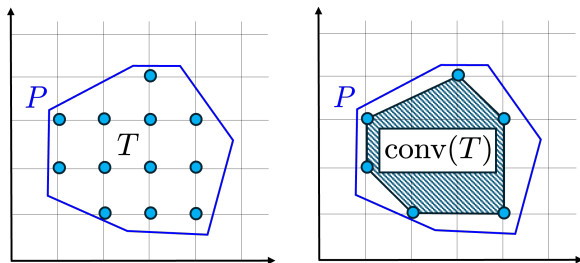


# Lecture 8

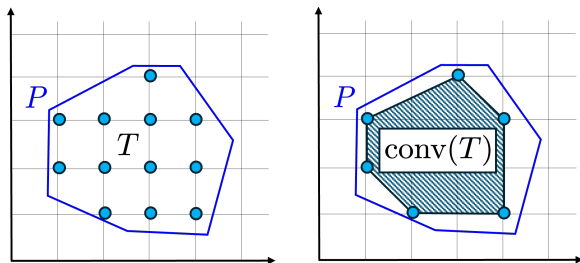
November 19, 2024

# Recall from Monday: Strength of IP Formulation



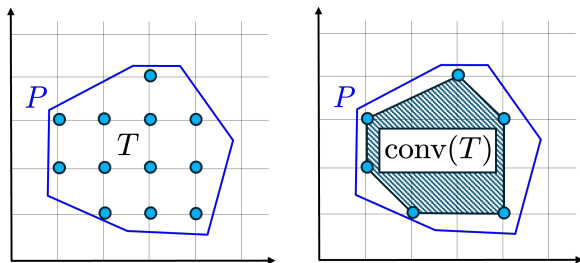
- Consider an IP with bounded feasible set
  - $T$  : all feasible points to the IP
  - $P$  : feasible set for LP relaxation to IP
  - $\text{conv}(T)$  : the convex hull of  $T$  (a polyhedral set)
  - Always have:  $T \subseteq \text{conv}(T) \subseteq P$ .

# Recall from Monday: Strength of IP Formulation



- Consider an IP with bounded feasible set
  - $T$  : all feasible points to the IP
  - $P$  : feasible set for LP relaxation to IP
  - $\text{conv}(T)$  : the convex hull of  $T$  (a polyhedral set)
  - Always have:  $T \subseteq \text{conv}(T) \subseteq P$ .
- **Ideal IP formulation:**  $P = \text{conv}(T)$

# Recall from Monday: Strength of IP Formulation



- Consider an IP with bounded feasible set
  - $T$  : all feasible points to the IP
  - $P$  : feasible set for LP relaxation to IP
  - $\text{conv}(T)$  : the convex hull of  $T$  (a polyhedral set)
  - Always have:  $T \subseteq \text{conv}(T) \subseteq P$ .

- **Ideal IP formulation:**  $P = \text{conv}(T)$

1. Discuss a few **ideal formulations** :  $P = \text{conv}(T)$
2. Discuss how to **improve** formulations by adding **cuts**
3. Discuss **algorithms/solution approaches**

# (Total) Unimodularity : Ideal Formulations

## Definition

1.  $A \in \mathbb{Z}^{m \times n}$  of full row rank is **unimodular** if  $\det(A_B) \in \{1, -1\}$  for every basis  $B$ .
2.  $A \in \{-1, 0, 1\}^{m \times n}$  is **totally unimodular** if the determinant of each square submatrix of  $A$  is 0, 1, or -1.

# (Total) Unimodularity : Ideal Formulations

## Definition

1.  $A \in \mathbb{Z}^{m \times n}$  of full row rank is **unimodular** if  $\det(A_B) \in \{1, -1\}$  for every basis  $B$ .
2.  $A \in \{-1, 0, 1\}^{m \times n}$  is **totally unimodular** if the determinant of each square submatrix of  $A$  is 0, 1, or -1.

## Theorem

1.  $A \in \mathbb{Z}^{m \times n}$  **unimodular** if and only if  $P(b) = \{x \in \mathbb{R}_+^n \mid Ax = b\}$  **is integral** for all  $b \in \mathbb{Z}^m$  with  $P(b) \neq \emptyset$ .
2.  $A$  is **totally unimodular** if and only if  $P(b) = \{x \in \mathbb{R}_+^n \mid Ax \leq b\}$  **is integral** for all  $b \in \mathbb{Z}^m$  with  $P(b) \neq \emptyset$ .

# Checking for Total Unimodularity

Proposition (Refreshed; **necessary**, but **not sufficient**.)

*A matrix  $A \in \{0, 1, -1\}^{m \times n}$  is totally unimodular if any of the following holds:*

# Checking for Total Unimodularity

Proposition (Refreshed; **necessary**, but **not sufficient**.)

A matrix  $A \in \{0, 1, -1\}^{m \times n}$  is totally unimodular if any of the following holds:

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.



# Checking for Total Unimodularity

Proposition (Refreshed; **necessary**, but **not sufficient**.)

A matrix  $A \in \{0, 1, -1\}^{m \times n}$  is totally unimodular if any of the following holds:

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.
4. Every subset  $R$  of rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that  $\sum_{i \in R_1} a_i - \sum_{i \in R_2} a_i \in \{0, +1, -1\}$ . (By 1, a similar result holds for **columns** of  $A$ .)

# Checking for Total Unimodularity

Proposition (Refreshed; **necessary**, but **not sufficient**.)

A matrix  $A \in \{0, 1, -1\}^{m \times n}$  is totally unimodular if any of the following holds:

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.
4. Every subset  $R$  of rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that  $\sum_{i \in R_1} a_i - \sum_{i \in R_2} a_i \in \{0, +1, -1\}$ . (By 1, a similar result holds for **columns** of  $A$ .)
5. Each column of  $A$  contains at most two nonzero elements and the rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that the two nonzero entries in a column are in the same  $R_i$  if they have different signs and are in different  $R_i$  if they have the same sign.

# Checking for Total Unimodularity

Proposition (Refreshed; **necessary**, but **not sufficient**.)

A matrix  $A \in \{0, 1, -1\}^{m \times n}$  is totally unimodular if any of the following holds:

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.
4. Every subset  $R$  of rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that  $\sum_{i \in R_1} a_i - \sum_{i \in R_2} a_i \in \{0, +1, -1\}$ . (By 1, a similar result holds for **columns** of  $A$ .)
5. Each column of  $A$  contains at most two nonzero elements and the rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that the two nonzero entries in a column are in the same  $R_i$  if they have different signs and are in different  $R_i$  if they have the same sign.
6.  $A$  contains no more than one  $+1$  and one  $-1$  in each column.

# Checking for Total Unimodularity

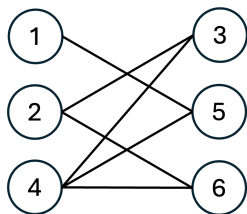
Proposition (Refreshed; **necessary**, but **not sufficient**.)

A matrix  $A \in \{0, 1, -1\}^{m \times n}$  is totally unimodular if any of the following holds:

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.
4. Every subset  $R$  of rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that  $\sum_{i \in R_1} a_i - \sum_{i \in R_2} a_i \in \{0, +1, -1\}$ . (By 1, a similar result holds for **columns** of  $A$ .)
5. Each column of  $A$  contains at most two nonzero elements and the rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that the two nonzero entries in a column are in the same  $R_i$  if they have different signs and are in different  $R_i$  if they have the same sign.
6.  $A$  contains no more than one  $+1$  and one  $-1$  in each column.
7.  $A$  has the **consecutive ones** property: for every column  $j$ ,  $a_{sj} = a_{tj} = 1$  implies  $a_{ij} = 1$  for  $s \leq i \leq t$ .

# Examples of TU Matrices #1

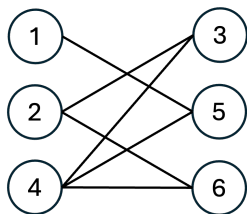
- $G = (\mathcal{N}, \mathcal{E})$  undirected graph
- $A \in \{0, 1\}^{|\mathcal{N}| \times |\mathcal{E}|}$  is the node-edge incidence matrix of  $G$   
 $A_{i,e} = 1$  if and only if  $i \in e$



	$\{1, 5\}$	$\{2, 3\}$	$\{2, 6\}$	$\{4, 3\}$	$\{4, 5\}$	$\{4, 6\}$
1	1	0	0	0	0	0
2	0	1	1	0	0	0
3	0	1	0	1	0	0
4	0	0	0	1	1	1
5	1	0	0	0	1	0
6	0	0	1	0	0	1

# Examples of TU Matrices #1

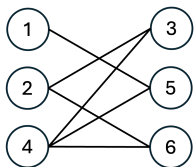
- $G = (\mathcal{N}, \mathcal{E})$  undirected graph
- $A \in \{0, 1\}^{|\mathcal{N}| \times |\mathcal{E}|}$  is the node-edge incidence matrix of  $G$   
 $A_{i,e} = 1$  if and only if  $i \in e$



	$\{1, 5\}$	$\{2, 3\}$	$\{2, 6\}$	$\{4, 3\}$	$\{4, 5\}$	$\{4, 6\}$
1	1	0	0	0	0	0
2	0	1	1	0	0	0
3	0	1	0	1	0	0
4	0	0	0	1	1	1
5	1	0	0	0	1	0
6	0	0	1	0	0	1

- $A$  is **TU** if and only if  $G$  is **bipartite**  
Can partition  $\mathcal{N}$  into  $S$  and  $T$  so that every  $e \in E$  is  $e = (s, t)$  with  $s \in S, t \in T$
- Bipartite matching problems have integral LP relaxations...

# Prove #1: $G$ **bipartite** implies $A$ is TU



	$\{1, 5\}$	$\{2, 3\}$	$\{2, 6\}$	$\{4, 3\}$	$\{4, 5\}$	$\{4, 6\}$
1	1	0	0	0	0	0
2	0	1	1	0	0	0
3	0	1	0	1	0	0
4	0	0	0	1	1	1
5	1	0	0	0	1	0
6	0	0	1	0	0	1

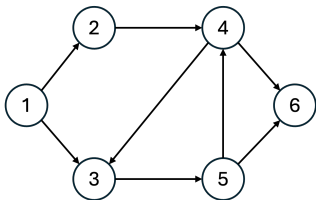
Proposition (Necessary conditions for  $A \in \{-1, 0, 1\}$  to be TU)

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.
4. Every subset  $R$  of rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that  $\sum_{i \in R_1} a_i - \sum_{i \in R_2} a_i \in \{0, +1, -1\}$ . (By 1, a similar result holds for **columns** of  $A$ .)
5. Each column of  $A$  contains at most two nonzero elements and the rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that the two nonzero entries in a column are in the same  $R_i$  if they have different signs and are in different  $R_i$  if they have the same sign.
6.  $A$  contains no more than one  $+1$  and one  $-1$  in each column.
7.  $A$  has the **consecutive ones** property: for every column  $j$ ,  $a_{sj} = a_{tj} = 1$  implies  $a_{ij} = 1$  for  $s \leq i \leq t$ .

## Examples of TU Matrices #2

- $D = (V, A)$  is a **directed graph**
- $M$  is the  $V \times A$  incidence matrix of  $D$

$$M_{v,a} = \begin{cases} 1 & \text{if and only if } a = (\cdot, v) \text{ (arc } a \text{ enters node } v) \\ -1 & \text{if and only if } a = (v, \cdot) \text{ (arc } a \text{ leaves node } v) \\ 0 & \text{otherwise.} \end{cases}$$



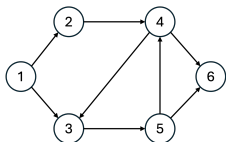
	(1, 2)	(1, 3)	(2, 4)	(4, 3)	(3, 5)	(5, 4)	(4, 6)	(5, 6)
1	-1	-1	0	0	0	0	0	0
2	1	0	-1	0	0	0	0	0
3	0	1	0	1	-1	0	0	0
4	0	0	1	-1	0	1	-1	0
5	0	0	0	0	1	-1	0	-1
6	0	0	0	0	0	0	1	1

- Then  $M$  is **TU**
- **Network flow problems** (e.g., **Prosche Motors**) with integral arc capacities and integral supply/demand have integral LP relaxations



## Prove #2 : Incidence Matrix of Directed Graph is TU

$$M_{v,a} = \begin{cases} 1 & \text{if and only if } a = (\cdot, v) \text{ (arc } a \text{ enters node } v) \\ -1 & \text{if and only if } a = (v, \cdot) \text{ (arc } a \text{ leaves node } v) \\ 0 & \text{otherwise.} \end{cases}$$



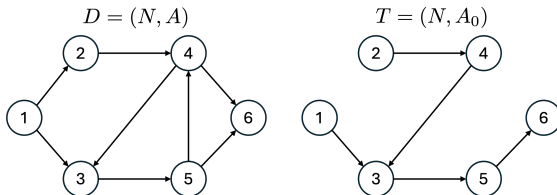
	(1, 2)	(1, 3)	(2, 4)	(4, 3)	(3, 5)	(5, 4)	(4, 6)	(5, 6)
1	-1	-1	0	0	0	0	0	0
2	1	0	-1	0	0	0	0	0
3	0	1	0	1	-1	0	0	0
4	0	0	1	-1	0	1	-1	0
5	0	0	0	0	1	-1	0	-1
6	0	0	0	0	0	0	1	1

### Proposition (Necessary conditions for $A \in \{-1, 0, 1\}$ to be TU)

1.  $A^T$  is totally unimodular.
2.  $-A$  is totally unimodular.
3.  $[A \ -A \ I \ -I]$  is totally unimodular.
4. Every subset  $R$  of rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that  $\sum_{i \in R_1} a_i - \sum_{i \in R_2} a_i \in \{0, +1, -1\}$ .  
(By 1, a similar result holds for **columns** of  $A$ .)
5. Each column of  $A$  contains at most two nonzero elements and the rows of  $A$  can be partitioned into  $R_1$  and  $R_2$  so that the two nonzero entries in a column are in the same  $R_i$  if they have different signs and are in different  $R_i$  if they have the same sign.
6.  $A$  contains no more than one  $+1$  and one  $-1$  in each column.
7.  $A$  has the **consecutive ones** property: for every column  $j$ ,  $a_{sj} = a_{tj} = 1$  implies  $a_{ij} = 1$  for  $s \leq i \leq t$ .

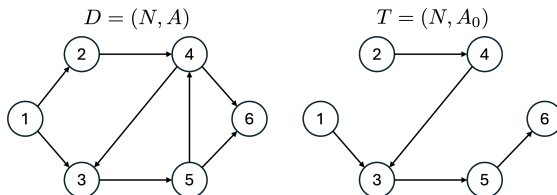
## Examples of TU Matrices #3

- $D = (V, A)$  is a **directed graph**. For  $A_0 \subseteq A$ ,  $T = (V, A_0)$  is a directed tree on  $V$ .



# Examples of TU Matrices #3

- $D = (V, A)$  is a **directed graph**. For  $A_0 \subseteq A$ ,  $T = (V, A_0)$  is a directed tree on  $V$ .



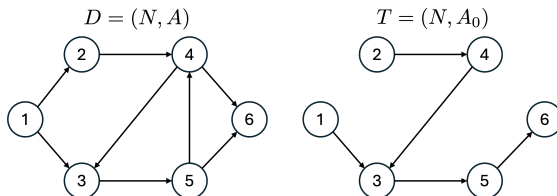
- $M$  is the  $A_0 \times A$  matrix defined as follows: for  $a = (v, w) \in A$  and  $a' \in A_0$ ,

$$M_{a',a} = \begin{cases} +1 & \text{if the unique } v - w \text{ path in } T \text{ passes through } a' \text{ forwardly} \\ -1 & \text{if the unique } v - w \text{ path in } T \text{ passes through } a' \text{ backwardly} \\ 0 & \text{if the unique } v - w \text{ path in } T \text{ does not pass through } a'. \end{cases}$$

	(1, 2)	(1, 3)	(2, 4)	(4, 3)	(3, 5)	(5, 4)	(4, 6)	(5, 6)
(1, 3)	1	1	1	0	0	0	0	0
(2, 4)	-1	0	0	0	0	0	0	0
(4, 3)	-1	0	0	1	0	-1	1	0
(3, 5)	0	0	0	0	1	-1	1	0
(5, 6)	0	0	0	0	0	0	1	1

# Examples of TU Matrices #3

- $D = (V, A)$  is a **directed graph**,  $T = (V, A_0)$  is a directed tree on  $V$



- $M$  is the  $A_0 \times A$  matrix defined as follows: for  $a = (v, w) \in A$  and  $a' \in A_0$ ,

$$M_{a', a} = \begin{cases} +1 & \text{if the unique } v - w \text{ path in } T \text{ passes through } a' \text{ forwardly} \\ -1 & \text{if the unique } v - w \text{ path in } T \text{ passes through } a' \text{ backwardly} \\ 0 & \text{if the unique } v - w \text{ path in } T \text{ does not pass through } a'. \end{cases}$$

- Then  $M$  is **TU**
- All previous examples were **special cases** of this
- Paul Seymour: **all TU matrices** generated from network matrices and **two** other matrices

# Dual Integrality and Submodular Functions

- Alternative conditions based on **LP** duality
- Simple observation: to show that LP relaxation is integral, it suffices to check that the optimal value of any LP with integer cost vector  $c$  is an integer

## Proposition

*$P$  polyhedron with at least one extreme point. Then  $P$  is integral **if and only if** the optimal value  $Z_{LP} := \min\{c^T x \mid x \in P\}$  is an integer for all  $c \in \mathbb{Z}^n$ .*

**Why?**

# Dual Integrality and Submodular Functions

- Alternative conditions based on **LP** duality
- Simple observation: to show that LP relaxation is integral, it suffices to check that the optimal value of any LP with integer cost vector  $c$  is an integer

## Proposition

*$P$  polyhedron with at least one extreme point. Then  $P$  is integral **if and only if** the optimal value  $Z_{LP} := \min\{c^T x \mid x \in P\}$  is an integer for all  $c \in \mathbb{Z}^n$ .*

## Why?

- To show integrality of  $P$ , we **construct an integral dual-optimal** solution (for any  $c \in \mathbb{Z}^n$ )
- Our discussion here is quite specific
  - broader concepts possible related to Total Dual Integrality
  - if interested, see notes for references

# Submodular Functions

## Definition

A function  $f(S)$  defined on subsets  $S$  of a finite set  $N = \{1, \dots, n\}$  is **submodular** if

$$f(S) + f(T) \geq f(S \cap T) + f(S \cup T), \quad \forall S, T \subset N. \quad (1)$$

$f$  is **supermodular** if the reverse inequality holds.

# Submodular Functions

## Definition

A function  $f(S)$  defined on subsets  $S$  of a finite set  $N = \{1, \dots, n\}$  is **submodular** if

$$f(S) + f(T) \geq f(S \cap T) + f(S \cup T), \quad \forall S, T \subset N. \quad (1)$$

$f$  is **supermodular** if the reverse inequality holds.

- More intuition: note that (1) is equivalent to

$$(1) \Leftrightarrow f(S) - f(S \cap T) \geq f(S \cup T) - f(T)$$

*What is the set difference between arguments on the left? And on the right?*



# Submodular Functions

## Definition

A function  $f(S)$  defined on subsets  $S$  of a finite set  $N = \{1, \dots, n\}$  is **submodular** if

$$f(S) + f(T) \geq f(S \cap T) + f(S \cup T), \quad \forall S, T \subset N. \quad (1)$$

$f$  is **supermodular** if the reverse inequality holds.

- More intuition: note that (1) is equivalent to

$$(1) \Leftrightarrow f(S) - f(S \cap T) \geq f(S \cup T) - f(T)$$

*What is the set difference between arguments on the left? And on the right?*

- Left:  $S \setminus (S \cap T) = S \setminus T$ . Right:  $(S \cup T) \setminus T = S \setminus T$ .

# Submodular Functions

## Definition

A function  $f(S)$  defined on subsets  $S$  of a finite set  $N = \{1, \dots, n\}$  is **submodular** if

$$f(S) + f(T) \geq f(S \cap T) + f(S \cup T), \quad \forall S, T \subset N. \quad (1)$$

$f$  is **supermodular** if the reverse inequality holds.

- More intuition: note that (1) is equivalent to

$$(1) \Leftrightarrow f(S) - f(S \cap T) \geq f(S \cup T) - f(T)$$

*What is the set difference between arguments on the left? And on the right?*

- Left:  $S \setminus (S \cap T) = S \setminus T$ . Right:  $(S \cup T) \setminus T = S \setminus T$ .
- **Submodularity**: gains when adding something to a smaller set ( $S \cap T$ ) are larger than when adding it to a larger set ( $T$ )

# Submodular Functions - Equivalent Definitions

## Proposition

A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular** if and only if:

- **Submodular**: “diminishing returns” or “decreasing differences”
  - cost: economies of scale/scope
  - profit: substitutability

Resembles concavity **in economic intuition**, but **not computationally!**  
(submodular functions are more like **convex** functions!)

- **Supermodular** is the opposite
- Subsequently, interested in non-negative and **increasing** submodular functions

$$f(S) \leq f(T), \quad \forall S \subset T \subseteq N.$$

# Submodular Functions - Equivalent Definitions

## Proposition

A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular** if and only if:

(a) For any  $S, T \subseteq N$  such that  $S \subseteq T$  and  $k \notin T$ :

$$f(S \cup \{k\}) - f(S) \geq f(T \cup \{k\}) - f(T).$$

- **Submodular**: “diminishing returns” or “decreasing differences”
  - cost: economies of scale/scope
  - profit: substitutability

Resembles concavity **in economic intuition**, but **not computationally**!  
(submodular functions are more like **convex** functions!)

- **Supermodular** is the opposite
- Subsequently, interested in non-negative and **increasing** submodular functions

$$f(S) \leq f(T), \quad \forall S \subset T \subseteq N.$$

# Submodular Functions - Equivalent Definitions

## Proposition

A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular** if and only if:

(a) For any  $S, T \subseteq N$  such that  $S \subseteq T$  and  $k \notin T$ :

$$f(S \cup \{k\}) - f(S) \geq f(T \cup \{k\}) - f(T).$$

(b) For any  $S \subseteq N$  and any  $j, k$  with  $j, k \notin S$  and  $j \neq k$ :

$$f(S \cup \{j\}) - f(S) \geq f(S \cup \{j, k\}) - f(S \cup \{k\}).$$

# Submodular Functions - Equivalent Definitions

## Proposition

A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular** if and only if:

(a) For any  $S, T \subseteq N$  such that  $S \subseteq T$  and  $k \notin T$ :

$$f(S \cup \{k\}) - f(S) \geq f(T \cup \{k\}) - f(T).$$

(b) For any  $S \subseteq N$  and any  $j, k$  with  $j, k \notin S$  and  $j \neq k$ :

$$f(S \cup \{j\}) - f(S) \geq f(S \cup \{j, k\}) - f(S \cup \{k\}).$$

- **Submodular**: “diminishing returns” or “decreasing differences”
  - cost: economies of scale/scope
  - profit: substitutability

Resembles concavity **in economic intuition**, but **not computationally**!  
(submodular functions are more like **convex** functions!)

- **Supermodular** is the opposite

# Submodular Functions - Equivalent Definitions

## Proposition

A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular** if and only if:

(a) For any  $S, T \subseteq N$  such that  $S \subseteq T$  and  $k \notin T$ :

$$f(S \cup \{k\}) - f(S) \geq f(T \cup \{k\}) - f(T).$$

(b) For any  $S \subseteq N$  and any  $j, k$  with  $j, k \notin S$  and  $j \neq k$ :

$$f(S \cup \{j\}) - f(S) \geq f(S \cup \{j, k\}) - f(S \cup \{k\}).$$

- **Submodular**: “diminishing returns” or “decreasing differences”

- cost: economies of scale/scope
- profit: substitutability

Resembles concavity **in economic intuition**, but **not computationally**!  
(submodular functions are more like **convex** functions!)

- **Supermodular** is the opposite

- Subsequently, interested in non-negative and **increasing** submodular functions

$$f(S) \leq f(T), \quad \forall S \subset T \subseteq N.$$

# Submodular Functions - Examples

Subsequently, consider a ground set  $N = \{1, 2, \dots, n\}$  and  $f : 2^N \rightarrow \mathbb{R}$ .



# Submodular Functions - Examples

Subsequently, consider a ground set  $N = \{1, 2, \dots, n\}$  and  $f : 2^N \rightarrow \mathbb{R}$ .

- **Linear functions.** For  $w \in \mathbb{R}^n$ ,  $f(S) = \sum_{i \in S} w_i$  is both sub- and super-modular.
- **Composition 2.** If  $w \geq 0$  and  $g$  concave, then  $f(S) = g\left(\sum_{i \in S} w_i\right)$  is submodular.
- **Optimal TSP cost on tree graphs.** Consider **undirected tree graph**  $G = (N, E)$  with a positive cost for traversing the edges ( $c_e \geq 0$  for every edge  $e \in E$ ). For every  $S \subseteq N$ , define  $f(S)$  as the optimal (i.e., smallest) cost for a TSP that goes through all the nodes in  $S$ . Then,  $f(S)$  is submodular.
- **Network optimization:** consider directed graph with capacities on edges that constrain how much flow can be transported; if  $f(S)$  is the maximum flow that can be received at a set of sink nodes  $S$ ,  $f(S)$  is submodular.
- **Operations management and economics:** perishable inventory systems, dual sourcing, inventory control problems with trans-shipment, ...
- **Machine learning and computer vision:** data summarization, distillation, data partitioning / clustering, ...

# Main Result

- For a submodular function  $f$ , consider the problem:

$$\begin{aligned} &\text{maximize} \quad \sum_{j=1}^n r_j \cdot x_j \\ &\quad \sum_{j \in S} x_j \leq f(S), \quad \forall S \subseteq N \\ &\quad x \in \mathbb{Z}_+^n. \end{aligned}$$

- $T$ : set of feasible integer solutions
- $P(f)$  the feasible set of the LP relaxation:

$$P(f) = \left\{ x \in \mathbb{R}_+^n \left| \sum_{j \in S} x_j \leq f(S), \quad \forall S \subseteq N \right. \right\}$$

# Main Result

- For a submodular function  $f$ , consider the problem:

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^n r_j \cdot x_j \\ &&& \sum_{j \in S} x_j \leq f(S), \quad \forall S \subseteq N \\ &&& x \in \mathbb{Z}_+^n. \end{aligned}$$

- $T$ : set of feasible integer solutions
- $P(f)$  the feasible set of the LP relaxation:

$$P(f) = \left\{ x \in \mathbb{R}_+^n \left| \sum_{j \in S} x_j \leq f(S), \quad \forall S \subseteq N \right. \right\}$$

## Theorem

*If  $f$  is submodular, increasing, integer valued, and  $f(\emptyset) = 0$ , then*

$$P(f) = \text{conv}(T).$$

# Main Result - Proof

**To show:**  $f$  is submodular, increasing, integer-valued,  $f(\emptyset) = 0$ , then  $P(f) = \text{conv}(T)$ .

**Proof sketch.** Consider the linear relaxation and its dual:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^n r_j x_j \\ & \sum_{j \in S} x_j \leq f(S), \quad S \subset N, \\ & x_j \geq 0, \quad j \in N \end{aligned}$$

# Main Result - Proof

**To show:**  $f$  is submodular, increasing, integer-valued,  $f(\emptyset) = 0$ , then  $P(f) = \text{conv}(T)$ .

**Proof sketch.** Consider the linear relaxation and its dual:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^n r_j x_j \\ & \sum_{j \in S} x_j \leq f(S), \quad S \subset N, \\ & x_j \geq 0, \quad j \in N \end{aligned}$$

# Main Result - Proof

**To show:**  $f$  is submodular, increasing, integer-valued,  $f(\emptyset) = 0$ , then  $P(f) = \text{conv}(T)$ .

**Proof sketch.** Consider the linear relaxation and its dual:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^n r_j x_j \\ & \sum_{j \in S} x_j \leq f(S), \quad S \subset N, \\ & x_j \geq 0, \quad j \in N \end{aligned}$$

- Key idea: construct feasible solutions for both, with equal value
- Key intuition: a **greedy** solution is optimal in the primal!

# Main Result - Proof

**To show:**  $f$  is submodular, increasing, integer-valued,  $f(\emptyset) = 0$ , then  $P(f) = \text{conv}(T)$ .

**Proof sketch.** Consider the linear relaxation and its dual:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^n r_j x_j \\ & \sum_{j \in S} x_j \leq f(S), \quad S \subset N, \\ & x_j \geq 0, \quad j \in N \end{aligned}$$

- Key idea: construct feasible solutions for both, with equal value
- **Key intuition: a greedy solution is optimal in the primal!**
- Suppose  $r_1 \geq r_2 \geq \dots \geq r_k > 0 \geq r_{k+1} \geq \dots \geq r_n$ .
- Let  $S^0 = \emptyset$  and  $S^j = \{1, \dots, j\}$  for  $j \in N$ .

# Main Result - Proof

**To show:**  $f$  is submodular, increasing, integer-valued,  $f(\emptyset) = 0$ , then  $P(f) = \text{conv}(T)$ .

**Proof sketch.** Consider the linear relaxation and its dual:

$$\begin{aligned} \text{maximize} \quad & \sum_{j=1}^n r_j x_j \\ & \sum_{j \in S} x_j \leq f(S), \quad S \subset N, \\ & x_j \geq 0, \quad j \in N \end{aligned}$$

- Key idea: construct feasible solutions for both, with equal value
- **Key intuition: a greedy solution is optimal in the primal!**
- Suppose  $r_1 \geq r_2 \geq \dots \geq r_k > 0 \geq r_{k+1} \geq \dots \geq r_n$ .
- Let  $S^0 = \emptyset$  and  $S^j = \{1, \dots, j\}$  for  $j \in N$ .
- Prove that the following  $x$  and  $y$  are optimal for the primal and dual, respectively.

$$x_j = \begin{cases} f(S^j) - f(S^{j-1}), & 1 \leq j \leq k, \\ 0, & j > k. \end{cases} \quad y_S = \begin{cases} r_j - r_{j+1}, & S = S^j, \quad 1 \leq j < k, \\ r_k, & S = S^k, \\ 0, & \text{otherwise.} \end{cases}$$



# From Discrete to Continuous: The Lovász Extension

- Submodular functions are inherently **discrete**:  $f : 2^N \rightarrow \mathbb{R}$ .
- To connect with convex optimization, we extend  $f$  to the **hypercube**  $[0, 1]^n$ .

# From Discrete to Continuous: The Lovász Extension

- Submodular functions are inherently **discrete**:  $f : 2^N \rightarrow \mathbb{R}$ .
- To connect with convex optimization, we extend  $f$  to the **hypercube**  $[0, 1]^n$ .
- Given  $x \in [0, 1]^n$  and a permutation  $\pi$  that sorts coordinates  $x_{\pi(1)} \geq x_{\pi(2)} \geq \dots \geq x_{\pi(n)}$ , define:

$$\hat{f}(x) = \sum_{k=1}^n x_{\pi(k)} (f(S_k) - f(S_{k-1})), \quad S_k = \{\pi(1), \dots, \pi(k)\}.$$

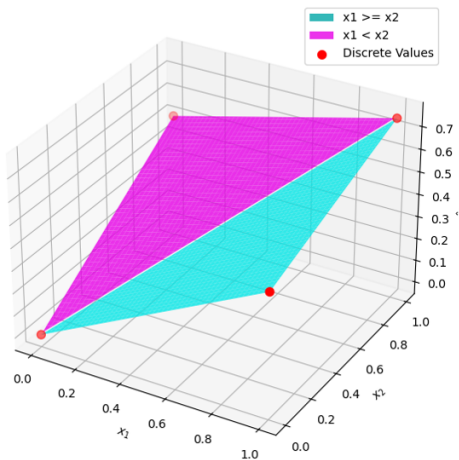
- $\hat{f}$  is the **Lovász extension** of  $f$  — a piecewise linear interpolation of  $f$ 's values over the vertices of  $[0, 1]^n$ .

# Geometry of the Lovász Extension on $[0, 1]^2$

For  $N = \{1, 2\}$  with

$$f(S) = \begin{cases} 0, & S = \emptyset, \\ \frac{1}{2}, & S = \{1\} \text{ or } S = \{2\}, \\ \frac{3}{4}, & S = \{1, 2\}. \end{cases}$$

Lovász extension of a submodular function on  $[0, 1]^2$



# Submodularity & Convexity: The Bridge

## Theorem

*Key Equivalence (Lovász 1983)* A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular**

$\iff$  its Lovász extension  $\hat{f}(x)$  is convex on  $[0, 1]^n$ .

# Submodularity & Convexity: The Bridge

## Theorem

*Key Equivalence (Lovász 1983)* A set function  $f : 2^N \rightarrow \mathbb{R}$  is **submodular**

$\iff$  its Lovász extension  $\hat{f}(x)$  is convex on  $[0, 1]^n$ .

- Submodularity  $\leftrightarrow$  *discrete convexity*.
- Supermodularity  $\leftrightarrow$  *discrete concavity*.
- $\hat{f}$  is piecewise linear with gradients corresponding to vertices of the **base polyhedron**

$$B(f) = \{y \in \mathbb{R}^n : y(S) \leq f(S) \ \forall S \subseteq N, \ y(N) = f(N)\}.$$

- Minimizing  $f$  over  $2^V$  is equivalent to minimizing  $\hat{f}$  over  $[0, 1]^n$ ; the minimum is always achieved at a binary vector.

# Optimization via the Lovász Extension

## Submodular **Minimization**

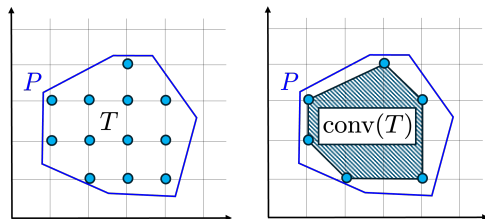
- $\min_{S \subseteq N} f(S) = \min_{x \in [0,1]^n} \hat{f}(x)$ .
- $\hat{f}$  convex  $\Rightarrow$  solvable by convex optimization.
- Algorithms:
  - Iwata–Fleischer–Fujishige (IFF)
  - Schrijver’s combinatorial method
  - Subgradient or cutting-plane over  $B(f)$

## Submodular **Maximization**

- NP-hard in general (non-convex counterpart).
- Continuous relaxations (multilinear extension) enable approximations.
- Greedy algorithms achieve:  
 $1 - \frac{1}{e}$  (monotone),  $\frac{1}{2}$  (non-monotone).

**Takeaway:** The Lovász extension unifies discrete and convex worlds—enabling exact minimization and principled relaxations for maximization.

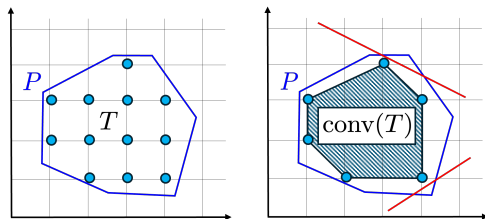
# Improving LP Relaxations With Cuts



- **Recall:**  $T$  are feasible points to an IP,  $\text{conv}(T)$  is their convex hull
- $P$  is the feasible region of an LP relaxation to the IP
- Typically, the formulation is **not ideal**:

$$\text{conv}(T) \subsetneq P$$

# Improving LP Relaxations With Cuts



- **Recall:**  $T$  are feasible points to an IP,  $\text{conv}(T)$  is their convex hull
- $P$  is the feasible region of an LP relaxation to the IP
- Typically, the formulation is **not ideal**:

$$\text{conv}(T) \subsetneq P$$

- How to **improve it by generating valid cuts**?
  - Linear inequalities **satisfied by  $T$  and  $\text{conv}(T)$ , but not by  $P$**



# Improving LP Relaxations With Cuts

- **Setup:**  $A, b, c$  with rational entries and the IP:

$$\text{minimize} \{ c^T x : Ax = b, x \geq 0, x \in \mathbb{Z}^n \}$$

- If  $x^* = [x_B^*; x_N^*]$  be a b.f.s. for the LP relaxation. Then we have:

$$A_B x_B^* + A_N x_N^* = b \quad \Leftrightarrow \quad x_B^* + A_B^{-1} A_N x_N^* = A_B^{-1} b$$

- Consider an equality in which the right-hand-side is **fractional**

# Improving LP Relaxations With Cuts

- **Setup:**  $A, b, c$  with rational entries and the IP:

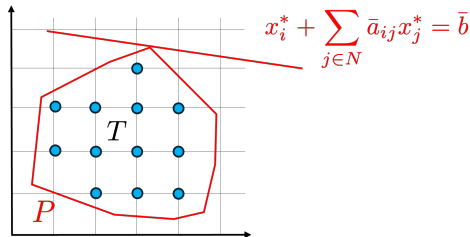
$$\text{minimize} \{ c^T x : Ax = b, x \geq 0, x \in \mathbb{Z}^n \}$$

- If  $x^* = [x_B^*; x_N^*]$  be a b.f.s. for the LP relaxation. Then we have:

$$A_B x_B^* + A_N x_N^* = b \quad \Leftrightarrow \quad x_B^* + A_B^{-1} A_N x_N^* = A_B^{-1} b$$

- Consider an equality in which the right-hand-side is **fractional**

$$x_i^* + \sum_{j \in N} \bar{a}_{ij} x_j^* = \bar{b}$$



# Improving LP Relaxations With Cuts

- **Setup:**  $A, b, c$  with rational entries and the IP:

$$\text{minimize} \{ c^T x : Ax = b, x \geq 0, x \in \mathbb{Z}^n \}$$

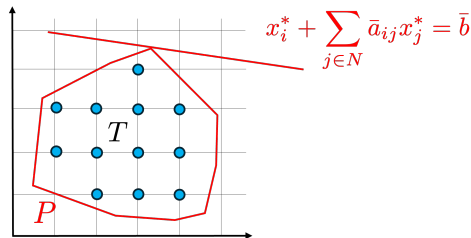
- If  $x^* = [x_B^*; x_N^*]$  be a b.f.s. for the LP relaxation. Then we have:

$$A_B x_B^* + A_N x_N^* = b \quad \Leftrightarrow \quad x_B^* + A_B^{-1} A_N x_N^* = A_B^{-1} b$$

- Consider an equality in which the right-hand-side is **fractional**

$$x_i^* + \sum_{j \in N} \bar{a}_{ij} x_j^* = \bar{b}$$

$$\forall x \in T \Rightarrow x \geq 0 \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \bar{b}$$



# Improving LP Relaxations With Cuts

- **Setup:**  $A, b, c$  with rational entries and the IP:

$$\text{minimize} \{ c^T x : Ax = b, x \geq 0, x \in \mathbb{Z}^n \}$$

- If  $x^* = [x_B^*; x_N^*]$  be a b.f.s. for the LP relaxation. Then we have:

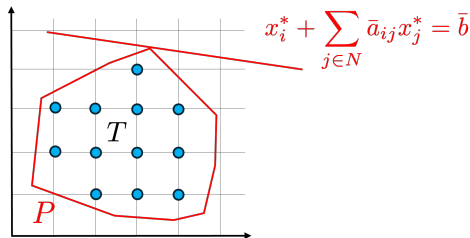
$$A_B x_B^* + A_N x_N^* = b \quad \Leftrightarrow \quad x_B^* + A_B^{-1} A_N x_N^* = A_B^{-1} b$$

- Consider an equality in which the right-hand-side is **fractional**

$$x_i^* + \sum_{j \in N} \bar{a}_{ij} x_j^* = \bar{b}$$

$$\forall x \in T \Rightarrow x \geq 0 \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \bar{b}$$

$$\forall x \in T \Rightarrow x \in \mathbb{Z}^n \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b} \rfloor$$



# Improving LP Relaxations With Cuts

- **Setup:**  $A, b, c$  with rational entries and the IP:

$$\text{minimize} \{ c^T x : Ax = b, x \geq 0, x \in \mathbb{Z}^n \}$$

- If  $x^* = [x_B^*; x_N^*]$  be a b.f.s. for the LP relaxation. Then we have:

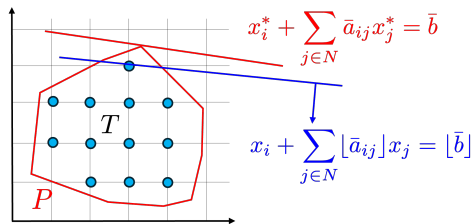
$$A_B x_B^* + A_N x_N^* = b \Leftrightarrow x_B^* + A_B^{-1} A_N x_N^* = A_B^{-1} b$$

- Consider an equality in which the right-hand-side is **fractional**

$$x_i^* + \sum_{j \in N} \bar{a}_{ij} x_j^* = \bar{b}$$

$$\forall x \in T \Rightarrow x \geq 0 \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \bar{b}$$

$$\forall x \in T \Rightarrow x \in \mathbb{Z}^n \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b} \rfloor$$



# Improving LP Relaxations With Cuts

- **Setup:**  $A, b, c$  with rational entries and the IP:

$$\text{minimize} \{ c^T x : Ax = b, x \geq 0, x \in \mathbb{Z}^n \}$$

- If  $x^* = [x_B^*; x_N^*]$  be a b.f.s. for the LP relaxation. Then we have:

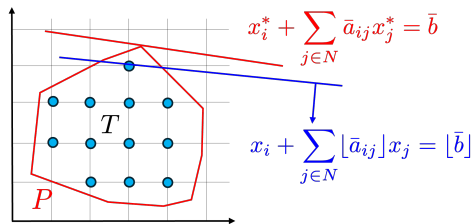
$$A_B x_B^* + A_N x_N^* = b \Leftrightarrow x_B^* + A_B^{-1} A_N x_N^* = A_B^{-1} b$$

- Consider an equality in which the right-hand-side is **fractional**

$$x_i^* + \sum_{j \in N} \bar{a}_{ij} x_j^* = \bar{b}$$

$$\forall x \in T \Rightarrow x \geq 0 \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \bar{b}$$

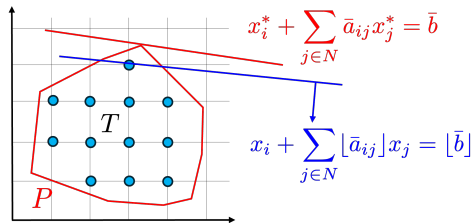
$$\forall x \in T \Rightarrow x \in \mathbb{Z}^n \Rightarrow x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b} \rfloor$$



- This inequality is **satisfied by all integer solutions**  $x \in T$
- It is **not satisfied** by  $x^*$  because  $x_i^* = \bar{b}$  is fractional
- **Gomory cut**

# Improving LP Relaxations With Cuts

$$x_i + \sum_{j \in N} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b} \rfloor, \forall x \in T$$



- **Gomory cut**
- Systematically adding all the Gomory cuts lead to first **cutting algorithm** for IP
  1. Solve the linear relaxation and get an optimal solution  $x^*$
  2. If  $x^*$  is integer stop
  3. If not, add a cut (i.e., linear inequality that all integer solutions satisfy but that  $x^*$  does not satisfy) and go to step 1 again.
- Can show that this is guaranteed to terminate
- *If you're wondering how this works for  $Ax \leq b$  or why it terminates, see notes!*

# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back



# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back
  1. Select  $j \in \{1, \dots, n\}$ .

# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back
  1. Select  $j \in \{1, \dots, n\}$ .
  2. Multiply each inequality with  $x_j$  and then  $1 - x_j$  to generate **nonlinear** inequalities:

$$x_j(Ax - b) \geq 0, \quad (1 - x_j)(Ax - b) \geq 0.$$

# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back
  1. Select  $j \in \{1, \dots, n\}$ .
  2. Multiply each inequality with  $x_j$  and then  $1 - x_j$  to generate **nonlinear** inequalities:

$$x_j(Ax - b) \geq 0, \quad (1 - x_j)(Ax - b) \geq 0.$$

3. Linearize system by substituting  $y_{ij}$  for  $x_i x_j$  (for  $i \neq j$ ), and  $x_j$  for  $x_j^2$ .  
Call resulting **polyhedron** in variables  $(x, y)$  as  $M_j$  (dimension  $\mathbb{R}^{2n}$ ).

# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back
  1. Select  $j \in \{1, \dots, n\}$ .
  2. Multiply each inequality with  $x_j$  and then  $1 - x_j$  to generate **nonlinear** inequalities:

$$x_j(Ax - b) \geq 0, \quad (1 - x_j)(Ax - b) \geq 0.$$

3. Linearize system by substituting  $y_{ij}$  for  $x_i x_j$  (for  $i \neq j$ ), and  $x_j$  for  $x_j^2$ .  
Call resulting **polyhedron** in variables  $(x, y)$  as  $M_j$  (dimension  $\mathbb{R}^{2n}$ ).
4. Project  $M_j$  onto the  $x$ -variables. Let  $P_j$  be the resulting polyhedron.

# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back
  1. Select  $j \in \{1, \dots, n\}$ .
  2. Multiply each inequality with  $x_j$  and then  $1 - x_j$  to generate **nonlinear** inequalities:

$$x_j(Ax - b) \geq 0, \quad (1 - x_j)(Ax - b) \geq 0.$$

3. Linearize system by substituting  $y_{ij}$  for  $x_i x_j$  (for  $i \neq j$ ), and  $x_j$  for  $x_j^2$ .  
Call resulting **polyhedron** in variables  $(x, y)$  as  $M_j$  (dimension  $\mathbb{R}^{2n}$ ).
  4. Project  $M_j$  onto the  $x$ -variables. Let  $P_j$  be the resulting polyhedron.
- **Claims.** (i) Every **binary**  $x \in P$  satisfies  $x \in P_j$ . (ii)  $P_j \subseteq P$ .

# Lift-and-Project

- Balas, Céria and Cornuéjols introduced a new approach
- **Binary** IP, feasible set  $x \in P \cap \{0, 1\}^n$  where  $P := \{x \in \mathbb{R}^n : Ax \geq b, x \geq 0\}$
- **Key idea:** lift linear relaxation polyhedron  $P$  to higher dimension where IP formulation is strengthened, and project back
  1. Select  $j \in \{1, \dots, n\}$ .
  2. Multiply each inequality with  $x_j$  and then  $1 - x_j$  to generate **nonlinear** inequalities:

$$x_j(Ax - b) \geq 0, \quad (1 - x_j)(Ax - b) \geq 0.$$

3. Linearize system by substituting  $y_{ij}$  for  $x_i x_j$  (for  $i \neq j$ ), and  $x_j$  for  $x_j^2$ .  
Call resulting **polyhedron** in variables  $(x, y)$  as  $M_j$  (dimension  $\mathbb{R}^{2n}$ ).
  4. Project  $M_j$  onto the  $x$ -variables. Let  $P_j$  be the resulting polyhedron.
- **Claims.** (i) Every **binary**  $x \in P$  satisfies  $x \in P_j$ . (ii)  $P_j \subseteq P$ .
  - $\bigcap_{j=1}^n P_j$  is called the **lift-and-project closure**. Clearly,  $\bigcap_{j=1}^n P_j \subseteq P$
  - Bonami and Minoux : 35 Mixed 0-1 IPs from MIPLIB library, lift-and-project closure reduces integrality gap by **37% on average**

# Other Cuts

- **Mixed-Integer Rounding (MIR) Cuts:** designed for general integer variables
- **Knapsack Cover Cuts:** applied for knapsack constraint

$$w \geq 0, w^T x \leq K \Rightarrow$$

# Other Cuts

- **Mixed-Integer Rounding (MIR) Cuts:** designed for general integer variables
- **Knapsack Cover Cuts:** applied for knapsack constraint

$$w \geq 0, w^T x \leq K \Rightarrow \sum_i x_i \leq |C| - 1 \text{ for any } C : \sum_{i \in C} w_i > K \text{ (Cover)}$$

- **Clique Cuts:** used to strengthen  $\sum_{i=1}^n x_i \leq 1$  when some of the  $x_i$  form a **clique**
- **Flow Cover** and **Flow Path Cuts:** specialized cuts for network flow problems
- **Lattice-Free Cuts, Multi-Branch Split Cuts**
- **Comb Inequalities** for TSP
- Solvers like Gurobi have many of these built-in and allow adding custom cuts
- Adding “good” cuts is problem-dependent; requires good understanding of combinatorial structure



# Solving IPs

IPs “hard,” but many methods devised

- **Exact algorithms:** guaranteed to find optimal solution, but may take exponential number of iterations
  - cutting planes
  - branch and bound
  - branch and cut
  - lift-and-project methods
  - dynamic programming methods
- **Approximation algorithms:** suboptimal solution with a bound on the degree of its suboptimality, in polynomial time
- **Heuristic algorithms:** suboptimal solution, typically no guarantees on its quality; typically run fast
  - local search methods
  - simulated annealing
  - ...

# Branch and Bound

- More general formulation: let  $F$  be set of feasible solutions to an IP
  1. Maintain upper bound  $U$ , lower bound  $L$  on problem's objective
  2. Partition  $F$  into finite collection of subsets  $F_i$
  3. Choose an unsolved subproblem and solve it; only need a **lower bound**  $\ell(F_i)$  on cost:

$$\ell(F_i) \leq \min_{x \in F_i} c^T x.$$

4. If  $\ell(F_i) \geq U$ , no need to explore subproblem  $F_i$  further!
5. Otherwise, partition  $F_i$  further and update collection of subproblems/nodes to explore
6. If we get a feasible solution, update the upper bound  $U$
7. If  $U - L \leq \epsilon$ , stop
8. When solving all children of a given node, can update lower bound at the node

# Branch and Bound

- More general formulation: let  $F$  be set of feasible solutions to an IP
  1. Maintain upper bound  $U$ , lower bound  $L$  on problem's objective
  2. Partition  $F$  into finite collection of subsets  $F_i$
  3. Choose an unsolved subproblem and solve it; only need a **lower bound**  $\ell(F_i)$  on cost:

$$\ell(F_i) \leq \min_{x \in F_i} c^T x.$$

4. If  $\ell(F_i) \geq U$ , no need to explore subproblem  $F_i$  further!
  5. Otherwise, partition  $F_i$  further and update collection of subproblems/nodes to explore
  6. If we get a feasible solution, update the upper bound  $U$
  7. If  $U - L \leq \epsilon$ , stop
  8. When solving all children of a given node, can update lower bound at the node
- Many **choices**:
    1. How to **explore subproblems**: “breadth-first search” vs “depth-first search” vs...
    2. How to **derive lower bound**  $\ell(F_i)$ : LP relaxation vs. Lagrangean duality
    3. Improve LP relaxations by **adding cuts**: **branch-and-cut** approaches
    4. How to **partition a problem** into subproblems? We used  $x_i \leq \lfloor x_i^* \rfloor$  and  $x_i \geq \lceil x_i^* \rceil$

# Lagrangian Duality in IP

- **Good lower bounds critical for MILPs!**

$$Z_{\text{IP}} := \min \{c^{\top}x : Ax \geq b, Dx \geq d, x \in \mathbb{Z}^n\}$$

- We get a lower bound from LP relaxation:

$$Z_{\text{LP}} := \min \{c^{\top}x : Ax \geq b, Dx \geq d\} \Rightarrow Z_{\text{LP}} \leq Z_{\text{IP}}$$

# Lagrangian Duality in IP

- **Good lower bounds critical for MILPs!**

$$Z_{\text{IP}} := \min \{c^T x : Ax \geq b, Dx \geq d, x \in \mathbb{Z}^n\}$$

- We get a lower bound from LP relaxation:

$$Z_{\text{LP}} := \min \{c^T x : Ax \geq b, Dx \geq d\} \Rightarrow Z_{\text{LP}} \leq Z_{\text{IP}}$$

- Suppose the **“ugly/hard” constraints** are  $Ax \geq b$  ...

... and we are able to **minimize efficiently**  $c^T x$  **over**  $\mathcal{X} := \{x \in \mathbb{Z}^n \mid Dx \geq d\}$

# Lagrangian Duality in IP

- **Good lower bounds critical for MILPs!**

$$Z_{\text{IP}} := \min \{c^\top x : Ax \geq b, Dx \geq d, x \in \mathbb{Z}^n\}$$

- We get a lower bound from LP relaxation:

$$Z_{\text{LP}} := \min \{c^\top x : Ax \geq b, Dx \geq d\} \Rightarrow Z_{\text{LP}} \leq Z_{\text{IP}}$$

- Suppose the **“ugly/hard” constraints** are  $Ax \geq b$  ...

... and we are able to **minimize efficiently**  $c^\top x$  **over**  $\mathcal{X} := \{x \in \mathbb{Z}^n \mid Dx \geq d\}$

- Let  $p \geq 0$  be dual variables (**Lagrange multipliers**) for  $Ax \geq b$ ; form Lagrangean:

$$\mathcal{L}(x, p) := c^\top x + p^\top (b - Ax)$$

# Lagrangian Duality in IP

- **Good lower bounds critical for MILPs!**

$$Z_{\text{IP}} := \min \{c^\top x : Ax \geq b, Dx \geq d, x \in \mathbb{Z}^n\}$$

- We get a lower bound from LP relaxation:

$$Z_{\text{LP}} := \min \{c^\top x : Ax \geq b, Dx \geq d\} \Rightarrow Z_{\text{LP}} \leq Z_{\text{IP}}$$

- Suppose the **“ugly/hard” constraints** are  $Ax \geq b$  ...

... and we are able to **minimize efficiently**  $c^\top x$  **over**  $\mathcal{X} := \{x \in \mathbb{Z}^n \mid Dx \geq d\}$

- Let  $p \geq 0$  be dual variables (**Lagrange multipliers**) for  $Ax \geq b$ ; form Lagrangean:

$$\mathcal{L}(x, p) := c^\top x + p^\top (b - Ax)$$

- Then we can get the following lower bound on  $Z_{\text{IP}}$ :

$$\forall p \geq 0, g(p) := \min_{x \in \mathcal{X}} [c^\top x + p^\top (b - Ax)] \Rightarrow g(p) \leq Z_{\text{IP}}$$

# Lagrangian Duality in IP

- **Good lower bounds critical for MILPs!**

$$Z_{\text{IP}} := \min \{ c^\top x : Ax \geq b, Dx \geq d, x \in \mathbb{Z}^n \}$$

- We get a lower bound from LP relaxation:

$$Z_{\text{LP}} := \min \{ c^\top x : Ax \geq b, Dx \geq d \} \Rightarrow Z_{\text{LP}} \leq Z_{\text{IP}}$$

- Suppose the **“ugly/hard” constraints** are  $Ax \geq b$  ...

... and we are able to **minimize efficiently**  $c^\top x$  **over**  $\mathcal{X} := \{x \in \mathbb{Z}^n \mid Dx \geq d\}$

- Let  $p \geq 0$  be dual variables (**Lagrange multipliers**) for  $Ax \geq b$ ; form Lagrangean:

$$\mathcal{L}(x, p) := c^\top x + p^\top (b - Ax)$$

- Then we can get the following lower bound on  $Z_{\text{IP}}$ :

$$\forall p \geq 0, g(p) := \min_{x \in \mathcal{X}} [c^\top x + p^\top (b - Ax)] \Rightarrow g(p) \leq Z_{\text{IP}}$$

- **Important!** We are **not dualizing** all the constraints!

- We keep the constraints  $x \in \mathcal{X}$  because these are “easy”
- Similar to LP developments: recall how we kept the constraints  $x_i \geq 0$  or  $x_i \leq 0$
- What matters is that we can easily compute  $g(p)$  for any  $p \geq 0$



# Lagrangian Duality in IP

- Because  $g(p) \leq Z_{\text{IP}}, \forall p \geq 0$ , we can look for **the best lower bound**:

$$Z_D := \max_{p \geq 0} g(p) \quad (2)$$

- This is the **Lagrangian dual** of our problem.
  - $g(p)$  **piece-wise linear, concave**; supergradient available
  - Can compute  $Z_D$  using first-order-methods
  - **Weak duality holds**:  $Z_D \leq Z_{\text{IP}}$
  - Unlike LP, we do **not** have a strong duality result!

# Lagrangian Duality in IP

- Because  $g(p) \leq Z_{\text{IP}}, \forall p \geq 0$ , we can look for **the best lower bound**:

$$Z_D := \max_{p \geq 0} g(p) \quad (2)$$

- This is the **Lagrangian dual** of our problem.
  - $g(p)$  **piece-wise linear, concave**; supergradient available
  - Can compute  $Z_D$  using first-order-methods
  - **Weak duality holds**:  $Z_D \leq Z_{\text{IP}}$
  - Unlike LP, we do **not** have a strong duality result!
- Most important result here (recall that  $\mathcal{X} := \{x \in \mathbb{Z}^n \mid Dx \geq d\}$ )

$$Z_D = \min \{c^\top x : Ax \geq b, \quad x \in \text{conv } \mathcal{X}\}.$$

- Immediate consequence: we get **stronger bounds than from LP relaxation**,

$$Z_{\text{IP}} \leq Z_D \leq Z_{\text{LP}}.$$

- Details, proofs: see notes

# Other Methods

- **Dynamic Programming** very powerful
- Can solve in pseudo-polynomial time IPs in **fixed dimension**
- Heuristics can also be powerful
  - Local search
  - Simulated annealing
  - Genetic algorithms, “ant colony optimization”, etc.