a  $\mu$ -fraction of every type of bead (instead of the classical 1/2:1/2 division).

# Outline

## ① Motivation and previous results

- Motivation
- Our results

## ② Feasibilization

## ③ A PTAS for 2-BUDGETED MATCHINGS

## ④ A PTAS for $K$ -BUDGETED MATROID INDEPENDENT SET

## ⑤ Conclusion

# Almost integral low-dimensional faces

## Theorem

Let  $F$  be a face of dimension  $d$  of a matroid polytope. Then any  $x \in F$  has at most  $2d$  non-integral components, summing up to at most  $d$ .

## Exploiting this fact to get a PTAS

- ▶ Matroid:  $M = (S, \mathcal{I})$ , Matroid polytope:  $P_M = \text{conv}(\{1_I \mid I \in \mathcal{I}\}) \subseteq 2^S$ .
- ▶ Switch to **vector notation**:

$$\begin{aligned} \max_{I \in \mathcal{I}} \quad & w(I) \\ & \ell_i(I) \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\Leftrightarrow$

$$\begin{aligned} \max_{x \text{ vertex of } P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

## Algorithm

1. Solve relaxation:

$$\begin{aligned} \max_{x \in P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\rightarrow$  optimal vertex solution  $x^*$ .

2. Set fractional values of  $x^*$  to 0  $\rightarrow$  return resulting solution  $I \in \mathcal{I}$ .

- ▶  $x^*$  is on  $\leq k$  dimensional face of  $P_M \rightarrow w(I) \geq w(x^*) - k \cdot w_{\max}$
- ▶ Preprocessing phase using **filtering**: Guess  $k/\epsilon$  heaviest elements of opt  $\Rightarrow w_{\max} \leq \frac{\epsilon}{k} OPT \Rightarrow w(I) \geq w(x^*) - k \cdot w_{\max} \geq (1 - \epsilon) OPT$ .

# Almost integral low-dimensional faces

## Theorem

Let  $F$  be a face of dimension  $d$  of a matroid polytope. Then any  $x \in F$  has at most  $2d$  non-integral components, summing up to at most  $d$ .

## Exploiting this fact to get a PTAS

- ▶ Matroid:  $M = (S, \mathcal{I})$ , Matroid polytope:  $P_M = \text{conv}(\{\mathbf{1}_I \mid I \in \mathcal{I}\}) \subseteq 2^S$ .
- ▶ Switch to **vector notation**:

$$\begin{aligned} \max_{I \in \mathcal{I}} \quad & w(I) \\ & \ell_i(I) \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\Leftrightarrow$

$$\begin{aligned} \max_{x \text{ vertex of } P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

## Algorithm

1. Solve relaxation:

$$\begin{aligned} \max_{x \in P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\rightarrow$  optimal vertex solution  $x^*$ .

2. Set fractional values of  $x^*$  to 0  $\rightarrow$  return resulting solution  $I \in \mathcal{I}$ .

- ▶  $x^*$  is on  $\leq k$  dimensional face of  $P_M \rightarrow w(I) \geq w(x^*) - k \cdot w_{\max}$
- ▶ Preprocessing phase using filtering: Guess  $k/\epsilon$  heaviest elements of opt  $\Rightarrow w_{\max} \leq \frac{\epsilon}{k} OPT \Rightarrow w(I) \geq w(x^*) - k \cdot w_{\max} \geq (1 - \epsilon) OPT$ .

# Almost integral low-dimensional faces

## Theorem

Let  $F$  be a face of dimension  $d$  of a matroid polytope. Then any  $x \in F$  has at most  $2d$  non-integral components, summing up to at most  $d$ .

## Exploiting this fact to get a PTAS

- ▶ Matroid:  $M = (S, \mathcal{I})$ , Matroid polytope:  $P_M = \text{conv}(\{\mathbf{1}_I \mid I \in \mathcal{I}\}) \subseteq 2^S$ .
- ▶ Switch to **vector notation**:

$$\begin{aligned} \max_{I \in \mathcal{I}} \quad & w(I) \\ & \ell_i(I) \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\Leftrightarrow$

$$\begin{aligned} \max_{x \text{ vertex of } P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

## Algorithm

1. Solve relaxation:  $\max_{x \in P_M} \begin{aligned} & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned} \rightarrow$  optimal vertex solution  $x^*$ .
2. Set fractional values of  $x^*$  to 0  $\rightarrow$  return resulting solution  $I \in \mathcal{I}$ .

- ▶  $x^*$  is on  $\leq k$  dimensional face of  $P_M \rightarrow w(I) \geq w(x^*) - k \cdot w_{\max}$
- ▶ Preprocessing phase using filtering: Guess  $k/\epsilon$  heaviest elements of opt  $\Rightarrow w_{\max} \leq \frac{\epsilon}{k} OPT \Rightarrow w(I) \geq w(x^*) - k \cdot w_{\max} \geq (1 - \epsilon) OPT$ .

# Almost integral low-dimensional faces

## Theorem

Let  $F$  be a face of dimension  $d$  of a matroid polytope. Then any  $x \in F$  has at most  $2d$  non-integral components, summing up to at most  $d$ .

## Exploiting this fact to get a PTAS

- ▶ Matroid:  $M = (S, \mathcal{I})$ , Matroid polytope:  $P_M = \text{conv}(\{\mathbf{1}_I \mid I \in \mathcal{I}\}) \subseteq 2^S$ .
- ▶ Switch to **vector notation**:

$$\begin{array}{ll} \max_{I \in \mathcal{I}} & w(I) \\ & \ell_i(I) \leq B_i \quad \forall i \in [k] \end{array}$$

$\Leftrightarrow$

$$\begin{array}{ll} \max_{x \text{ vertex of } P_M} & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{array}$$

## Algorithm

1. Solve relaxation:

$$\begin{array}{ll} \max_{x \in P_M} & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{array}$$

$\rightarrow$  optimal vertex solution  $x^*$ .

2. Set fractional values of  $x^*$  to 0  $\rightarrow$  return resulting solution  $I \in \mathcal{I}$ .

- ▶  $x^*$  is on  $\leq k$  dimensional face of  $P_M \rightarrow w(I) \geq w(x^*) - k \cdot w_{\max}$
- ▶ Preprocessing phase using **filtering**: Guess  $k/\epsilon$  heaviest elements of opt  $\Rightarrow w_{\max} \leq \frac{\epsilon}{k} OPT \Rightarrow w(I) \geq w(x^*) - k \cdot w_{\max} \geq (1 - \epsilon) OPT$ .

# Almost integral low-dimensional faces

## Theorem

Let  $F$  be a face of dimension  $d$  of a matroid polytope. Then any  $x \in F$  has at most  $2d$  non-integral components, summing up to at most  $d$ .

## Exploiting this fact to get a PTAS

- ▶ Matroid:  $M = (S, \mathcal{I})$ , Matroid polytope:  $P_M = \text{conv}(\{\mathbf{1}_I \mid I \in \mathcal{I}\}) \subseteq 2^S$ .
- ▶ Switch to **vector notation**:

$$\begin{aligned} \max_{I \in \mathcal{I}} \quad & w(I) \\ & \ell_i(I) \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\Leftrightarrow$

$$\begin{aligned} \max_{x \text{ vertex of } P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

## Algorithm

1. Solve relaxation:

$$\begin{aligned} \max_{x \in P_M} \quad & w^T x \\ & \ell_i^T x \leq B_i \quad \forall i \in [k] \end{aligned}$$

$\rightarrow$  optimal vertex solution  $x^*$ .

2. Set fractional values of  $x^*$  to 0  $\rightarrow$  return resulting solution  $I \in \mathcal{I}$ .

- ▶  $x^*$  is on  $\leq k$  dimensional face of  $P_M \rightarrow w(I) \geq w(x^*) - k \cdot w_{\max}$
- ▶ Preprocessing phase using **filtering**: Guess  $k/\epsilon$  heaviest elements of opt  $\Rightarrow w_{\max} \leq \frac{\epsilon}{k} OPT \Rightarrow w(I) \geq w(x^*) - k \cdot w_{\max} \geq (1 - \epsilon) OPT$ .

# Sketch of proof for face theorem (for the forest polytope)

- ▶  $G = (V, E)$ : undirected graph.
- ▶ Indep. sets = forests, i.e.,  $M = (E, \mathcal{I})$ ,  $\mathcal{I} = \{U \subseteq E \mid U \text{ is a forest}\}$ .

**Proof** (based on combinatorial uncrossing)

- ▶ The forest polytope  $P_G = P_M$  consists of all  $x \in \mathbb{R}^E$  satisfying:

$$\begin{array}{rcl} x(U) & \leq & |V(U)| - c(U) \quad \forall U \subseteq E \\ x & \geq & 0, \end{array}$$

where  $c(U)$ : # connected components of  $(V(U), U)$ .

- ▶ Let  $F$  be a face of  $P_G$  of dimension  $d$ :

$$F = \{x \in P_G \mid x(e) = 0 \quad \forall e \in N, \quad x(U_i) = |V(U_i)| - c(U_i) \quad \forall i \in [p]\},$$

where,  $N \subseteq E$ ,  $U_i \subseteq V$  for  $i \in [p]$ , and  $|N| + p = |E| - d$ .

- ▶ If  $N \neq \emptyset$ , we can reduce the dimension of the problem:
  - ▶ Remove  $N$  from  $G = (V, E) \rightarrow G' = (V, E' = E \setminus N)$ .
  - ▶ Consider  $F' = \{x \in P_{G'} \mid x(U_i) = |V(U_i)| - c(U_i) \quad \forall i \in [p]\}$
  - ▶  $F'$  is the projection of  $F$  onto  $E'$ .
    - $\rightarrow$  Sufficient to prove that any point on  $F'$  has at most  $2d$  fractional components summing up to at most  $d$ .
- ▶ We assume  $N = \emptyset$ .

# Continuation of Proof

- ▶  $F$  is a face of  $P_G$  of dimension  $d$  given by:

$$F = \{x \in P_G \mid x(U_i) = |V(U_i)| - c(U_i) \quad \forall i \in [p]\},$$

where  $p = |E| - d$ .

- ▶ Using combinatorial uncrossing we can assume

$$\emptyset \subsetneq U_1 \subsetneq U_2 \subsetneq \cdots \subsetneq U_p \subsetneq E$$

→ A chain increasing each step by one edge with  $d$  missing sets.

- ▶ Consider the disjoint sets:  $A_i = U_{i+1} \setminus U_i$  for  $i \in [p]$  ( $A_0 := \emptyset$ ).
- ▶ For  $x \in F$ ,

$$x(A_i) = x(U_{i+1}) - x(U_i) = |V(U_{i+1})| - c(U_{i+1}) - |V(U_i)| + c(U_i).$$

- ▶ If  $A_i = \{e\} \Rightarrow x(e) \in \{0, 1\} \quad \forall x \in F$ .

→ A point  $x \in F$  has fractional coordinates in  $A_i$  only if  $|A_i| > 2$ .

- ▶ Consider  $J = \{i \in [p] \mid |A_i| \geq 2\}$ .

- ▶ # of fractional components of  $x \in F \leq \sum_{i: |A_i| \geq 2} |A_i| \leq 2d$ .

□ first part of theorem

## Continuation of Proof

- ▶  $x(A_i)$  integral  $\Rightarrow$  either  $\begin{cases} x(A_i) = |A_i| \\ x(A_i) \leq |A_i| - 1. \end{cases} \Rightarrow x(e) = 1 \forall e \in A_i,$

→ In any case the sum of the fractional components of  $x \in F$  in  $A_i$  is bounded by  $|A_i| - 1$ .

- ▶ Total sum of fractional components of  $x \in F$  is bounded by:

$$\sum_{i \in J} (|A_i| - 1) = \left( \sum_{i \in J} |A_i| \right) - |J|.$$

- ▶ It remains to observe that  $\sum_{i \in J} |A_i| = d + |J|$ .

□ second part of theorem

# Outline

## ① Motivation and previous results

- Motivation
- Our results

## ② Feasibilization

## ③ A PTAS for 2-BUDGETED MATCHINGS

## ④ A PTAS for K-BUDGETED MATROID INDEPENDENT SET

## ⑤ Conclusion

# Conclusions

- ▶ **Feasibilization** is a simple but very useful mechanism to transform multi-criteria algorithms into multi-budgeted algorithm.
  - ▶ **Low-dimensional faces** of matroid polytopes are **almost integral** → PTAS for  $k$ -BUDGETED MATROID INDEPENDENT SET.
  - ▶ **PTAS for 2-BUDGETED MATCHING.**
- 
- ▶ Is there an **FPTAS** for 1-BUDGETED MATCHING or 1-BUDGETED SPANNING TREE?
  - ▶ How to **extend the general framework presented for 2-BUDGETED MATCHING** to other problems?
  - ▶ **Generalized Necklace Splitting Conjecture.**

## References I

- A. Berger, V. Bonifaci, F. Grandoni, and G. Schäfer. Budgeted matching and budgeted matroid intersection via the gasoline puzzle. *Mathematical Programming, Series A*, 2009.
- P.M. Camerini, G. Galbiati, and F. Maffioli. Random pseudo-polynomial algorithms for exact matroid problems. *Journal of Algorithms*, 13(2): 258–273, 1992. ISSN 0196-6774. doi: DOI:10.1016/0196-6774(92)90018-8. URL <http://www.sciencedirect.com/science/article/B6WH3-4D7JNMD-8R/2/862564aa1224c875053bf6ee3805de83>.
- R. Hassin. Approximation schemes for the restricted shortest path problem. *Math. Oper. Res.*, 17(1):36–42, 1992. ISSN 0364-765X.
- D. H. Lorenz and D. R. Raz. A simple efficient approximation scheme for the restricted shortest path problem. *Operations research letters*, 28 (5):213–219, 2001.

## References II

- K. Mulmuley, U.V. Vazirani, and V.V. Vazirani. Matching is as easy as matrix inversion. In *STOC '87: Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 345–354, New York, NY, USA, 1987. ACM. ISBN 0-89791-221-7. doi: <http://doi.acm.org/10.1145/28395.383347>.
- C. H. Papadimitriou and M. Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 86, Washington, DC, USA, 2000. IEEE Computer Society. ISBN 0-7695-0850-2.
- R. Ravi and M. X. Goemans. The constrained minimum spanning tree problem. In *Algorithm Theory – SWAT'96*, volume 1097 of *Lecture Notes in Computer Science*, pages 66–75. Springer Berlin / Heidelberg, 1996. ISBN 978-3-540-61422-7.
- A. Warburton. Approximation of pareto optima in multiple-objective, shortest-path problems. *Operations Research*, 35(1):70–79, 1987.