Integer Programming, IP - Part 1

Stefano Gualandi
Università di Pavia, Dipartimento di Matematica

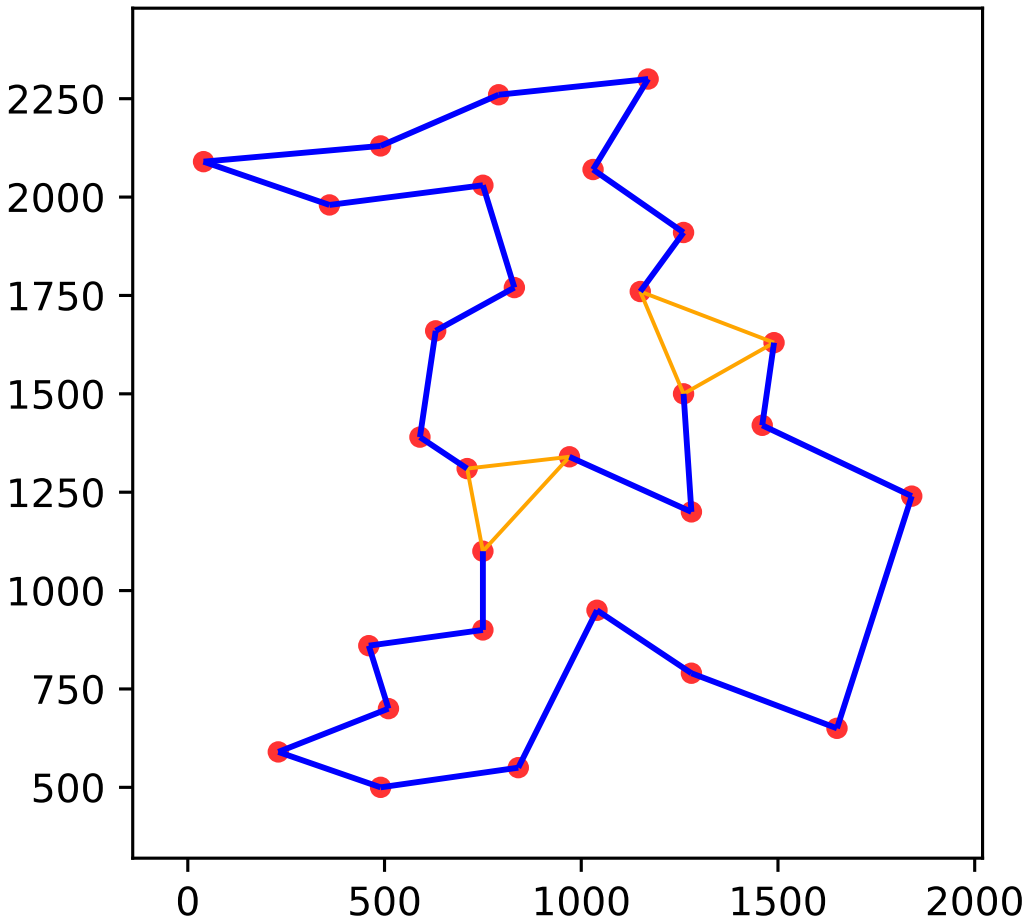
email: stefano.gualandi@unipv.it
twitter: @famo2spaghi
blog: http://stegua.github.com

Introduction

The program of this course on **Operations Research** is:

1. Modeling Optimization Problems
2. Network Flows Models and Algorithms
3. Linear Programming (LP)
4. Integer Linear Programming (ILP)

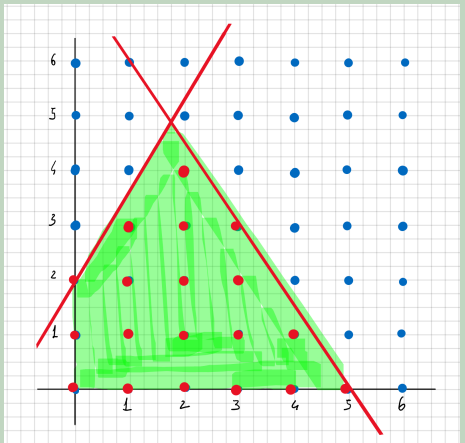
Comb Inequality for TSP: Intuition



Example 1 and 2: Integer Programs with two variables

Consider the following Integer Program (IP) (Wolsey's book, Chap. 1):

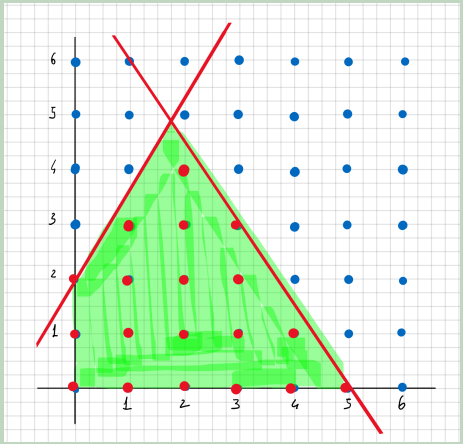
$$\begin{aligned} \max \quad & x_1 + 0.64x_2 \\ \text{s.t.} \quad & 50x_1 + 31x_2 \leq 250 \\ & 3x_1 - 2x_2 \geq -4 \\ & x_1, x_2 \in \mathbb{Z}. \end{aligned}$$



Example 1 and 2: Integer Programs with two variables

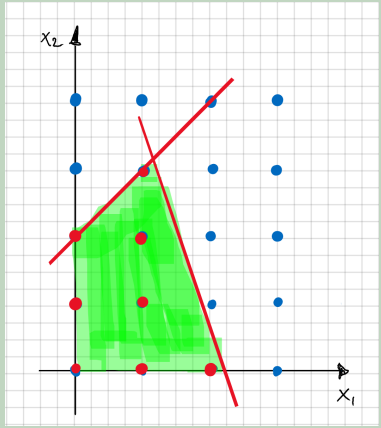
Consider the following Integer Program (IP) (Wolsey's book, Chap. 1):

$$\begin{aligned} \max \quad & x_1 + 0.64x_2 \\ \text{s.t.} \quad & 50x_1 + 31x_2 \leq 250 \\ & 3x_1 - 2x_2 \geq -4 \\ & x_1, x_2 \in \mathbb{Z}. \end{aligned}$$



Here, it is another IP (Ref. Chap. 1):

$$\begin{aligned} \max \quad & 5.5x_1 + 2.1x_2 \\ \text{s.t.} \quad & -x_1 + x_2 \leq 2 \\ & 8x_1 + 2x_2 \leq 17 \\ & x_1, x_2 \in \mathbb{Z}. \end{aligned}$$



Pure and Mixed Integer Linear Programming

A *pure integer linear program* is

$$\begin{aligned} (IP) \quad \max \quad & c^\intercal x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0, x \in \mathbb{Z}^n. \end{aligned}$$

Where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $A \in \mathbb{R}^{m \times n}$.
The set

$$S := \{x \in \mathbb{Z}^n_+ \mid Ax \leq b\}$$

is called a *pure integer linear set*.

A **mixed** *integer linear program* is

$$\begin{aligned} (MIP) \quad \max \quad & c^\intercal x + h^\intercal y \\ \text{s.t.} \quad & Ax + Gy \leq b \\ & x \geq 0, x \in \mathbb{Z}^n \\ & y \geq 0, y \in \mathbb{R}^p. \end{aligned}$$

Where $h \in \mathbb{R}^m$ and $G \in \mathbb{R}^{m \times p}$. The set

$$S := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$$

is called a *mixed integer linear set*.

Solving Integer Programs is a difficult task!

Pure and Mixed Integer Linear Programming

A *pure integer linear program* is

$$\begin{aligned} (IP) \quad \max \quad & c^\intercal x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0, x \in \mathbb{Z}^n. \end{aligned}$$

Where $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, and $A \in \mathbb{R}^{m \times n}$. The set

$$S := \{x \in \mathbb{Z}^n_+ \mid Ax \leq b\}$$

is called a *pure integer linear set*.

A **mixed** *integer linear program* is

$$\begin{aligned} (MIP) \quad \max \quad & c^\intercal x + h^\intercal y \\ \text{s.t.} \quad & Ax + Gy \leq b \\ & x \geq 0, x \in \mathbb{Z}^n \\ & y \geq 0, y \in \mathbb{R}^p. \end{aligned}$$

Where $h \in \mathbb{R}^m$ and $G \in \mathbb{R}^{m \times p}$. The set

$$S := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$$

is called a *mixed integer linear set*.

Solving Integer Programs is a difficult task!

A successful approach in computational mathematics is to find a relaxation that is easier to solve numerically and gives a good approximation.

IP Examples

- 0-1 Knapsack Problem: among n possible projects for investment, given a budget b, where a_j is the investment cost for project j, and c_j is the expected profit, you have to select the overall best combination of projects:

$$\max \{c^T x \mid a^T x \leq b, x \in \{0,1\}^n\}.$$

IP Examples

- 0-1 Knapsack Problem: among n possible projects for investment, given a budget b, where a_j is the investment cost for project j, and c_j is the expected profit, you have to select the overall best combination of projects:

$$\max \{c^T x \mid a^T x \leq b, x \in \{0,1\}^n\}.$$

- Set Covering: Given a given set of districts $M = \{1, \ldots, m\}$, you have to decide where to install new emergency centers, given a set $N = \{1, \ldots, n\}$ of potential locations.

IP Examples

- 0-1 Knapsack Problem: among n possible projects for investment, given a budget b, where a_j is the investment cost for project j, and c_j is the expected profit, you have to select the overall best combination of projects:

$$\max \{c^T x \mid a^T x \leq b, x \in \{0,1\}^n\}.$$

- Set Covering: Given a given set of districts $M = \{1, \ldots, m\}$, you have to decide where to install new emergency centers, given a set $N = \{1, \ldots, n\}$ of potential locations. Let c_j the cost of opening center j, and let $a_{ij} = 1$ if district i is served by center j, and $a_{ij} = 0$ otherwise.

IP Examples

- 0-1 Knapsack Problem: among n possible projects for investment, given a budget b, where a_j is the investment cost for project j, and c_j is the expected profit, you have to select the overall best combination of projects:

$$\max \{c^\intercal x \mid a^\intercal x \leq b, x \in \{0,1\}^n\}.$$

- Set Covering: Given a given set of districts $M = \{1, \ldots, m\}$, you have to decide where to install new emergency centers, given a set $N = \{1, \ldots, n\}$ of potential locations. Let c_j the cost of opening center j, and let $a_{ij} = 1$ if district i is served by center j, and $a_{ij} = 0$ otherwise.
  We have to minimize the overall opening cost while serving every district:

$$\min \left\{ c^\intercal x \mid \sum_{j=1}^{n} a_{ij} x_j \geq 1, \ i = 1, \ldots, m, \ \ x \in \{0,1\}^n \right\}.$$

IP Examples

- 0-1 Knapsack Problem: among n possible projects for investment, given a budget b, where a_j is the investment cost for project j, and c_j is the expected profit, you have to select the overall best combination of projects:

$$\max \{c^\intercal x \mid a^\intercal x \leq b, x \in \{0,1\}^n\}.$$

- Set Covering: Given a given set of districts $M = \{1, \ldots, m\}$, you have to decide where to install new emergency centers, given a set $N = \{1, \ldots, n\}$ of potential locations. Let c_j the cost of opening center j, and let $a_{ij} = 1$ if district i is served by center j, and $a_{ij} = 0$ otherwise.
  We have to minimize the overall opening cost while serving every district:

$$\min \left\{ c^\intercal x \mid \sum_{j=1}^{n} a_{ij} x_j \geq 1, \ i = 1, \ldots, m, \ \ x \in \{0,1\}^n \right\}.$$

- Traveling Salesman Problem (TSP): A student from the University of Pavia must sell the tickets for the next post-Covid19 re-opening party, planned on March, 1st, 2021. For this reason, she must visit all the n residences of the city exactly once, and then she has to return back to her own residence. The time taken to go from residence i to residence j is c_{ij}, and the visiting time is fixed for each residence.

Combinatorial Explosion

- Assignment Problem: the are $n!$ possible solutions.

Combinatorial Explosion

- Assignment Problem: the are $n!$ possible solutions.
- 0-1 Knapsack Problem, Set Covering: the are 2^n possible solutions.

Combinatorial Explosion

- Assignment Problem: the are $n!$ possible solutions.
- 0-1 Knapsack Problem, Set Covering: the are 2^n possible solutions.
- Traveling Salesman Problem (TSP): there are $(n-1)!$ possible tours.

Combinatorial Explosion

- Assignment Problem: the are $n!$ possible solutions.
- 0-1 Knapsack Problem, Set Covering: the are 2^n possible solutions.
- Traveling Salesman Problem (TSP): there are $(n-1)!$ possible tours.

Computational implications:

n	$\log(n)$	$\sqrt{n}$	n^2	2^n	$n!$
10	3.32	3.16	10^2	1.02×10^3	3.60×10^6

Combinatorial Explosion

- Assignment Problem: the are $n!$ possible solutions.
- 0-1 Knapsack Problem, Set Covering: the are 2^n possible solutions.
- Traveling Salesman Problem (TSP): there are $(n-1)!$ possible tours.

Computational implications:

n	$\log(n)$	$\sqrt{n}$	n^2	2^n	$n!$
10	3.32	3.16	10^2	1.02×10^3	3.60×10^6
100	6.64	10.0	10^4	1.27×10^{30}	9.33×10^{157}

Combinatorial Explosion

- Assignment Problem: the are $n!$ possible solutions.
- 0-1 Knapsack Problem, Set Covering: the are 2^n possible solutions.
- Traveling Salesman Problem (TSP): there are $(n-1)!$ possible tours.

Computational implications:

n	$\log(n)$	$\sqrt{n}$	n^2	2^n	$n!$
10	3.32	3.16	10^2	1.02×10^3	3.60×10^6
100	6.64	10.0	10^4	1.27×10^{30}	9.33×10^{157}
1000	9.97	31.6	10^6	1.07×10^{301}	4.02×10^{2567}

Mixed Integer Programming: Uncapacitated Lot-Sizing

Problem: To decide a *production plan* for an n-period horizon for a single product.

Mixed Integer Programming: Uncapacitated Lot-Sizing

Problem: To decide a *production plan* for an n-period horizon for a single product.

Data:

- f_t: is the fixed cost of producing in period t
- p_t: is the unit production cost in period t
- h_t: is the unit storage cost in period t
- d_t: is the demand in period t

Mixed Integer Programming: Uncapacitated Lot-Sizing

Problem: To decide a *production plan* for an n-period horizon for a single product.

Data:

- f_t: is the fixed cost of producing in period t
- p_t: is the unit production cost in period t
- h_t: is the unit storage cost in period t
- d_t: is the demand in period t

Variables:

- x_t: is the amount produced in period t
- s_t: is the stock at the end of period t
- y_t: equal to 1 if production occurs in period t, and equal to 0 otherwise

Mixed Integer Programming: Uncapacitated Lot-Sizing

Problem: To decide a *production plan* for an n-period horizon for a single product.

Data:

- f_t: is the fixed cost of producing in period t
- p_t: is the unit production cost in period t
- h_t: is the unit storage cost in period t
- d_t: is the demand in period t

Variables:

- x_t: is the amount produced in period t
- s_t: is the stock at the end of period t
- y_t: equal to 1 if production occurs in period t, and equal to 0 otherwise

Model:

$$\min \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t$$

$$\begin{aligned} \text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t && t = 1, \ldots, n \\ & x_t \leq M y_t && t = 1, \ldots, n \\ & s_0 = 0, s_t, x_t \geq 0, y_t \in \{0,1\} && t = 1, \ldots, n \end{aligned}$$

Linear Programming Relaxation

Given a mixed integer set $S \subseteq \mathbb{Z}^n \times \mathbb{R}^p$, a **linear relaxation** of S is a set of the form

$$P' := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^p \mid A'x + G'y \leq b'\}$$

that contains S, that is, $S \subseteq P'$.

Linear Programming Relaxation

Given a mixed integer set $S \subseteq \mathbb{Z}^n \times \mathbb{R}^p$, a **linear relaxation** of S is a set of the form

$$P' := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^p \mid A'x + G'y \leq b'\}$$

that contains S, that is, $S \subseteq P'$.

A **Linear Programming Relaxation** of (MIP) is a linear program

$$\max \ \{c^\intercal x + h^\intercal y \mid (x, y) \in P'\}$$

Linear Programming Relaxation

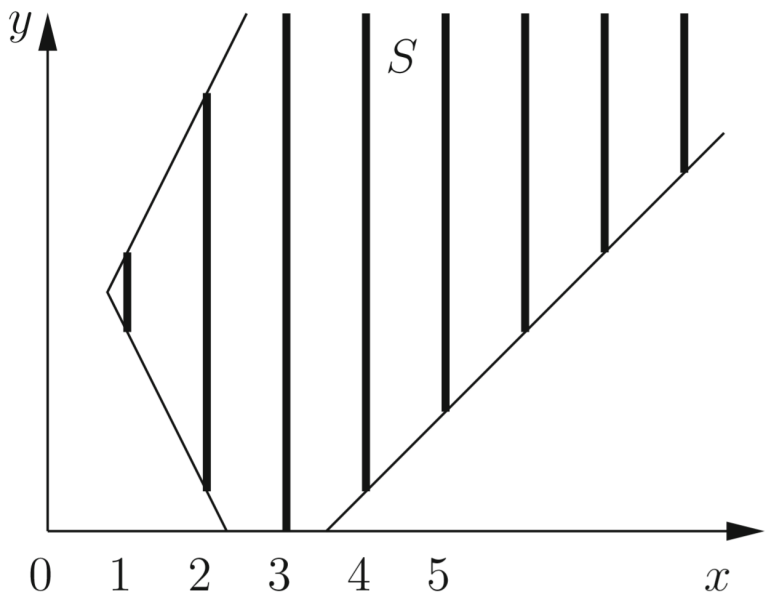
Given a mixed integer set $S \subseteq \mathbb{Z}^n \times \mathbb{R}^p$, a **linear relaxation** of S is a set of the form

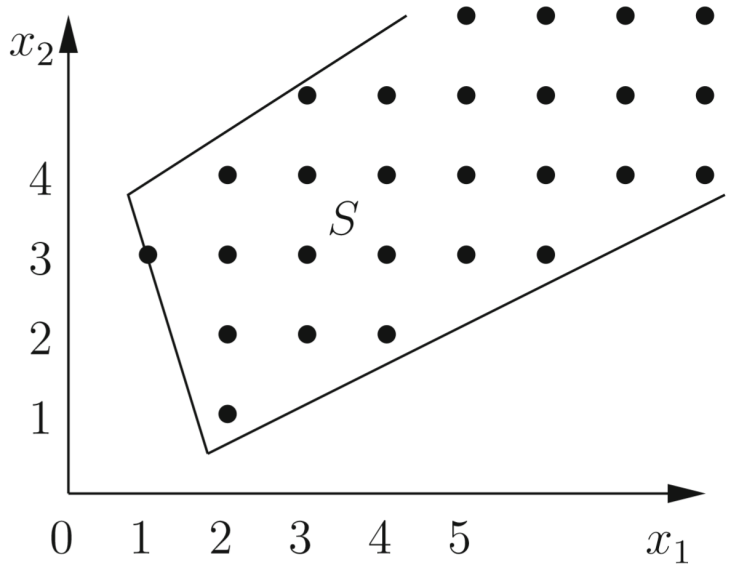
$$P' := \{(x, y) \in \mathbb{R}^n \times \mathbb{R}^p \mid A'x + G'y \leq b'\}$$

that contains S, that is, $S \subseteq P'$.

A **Linear Programming Relaxation** of (MIP) is a linear program

$$\max \ \{c^\intercal x + h^\intercal y \mid (x, y) \in P'\}$$





Methods for Solving Integer Programs

The two main methods for solving Integer Programs are:

Methods for Solving Integer Programs

The two main methods for solving Integer Programs are:

1. Cutting planes algorithms

Methods for Solving Integer Programs

The two main methods for solving Integer Programs are:

1. Cutting planes algorithms
2. Branch-and-bound methods

Methods for Solving Integer Programs

The two main methods for solving Integer Programs are:

1. Cutting planes algorithms
2. Branch-and-bound methods

Both methods are based on simple ideas, but mathematical sophistication is necessary to make them successful.

Methods for Solving Integer Programs

The two main methods for solving Integer Programs are:

1. Cutting planes algorithms
2. Branch-and-bound methods

Both methods are based on simple ideas, but mathematical sophistication is necessary to make them successful.

We assume that we have an efficient Linear Programming solver at our disposal.

Branch-and-bound 1/2

Consider the following problem:

$$\begin{aligned} \text{MILP}_0: \quad & \max \quad c^\intercal x + h^\intercal y \\ & \text{s.t.} \quad S_0 := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \end{aligned}$$

Let $P_0 := \{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.

Branch-and-bound 1/2

Consider the following problem:

$$\begin{aligned} \text{MILP}_0: \quad & \max \quad c^\intercal x + h^\intercal y \\ & \text{s.t.} \quad S_0 := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \end{aligned}$$

Let $P_0 := \{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.
Choose an index j such that x^0_j is fractional. Then, we define two subsets:

$$S_1 := S_0 \cap \{(x, y) \mid x_j \leq \lfloor x^0_j \rfloor\},$$

Branch-and-bound 1/2

Consider the following problem:

$$\begin{aligned} \text{MILP}_0: \quad & \max \; c^\intercal x + h^\intercal y \\ & \text{s.t.} \; S_0 := \{(x,y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \end{aligned}$$

Let $P_0 := \{(x,y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.
Choose an index j such that x^0_j is fractional. Then, we define two subsets:

$$S_1 := S_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\}, \qquad S_2 := S_0 \cap \{(x,y) \mid x_j \geq \lceil x^0_j \rceil\}$$

Branch-and-bound 1/2

Consider the following problem:

$$\begin{aligned} \text{MILP}_0: \quad & \max \; c^\intercal x + h^\intercal y \\ & \text{s.t.} \; S_0 := \{(x,y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \end{aligned}$$

Let $P_0 := \{(x,y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.
Choose an index j such that x^0_j is fractional. Then, we define two subsets:

$$S_1 := S_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\}, \qquad S_2 := S_0 \cap \{(x,y) \mid x_j \geq \lceil x^0_j \rceil\}$$

note that (S_1, S_2) is a partition of S.

Branch-and-bound 1/2

Consider the following problem:

$$\begin{aligned} \text{MILP}_0: \ \max \ & c^\intercal x + h^\intercal y \\ \text{s.t.} \ & S_0 := \{(x,y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \end{aligned}$$

Let $P_0 := \{(x,y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.
Choose an index j such that x^0_j is fractional. Then, we define two subsets:

$$S_1 := S_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\}, \qquad S_2 := S_0 \cap \{(x,y) \mid x_j \geq \lceil x^0_j \rceil\}$$

note that (S_1, S_2) is a partition of S. Consider now the two problems:

$$\text{MILP}_1: \ \max\{c^\intercal x + h^\intercal y \mid (x,y) \in S_1\}, \qquad \text{MILP}_2: \ \max\{c^\intercal x + h^\intercal y \mid (x,y) \in S_2\}$$

The solution to MILP_0 is equal to the best solution of these two problems.

Branch-and-bound 1/2

Consider the following problem:

$$\text{MILP}_0: \max\ c^\intercal x + h^\intercal y \quad \text{s.t. } S_0 := \{(x,y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$$

Let $P_0 := \{(x,y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.
Choose an index j such that x^0_j is fractional. Then, we define two subsets:

$$S_1 := S_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\}, \qquad S_2 := S_0 \cap \{(x,y) \mid x_j \geq \lceil x^0_j \rceil\}$$

note that (S_1, S_2) is a partition of S. Consider now the two problems:

$$\text{MILP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in S_1\}, \qquad \text{MILP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in S_2\}$$

The solution to MILP_0 is equal to the best solution of these two problems. Considering the linear formulations of S_1 and S_2:

$$P_1 := P_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\},$$

Branch-and-bound 1/2

Consider the following problem:

$$\text{MILP}_0: \max\ c^\intercal x + h^\intercal y \quad \text{s.t. } S_0 := \{(x,y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$$

Let $P_0 := \{(x,y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S_0$.
Choose an index j such that x^0_j is fractional. Then, we define two subsets:

$$S_1 := S_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\}, \qquad S_2 := S_0 \cap \{(x,y) \mid x_j \geq \lceil x^0_j \rceil\}$$

note that (S_1, S_2) is a partition of S. Consider now the two problems:

$$\text{MILP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in S_1\}, \qquad \text{MILP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in S_2\}$$

The solution to MILP_0 is equal to the best solution of these two problems. Considering the linear formulations of S_1 and S_2:

$$P_1 := P_0 \cap \{(x,y) \mid x_j \leq \lfloor x^0_j \rfloor\}, \qquad P_2 := P_0 \cap \{(x,y) \mid x_j \geq \lceil x^0_j \rceil\}$$

and, finally, consider the two LP relaxations:

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

Branch-and-bound 2/2

Starting from

$$\text{LP}_1 : \max\{c^\intercal x + h^\intercal y \mid (x, y) \in P_1\}, \qquad \text{LP}_2 : \max\{c^\intercal x + h^\intercal y \mid (x, y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP.

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP. Since $S_i \subseteq S$, then

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP. Since $S_i \subseteq S$, then $z_i \leq z^*$

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP. Since $S_i \subseteq S$, then $z_i \leq z^*$ (that is, z_i is a lower bound of MILP)
   (b) **Prune by Bound**: If x^i is not integral and $z_i < \underline{z}$ (best feasible solution), then S_i cannot contain a better solution.

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP. Since $S_i \subseteq S$, then $z_i \leq z^*$ (that is, z_i is a lower bound of MILP)
   (b) **Prune by Bound**: If x^i is not integral and $z_i < \underline{z}$ (best feasible solution), then S_i cannot contain a better solution.
   (c) **Branching**: if x^i is not integral and $z_i > \underline{z}$, then S_i can contain an improving solution for MILP.

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP. Since $S_i \subseteq S$, then $z_i \leq z^*$ (that is, z_i is a lower bound of MILP)
   (b) **Prune by Bound**: If x^i is not integral and $z_i < \underline{z}$ (best feasible solution), then S_i cannot contain a better solution.
   (c) **Branching**: if x^i is not integral and $z_i > \underline{z}$, then S_i can contain an improving solution for MILP. Let x^i_h be a fractional component, define the two sets:

$$S_{i_1} := S_i \cap \{(x,y) \mid x_h \leq \lfloor x^i_h \rfloor\},$$

Branch-and-bound 2/2

Starting from

$$\text{LP}_1: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_1\}, \qquad \text{LP}_2: \max\{c^\intercal x + h^\intercal y \mid (x,y) \in P_2\}$$

We have the following cases:

1. **Prune by Infeasibility**: If any $P_i = \emptyset$, then we have $S_i = \emptyset$, since $S_i \subseteq P_i$. Thus, MILP_i is infeasible and is no longer considered.

2. Let (x^i, y^i) be an optimal solution of LP_i and let z_i be its value, $i = 1, 2$.
   (a) **Prune by Integrality**: If x^i is **integral**, then (x^i, y^i) is optimal for MILP_i and feasible for MILP. Since $S_i \subseteq S$, then $z_i \leq z^*$ (that is, z_i is a lower bound of MILP)
   (b) **Prune by Bound**: If x^i is not integral and $z_i < \underline{z}$ (best feasible solution), then S_i cannot contain a better solution.
   (c) **Branching**: if x^i is not integral and $z_i > \underline{z}$, then S_i can contain an improving solution for MILP. Let x^i_h be a fractional component, define the two sets:

$$S_{i_1} := S_i \cap \{(x,y) \mid x_h \leq \lfloor x^i_h \rfloor\}, \qquad S_{i_2} := S_i \cap \{(x,y) \mid x_h \geq \lceil x^i_h \rceil\}$$

   and then, iterate the process recursively.

EXAMPLE CONFORTI'S BOOK ON WHITE BOARD

Branch-and-bound

- **Branching**: partitioning the set S into subsets.

Branch-and-bound

- **Branching**: partitioning the set S into subsets. In the example, branching by restricting variable domains, that is $x_j \leq u_j$ and $x_j \geq l_j$.

Branch-and-bound

- **Branching**: partitioning the set S into subsets. In the example, branching by restricting variable domains, that is $x_j \leq u_j$ and $x_j \geq l_j$.
- **Bounding**: prune the enumeration by *bounding* the objective value if the subproblems of every partition.

Branch-and-bound

- **Branching**: partitioning the set S into subsets. In the example, branching by restricting variable domains, that is $x_j \leq u_j$ and $x_j \geq l_j$.
- **Bounding**: prune the enumeration by *bounding* the objective value if the subproblems of every partition. In the example, bounding by solving the *linear programming relaxation*.

Branch-and-bound

- **Branching**: partitioning the set S into subsets. In the example, branching by restricting variable domains, that is $x_j \leq u_j$ and $x_j \geq l_j$.
- **Bounding**: prune the enumeration by *bounding* the objective value if the subproblems of every partition. In the example, bounding by solving the *linear programming relaxation*.

Let N_i be a node of the enumeration tree, and z_i the value of LP_i, and N_0 is the LP relaxation of the original MILP. Let $\mathcal{L}$ be the list of nodes to be solved. Let $\underline{z}$ be a lower bound to the optimal value z^*.

Branch-and-bound Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.

Branch-and-bound Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.

Branch-and-bound Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.
3. **Select node**: Select and remove a node N_i from $\mathcal{L}$.

Branch-and-bound Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.
3. **Select node**: Select and remove a node N_i from $\mathcal{L}$.
4. **Bound**: Solve LP_i:
   - If it is infeasible go back to 2.
   - Let (x^i, y^i) the solution of LP_i and z_i its objective value.

Branch-and-bound Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.
3. **Select node**: Select and remove a node N_i from $\mathcal{L}$.
4. **Bound**: Solve LP_i:
   - If it is infeasible go back to 2.
   - Let (x^i, y^i) the solution of LP_i and z_i its objective value.
5. **Prune**: If $z_i \leq \underline{z}_i$ go back to 2.
   If (x^i, y^i) is feasible to MILP, set $\underline{z} := z_i$, $(x^*, y^*) := (x^i, y^i)$ and go to 2.

Branch-and-bound Pseudo-code

1 **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.

2 **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.

3 **Select node**: Select and remove a node N_i from $\mathcal{L}$.

4 **Bound**: Solve LP_i:

- If it is infeasible go back to 2.
- Let (x^i, y^i) the solution of LP_i and z_i its objective value.

5 **Prune**: If $z_i \leq \underline{z}_i$ go back to 2.
If (x^i, y^i) is feasible to MILP, set $\underline{z} := z_i$, $(x^*, y^*) := (x^i, y^i)$ and go to 2.

6 **Branch**: From LP_i, define $k \geq 2$ problems $\text{LP}_{i_1}, \ldots, \text{LP}_{i_k}$ with smaller feasible regions whose union does not contain (x^i, y^i), but contains all the solutions of LP_i with $x \in \mathcal{Z}^n$.

Add the corresponding new nodes $N_{i_1}, \ldots, N_{i_k}$ to $\mathcal{L}$ and go to 2.

Gurobi Log

```
Optimize a model with 499 rows, 272 columns and 1639 nonzeros
Variable types: 16 continuous, 256 integer (256 binary)
Coefficient statistics:
  Matrix range     [1e+00, 2e+01]
  Objective range  [4e-01, 1e+05]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 2e+01]
Presolve removed 137 rows and 17 columns
Presolve time: 0.02s
Presolved: 362 rows, 255 columns, 1350 nonzeros
Variable types: 15 continuous, 240 integer (240 binary)
Found heuristic solution: objective 81.6950029

Root relaxation: objective 6.317888e+01, 50 iterations, 0.00 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   63.17888    0   26   81.69500   63.17888  22.7%     -    0s
     0     0   63.17888    0   26   81.69500   63.17888  22.7%     -    0s
     0     0   63.17888    0   26   81.69500   63.17888  22.7%     -    0s
     0     0   63.45044    0   28   81.69500   63.45044  22.3%     -    0s
     0     0   63.45044    0   28   81.69500   63.45044  22.3%     -    0s
     0     2   63.45044    0   28   81.69500   63.45044  22.3%     -    0s
*   59    57              15       75.7419933   63.75558  15.8%   5.2    0s
*  162   102              25       74.3674399   64.53335  13.2%   5.1    0s
H  728   406                       73.9998263   66.44139  10.2%   4.2    0s
* 1928   750              39       73.9876180   66.71313  9.83%   4.5    0s
```

Gurobi Log

```
    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   63.17888    0   26   81.69500   63.17888  22.7%     -    0s
     0     0   63.17888    0   26   81.69500   63.17888  22.7%     -    0s
     0     0   63.17888    0   26   81.69500   63.17888  22.7%     -    0s
     0     0   63.45044    0   28   81.69500   63.45044  22.3%     -    0s
     0     0   63.45044    0   28   81.69500   63.45044  22.3%     -    0s
     0     2   63.45044    0   28   81.69500   63.45044  22.3%     -    0s
*   59    57              15       75.7419933   63.75558  15.8%   5.2    0s
*  162   102              25       74.3674399   64.53335  13.2%   5.1    0s
H  728   406                       73.9998263   66.44139  10.2%   4.2    0s
* 1928   750              39       73.9876180   66.71313  9.83%   4.5    0s

Cutting planes:
  Gomory: 4
  MIR: 4

Explored 37902 nodes (183632 simplex iterations) in 1.83 seconds
Thread count was 8 (of 8 available processors)

Solution count 5: 73.9876 73.9998 74.3674 ... 81.695

Optimal solution found (tolerance 1.00e-04)
Best objective 7.398761804518e+01, best bound 7.398761804518e+01, gap 0.0000%
```

Main Issues in solving IP by branch-and-bound

1. Formulation (the gap $z_0 - z^*$ should be as small as possible)

Main Issues in solving IP by branch-and-bound

1. Formulation (the gap $z_0 - z^*$ should be as small as possible)
2. Heuristic (to find tight lower bounds $\underline{z}$)

Main Issues in solving IP by branch-and-bound

1. Formulation (the gap $z_0 - z^*$ should be as small as possible)
2. Heuristic (to find tight lower bounds $\underline{z}$)
3. Branching (smart partition of the feasible regions into disjoint subsets)

Main Issues in solving IP by branch-and-bound

1. Formulation (the gap $z_0 - z^*$ should be as small as possible)
2. Heuristic (to find tight lower bounds $\underline{z}$)
3. Branching (smart partition of the feasible regions into disjoint subsets)
4. Node selection (to decide which node to explore first)

Main Issues in solving IP by branch-and-bound

1. Formulation (the gap $z_0 - z^*$ should be as small as possible)
2. Heuristic (to find tight lower bounds $\underline{z}$)
3. Branching (smart partition of the feasible regions into disjoint subsets)
4. Node selection (to decide which node to explore first)

The most used method to tighten the formulation is by adding **Cutting Planes**

Valid Inequality

Consider the problem

$$\text{MILP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in S\}$$

where again $S := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$.

Valid Inequality

Consider the problem

$$\text{MILP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in S\}$$

where again $S := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$.

Let $P_0 := \{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S$ and solve

$$\text{LP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in P_0\}.$$

Valid Inequality

Consider the problem

$$\text{MILP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in S\}$$

where again $S := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$.

Let $P_0 := \{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S$ and solve

$$\text{LP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in P_0\}.$$

Let z_0 be the optimal value of LP and let (x^0, y^0) be an **optimal basic solution**.

Suppose that $(x^0, y^0) \notin S$.

Valid Inequality

Consider the problem

$$\text{MILP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in S\}$$

where again $S := \{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\}$.

Let $P_0 := \{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy \leq b\} \supseteq S$ and solve

$$\text{LP}: \qquad \max\{c^\intercal x + h^\intercal y : (x, y) \in P_0\}.$$

Let z_0 be the optimal value of LP and let (x^0, y^0) be an **optimal basic solution**.

Suppose that $(x^0, y^0) \notin S$. The main idea is to find and inequality

$$\alpha x^0 + \gamma y^0 \leq \beta$$

that is satisfied by every point in S, and such that

$$\alpha x^0 + \gamma y^0 > \beta.$$

Definition 1 (Valid Inequality)

An inequality $\alpha u \leq \beta$ is **valid** for a set $K \subseteq \mathbb{R}^d$ is it is satisfied by every point $\bar{u} \in K$.

Definition 1 (Valid Inequality)

An inequality $\alpha u \leq \beta$ is **valid** for a set $K \subseteq \mathbb{R}^d$ is it is satisfied by every point $\bar{u} \in K$.

Definition 2 (Cutting Plane)

A valid inequality $\alpha x + \gamma y \leq \beta$ for S that is violated by (x^0, y^0) is a **cutting plane** separating (x^0, y^0) from S.

Definition 1 (Valid Inequality)

An inequality $\alpha u \leq \beta$ is **valid** for a set $K \subseteq \mathbb{R}^d$ is it is satisfied by every point $\bar{u} \in K$.

Definition 2 (Cutting Plane)

A valid inequality $\alpha x + \gamma y \leq \beta$ for S that is violated by (x^0, y^0) is a **cutting plane** separating (x^0, y^0) from S.

Let $\alpha x + \gamma y \leq \beta$ be a cutting plane and define:

$$P_1 := P_0 \cap \{(x, y) \mid \alpha x + \gamma y \leq \beta\}$$

Since $S \subseteq P_1 \subset P_0$ the LP relaxation based on P_1 is stronger than that one based on P_0, and $z_0 \geq z_1 \geq z^*$.

Definition 1 (Valid Inequality)

An inequality $\alpha u \leq \beta$ is **valid** for a set $K \subseteq \mathbb{R}^d$ is it is satisfied by every point $\bar{u} \in K$.

Definition 2 (Cutting Plane)

A valid inequality $\alpha x + \gamma y \leq \beta$ for S that is violated by (x^0, y^0) is a **cutting plane** separating (x^0, y^0) from S.

Let $\alpha x + \gamma y \leq \beta$ be a cutting plane and define:

$$P_1 := P_0 \cap \{(x, y) \mid \alpha x + \gamma y \leq \beta\}$$

Since $S \subseteq P_1 \subset P_0$ the LP relaxation based on P_1 is stronger than that one based on P_0, and $z_0 \geq z_1 \geq z^*$.

The recursive application of this ides lead to the cutting plane algorithm.

Cutting Plane Approach

Starting with $i = 0$, repeat:

Recursive Step: Solve the LP problem $\max \{c^\intercal x + h^\intercal y : (x, y) \in P_i\}$.

1. If the optimal basic solution (x^i, y^i) is in S, stop.

Cutting Plane Approach

Starting with $i = 0$, repeat:

Recursive Step: Solve the LP problem $\max \{c^\intercal x + h^\intercal y : (x, y) \in P_i\}$.

1. If the optimal basic solution (x^i, y^i) is in S, stop.
2. Otherwise solve the **Separation Problem**:
   Find a cutting plane $\alpha x + \gamma y \leq \beta$ separating (x^i, y^i) from S.
   Set $P_{i+1} := P_i \cap \{(x, y) \mid \alpha x + \gamma y \leq \beta\}$ and repeat the recursive step.

Cutting Plane Approach

Starting with $i = 0$, repeat:

Recursive Step: Solve the LP problem $\max \{c^\intercal x + h^\intercal y : (x, y) \in P_i\}$.

1. If the optimal basic solution (x^i, y^i) is in S, stop.
2. Otherwise solve the **Separation Problem**:
   Find a cutting plane $\alpha x + \gamma y \leq \beta$ separating (x^i, y^i) from S.
   Set $P_{i+1} := P_i \cap \{(x, y) \mid \alpha x + \gamma y \leq \beta\}$ and repeat the recursive step.

Given that there are infinitely many cutting planes, how to produce effective cuts?

Cutting Plane Approach

Starting with $i = 0$, repeat:

Recursive Step: Solve the LP problem $\max \{c^T x + h^T y : (x, y) \in P_i\}$.

1. If the optimal basic solution (x^i, y^i) is in S, stop.
2. Otherwise solve the **Separation Problem**:
   Find a cutting plane $\alpha x + \gamma y \leq \beta$ separating (x^i, y^i) from S.
   Set $P_{i+1} := P_i \cap \{(x, y) \mid \alpha x + \gamma y \leq \beta\}$ and repeat the recursive step.

Given that there are infinitely many cutting planes, how to produce effective cuts?

Usually, there is a trade off between the running time of a separation procedure and the quality of the cutting planes it produces.

Cutting Plane Approach

Starting with $i = 0$, repeat:

Recursive Step: Solve the LP problem $\max \{c^\intercal x + h^\intercal y : (x, y) \in P_i\}$.

1. If the optimal basic solution (x^i, y^i) is in S, stop.
2. Otherwise solve the **Separation Problem**:
   Find a cutting plane $\alpha x + \gamma y \leq \beta$ separating (x^i, y^i) from S.
   Set $P_{i+1} := P_i \cap \{(x, y) \mid \alpha x + \gamma y \leq \beta\}$ and repeat the recursive step.

Given that there are infinitely many cutting planes, how to produce effective cuts?

Usually, there is a trade off between the running time of a separation procedure and the quality of the cutting planes it produces.

Moreover, from the same point (x^i, y^i) several cuts are generated at once.

Research Activities



Research Activities



Research Activities



Gomory Fractional Cut

Given n nonnegative integer variables $x_1, \ldots, x_n$ subject to the constraint

$$\sum_{j=1}^{n} a_j x_j = a_0$$

where $a_0 \notin \mathbb{Z}$,

Gomory Fractional Cut

Given n nonnegative integer variables $x_1, \ldots, x_n$ subject to the constraint

$$\sum_{j=1}^{n} a_j x_j = a_0$$

where $a_0 \notin \mathbb{Z}$, the **Gomory Fractional Cut** is

$$\sum_{j=1}^{n} (a_j - \lfloor a_j \rfloor) x_j \geq a_0 - \lfloor a_0 \rfloor.$$

This inequality is satisfied by any $x \in \mathbb{Z}_+^n$ satisfying the equation $\sum_{j=1}^{n} a_j x_j = a_0$ because $\sum_{j=1}^{n} (a_j - \lfloor a_j \rfloor) x_j = a_0 - \lfloor a_0 \rfloor + k$ for some integer k.

Gomory Fractional Cut

Given n nonnegative integer variables $x_1, \ldots, x_n$ subject to the constraint

$$\sum_{j=1}^{n} a_j x_j = a_0$$

where $a_0 \notin \mathbb{Z}$, the **Gomory Fractional Cut** is

$$\sum_{j=1}^{n} (a_j - \lfloor a_j \rfloor) x_j \geq a_0 - \lfloor a_0 \rfloor.$$

This inequality is satisfied by any $x \in \mathbb{Z}_+^n$ satisfying the equation $\sum_{j=1}^{n} a_j x_j = a_0$ because $\sum_{j=1}^{n} (a_j - \lfloor a_j \rfloor) x_j = a_0 - \lfloor a_0 \rfloor + k$ for some integer k.

Furthermore, $k \geq 0$ since the left-hand side is nonnegative.

Branch-and-cut Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.
3. **Select node**: Select and remove a node N_i from $\mathcal{L}$.
4. **Bound**: Solve LP_i:
   - If it is infeasible go back to 2.
   - Let (x^i, y^i) the solution of LP_i and z_i its objective value.
5. **Prune**: If $z_i \leq \underline{z}_i$ go back to 2.
   If (x^i, y^i) is feasible to MILP, set $\underline{z} := z_i$, $(x^*, y^*) := (x^i, y^i)$ and go to 2.

Branch-and-cut Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.
3. **Select node**: Select and remove a node N_i from $\mathcal{L}$.
4. **Bound**: Solve LP_i:
   - If it is infeasible go back to 2.
   - Let (x^i, y^i) the solution of LP_i and z_i its objective value.
5. **Prune**: If $z_i \leq \underline{z}_i$ go back to 2.
   If (x^i, y^i) is feasible to MILP, set $\underline{z} := z_i$, $(x^*, y^*) := (x^i, y^i)$ and go to 2.
6. **Add cuts?** Decide whether to branch (go to 7)

Branch-and-cut Pseudo-code

1. **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.
2. **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.
3. **Select node**: Select and remove a node N_i from $\mathcal{L}$.
4. **Bound**: Solve LP_i:
   - If it is infeasible go back to 2.
   - Let (x^i, y^i) the solution of LP_i and z_i its objective value.
5. **Prune**: If $z_i \leq \underline{z}_i$ go back to 2.
   If (x^i, y^i) is feasible to MILP, set $\underline{z} := z_i$, $(x^*, y^*) := (x^i, y^i)$ and go to 2.
6. **Add cuts?** Decide whether to branch (go to 7) or to strengthen LP_i by adding cutting planes, and then go back to 4.

Branch-and-cut Pseudo-code

1 **Initialize**: $\mathcal{L} := \{N_0\}, \underline{z} := -\infty, (x^*, y^*) := \emptyset$.

2 **Terminate?** If $\mathcal{L} = \emptyset$, then (x^*, y^*) is optimal.

3 **Select node**: Select and remove a node N_i from $\mathcal{L}$.

4 **Bound**: Solve LP_i:

- If it is infeasible go back to 2.
- Let (x^i, y^i) the solution of LP_i and z_i its objective value.

5 **Prune**: If $z_i \leq \underline{z}_i$ go back to 2.
If (x^i, y^i) is feasible to MILP, set $\underline{z} := z_i$, $(x^*, y^*) := (x^i, y^i)$ and go to 2.

6 **Add cuts?** Decide whether to branch (go to 7) or to strengthen LP_i by adding cutting planes, and then go back to 4.

7 **Branch**: From LP_i, define $k \geq 2$ problem $\text{LP}_{i_1}, \ldots, \text{LP}_{i_k}$ with smaller feasible regions whose union does not contain (x^i, y^i), but contains all the solutions of LP_i with $x \in \mathcal{Z}^n$.

Add the corresponding new nodes $N_{i_1}, \ldots, N_{i_k}$ to $\mathcal{L}$ and go to 2.

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection
4. Branching

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection
4. Branching
5. Node selection

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection
4. Branching
5. Node selection

How can we compare different formulations?

Does a **Perfect Formulation** exist?

Definition 3 (Polyhedron)

A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P := \{z \in \mathbb{R}^n \mid Ax \leq B\}$ is a Polyhedron.

Definition 3 (Polyhedron)

A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P := \{z \in \mathbb{R}^n \mid Ax \leq B\}$ is a Polyhedron.

Definition 4 (Formulation)

A polyhedron $P \subseteq \mathbb{R}^{n+p}$ is a **Formulation** for a set $X \subseteq \mathbb{Z}^n \times \mathbb{R}^p$ if and only if $X := P \cap (\mathbb{Z}^n \times \mathbb{R}^p)$.

Definition 3 (Polyhedron)

A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P := \{z \in \mathbb{R}^n \mid Ax \leq B\}$ is a Polyhedron.

Definition 4 (Formulation)

A polyhedron $P \subseteq \mathbb{R}^{n+p}$ is a **Formulation** for a set $X \subseteq \mathbb{Z}^n \times \mathbb{R}^p$ if and only if $X := P \cap (\mathbb{Z}^n \times \mathbb{R}^p)$.

Definition 5 (Comparing formulations)

Given a set $X \subseteq \mathbb{Z}^n$, and two formulations P_1 and P_2 for X, P_1 is a better formulation than P_2 if $P_1 \subset P_2$.

Definition 3 (Polyhedron)

A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P := \{z \in \mathbb{R}^n \mid Ax \leq B\}$ is a Polyhedron.

Definition 4 (Formulation)

A polyhedron $P \subseteq \mathbb{R}^{n+p}$ is a **Formulation** for a set $X \subseteq \mathbb{Z}^n \times \mathbb{R}^p$ if and only if $X := P \cap (\mathbb{Z}^n \times \mathbb{R}^p)$.

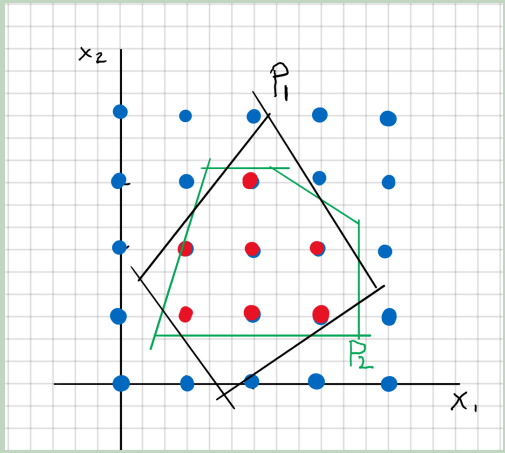
Definition 5 (Comparing formulations)

Given a set $X \subseteq \mathbb{Z}^n$, and two formulations P_1 and P_2 for X, P_1 is a better formulation than P_2 if $P_1 \subset P_2$.

Alternative Formulations

Consider the set:

$$X := \{(1,1),(2,1),(3,1),(1,2),(2,2),(3,2),(2,3)\}$$



Definition 3 (Polyhedron)

A subset of $\mathbb{R}^n$ described by a finite set of linear constraints $P := \{z \in \mathbb{R}^n \mid Ax \leq B\}$ is a Polyhedron.

Definition 4 (Formulation)

A polyhedron $P \subseteq \mathbb{R}^{n+p}$ is a **Formulation** for a set $X \subseteq \mathbb{Z}^n \times \mathbb{R}^p$ if and only if $X := P \cap (\mathbb{Z}^n \times \mathbb{R}^p)$.

Definition 5 (Comparing formulations)

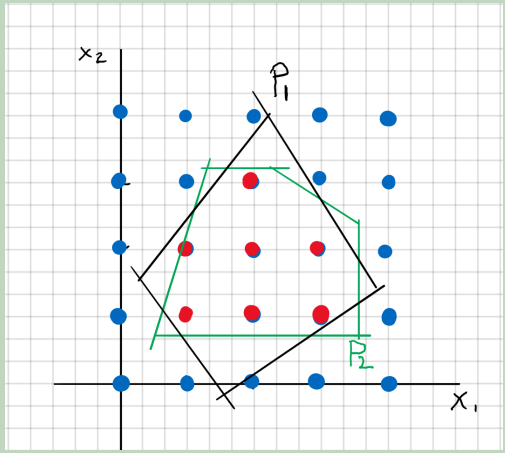
Given a set $X \subseteq \mathbb{Z}^n$, and two formulations P_1 and P_2 for X, P_1 is a better formulation than P_2 if $P_1 \subset P_2$.

Alternative Formulations

Consider the set:

$$X := \{(1,1),(2,1),(3,1),(1,2),(2,2),(3,2),(2,3)\}$$

Question: How to write a formulation for the set $X \setminus \{(2,2)\}$?



Alternative Formulations: Pool question!

0-1 Knapsack Set

Consider the set of points:

$$X := \{(0,0,0,0),(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),(0,1,0,1),(0,0,1,1)\}$$

Which among the following polyhedron are formulations of X?

$$\begin{aligned}
P_1 &:= \{x \in \mathbb{R}^4 \quad : \quad 0 \le x \le 1, \quad 83x_1 + 61x_2 + 49x_3 + 20x_4 \le 100\},\\
P_2 &:= \{x \in \mathbb{R}^4 \quad : \quad 0 \le x \le 1, \quad 4x_1 + 3x_2 + 2x_3 + x_4 \le 4\},\\
P_3 &:= \{x \in \mathbb{R}^4 \quad : \quad x_1 + x_2 + x_3 \le 1\\
&\qquad\qquad\qquad\qquad\quad x_1 + x_4 \le 1\\
&\qquad\qquad\qquad\qquad\quad x_1, x_2, x_3, x_4 \ge 0\}.
\end{aligned}$$

Alternative Formulations: Pool question!

0-1 Knapsack Set

Consider the set of points:

$$X := \{(0,0,0,0),(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),(0,1,0,1),(0,0,1,1)\}$$

Which among the following polyhedron are formulations of X?

$$\begin{aligned}
P_1 &:= \{x \in \mathbb{R}^4 \quad : \quad 0 \le x \le 1, \quad 83x_1 + 61x_2 + 49x_3 + 20x_4 \le 100\},\\
P_2 &:= \{x \in \mathbb{R}^4 \quad : \quad 0 \le x \le 1, \quad 4x_1 + 3x_2 + 2x_3 + x_4 \le 4\},\\
P_3 &:= \{x \in \mathbb{R}^4 \quad : \quad x_1 + x_2 + x_3 \le 1\\
&\qquad\qquad\qquad\qquad\quad x_1 + x_4 \le 1\\
&\qquad\qquad\qquad\qquad\quad x_1, x_2, x_3, x_4 \ge 0\}.
\end{aligned}$$

QUESTION: How to compare these three formulations?

(Re-)Formulations: Uncapacitated Lot-Sizing

Problem: To decide a *production plan* for an n-period horizon for a single product.

Data:

- f_t: is the fixed cost of producing in period t
- p_t: is the unit production cost in period t
- h_t: is the unit storage cost in period t
- d_t: is the demand in period t

Variables:

- x_t: is the amount produced in period t
- s_t: is the stock at the end of period t
- y_t: equal to 1 if production occurs in period t, and equal to 0 otherwise

Model:

$$\min \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t$$

$$\begin{aligned} \text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t && t = 1, \ldots, n \\ & x_t \leq M y_t && t = 1, \ldots, n \\ & s_0 = 0, s_t, x_t \geq 0, y_t \in \{0,1\} && t = 1, \ldots, n \end{aligned}$$

Extended Formulations: Uncapacitated Lot-Sizing

$$\begin{aligned}
\min \quad & \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t & \\
\text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t & t = 1, \ldots, n \\
& x_t \leq M y_t & t = 1, \ldots, n \\
& s_0 = 0, s_t, x_t \geq 0, y_t \in \{0, 1\} & t = 1, \ldots, n
\end{aligned}$$

We use the relation $s_t = \sum_{i=1}^{t} x_i - \sum_{i=1}^{t} d_i$ to rewrite the objective function as

$$\min \quad \sum_{t=1}^{n} c_t x_t + \sum_{t=1}^{n} f_t y_t - K$$

where $c_t = p_t + h_t + \cdots + h_n$ and the constant $K = \sum_{t=1}^{n} h_t \left(\sum_{i=1}^{t} d_i \right)$.

Extended Formulations: Uncapacitated Lot-Sizing

$$\begin{aligned}
\min \quad & \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t & \\
\text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t & t = 1, \ldots, n \\
& x_t \leq M y_t & t = 1, \ldots, n \\
& s_0 = 0, s_t, x_t \geq 0, y_t \in \{0, 1\} & t = 1, \ldots, n
\end{aligned}$$

We use the relation $s_t = \sum_{i=1}^{t} x_i - \sum_{i=1}^{t} d_i$ to rewrite the objective function as

$$\min \quad \sum_{t=1}^{n} c_t x_t + \sum_{t=1}^{n} f_t y_t - K$$

where $c_t = p_t + h_t + \cdots + h_n$ and the constant $K = \sum_{t=1}^{n} h_t \left(\sum_{i=1}^{t} d_i \right)$.

New variables: w_{it} amount produced in period i to satisfy demand in period t

Extended Formulations: Uncapacitated Lot-Sizing

$$\begin{aligned}
\min \quad & \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t & \\
\text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t & t = 1, \ldots, n \\
& x_t \leq M y_t & t = 1, \ldots, n \\
& s_0 = 0, s_t, x_t \geq 0, y_t \in \{0, 1\} & t = 1, \ldots, n
\end{aligned}$$

We use the relation $s_t = \sum_{i=1}^{t} x_i - \sum_{i=1}^{t} d_i$ to rewrite the objective function as

$$\min \quad \sum_{t=1}^{n} c_t x_t + \sum_{t=1}^{n} f_t y_t - K$$

where $c_t = p_t + h_t + \cdots + h_n$ and the constant $K = \sum_{t=1}^{n} h_t \left(\sum_{i=1}^{t} d_i \right)$.

New variables: w_{it} amount produced in period i to satisfy demand in period t

Demand: $\sum_{i=1}^{t} w_{it} = d_t$ for all t. (note $x_i = \sum_{t=i}^{n} w_{it}$)

Extended Formulations: Uncapacitated Lot-Sizing

$$\begin{aligned}
\min \quad & \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t & \\
\text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t & t = 1, \ldots, n \\
& x_t \leq M y_t & t = 1, \ldots, n \\
& s_0 = 0, s_t, x_t \geq 0, y_t \in \{0, 1\} & t = 1, \ldots, n
\end{aligned}$$

We use the relation $s_t = \sum_{i=1}^{t} x_i - \sum_{i=1}^{t} d_i$ to rewrite the objective function as

$$\min \quad \sum_{t=1}^{n} c_t x_t + \sum_{t=1}^{n} f_t y_t - K$$

where $c_t = p_t + h_t + \cdots + h_n$ and the constant $K = \sum_{t=1}^{n} h_t \left(\sum_{i=1}^{t} d_i \right)$.

New variables: w_{it} amount produced in period i to satisfy demand in period t

Demand: $\sum_{i=1}^{t} w_{it} = d_t$ for all t. (note $x_i = \sum_{t=i}^{n} w_{it}$)

Upper Bound: $w_{it} \leq d_t y_i$ for all $i, t, i \leq t$

Extended Formulations: Uncapacitated Lot-Sizing

$$\begin{aligned}
\min \quad & \sum_{t=1}^{n} p_t x_t + \sum_{t=1}^{n} h_t s_t + \sum_{t=1}^{n} f_t y_t & \\
\text{s.t.} \quad & s_{t-1} + x_t = d_t + s_t & t = 1, \ldots, n \\
& x_t \leq M y_t & t = 1, \ldots, n \\
& s_0 = 0, s_t, x_t \geq 0, y_t \in \{0, 1\} & t = 1, \ldots, n
\end{aligned}$$

We use the relation $s_t = \sum_{i=1}^{t} x_i - \sum_{i=1}^{t} d_i$ to rewrite the objective function as

$$\min \quad \sum_{t=1}^{n} c_t x_t + \sum_{t=1}^{n} f_t y_t - K$$

where $c_t = p_t + h_t + \cdots + h_n$ and the constant $K = \sum_{t=1}^{n} h_t \left(\sum_{i=1}^{t} d_i \right)$.

New variables: w_{it} amount produced in period i to satisfy demand in period t

Demand: $\sum_{i=1}^{t} w_{it} = d_t$ for all t. (note $x_i = \sum_{t=i}^{n} w_{it}$)

Upper Bound: $w_{it} \leq d_t y_i$ for all $i, t, i \leq t$

Objective Function: min $\sum_{i=1}^{n} \sum_{t=1}^{n} c_i w_{it} + \sum_{t=1}^{n} f_t y_t$.

QUESTION: How to compare the two given formulations for the Uncapacitated Lot Sizing problem?

QUESTION: How to compare the two given formulations for the Uncapacitated Lot Sizing problem?

Definition 6 (Projection of a Polyhedron)

Given a polyhedron $Q \subseteq (\mathbb{R}^n \times \mathbb{R}^p)$, the projection of Q into the subspace $\mathbb{R}^n$, denoted $\text{proj}_x Q$, is defined as:

$$\text{proj}_x Q := \{x \in \mathbb{R}^n \mid (x, w) \in Q \text{ for some } w \in \mathbb{R}^p\}.$$

QUESTION: How to compare the two given formulations for the Uncapacitated Lot Sizing problem?

Definition 6 (Projection of a Polyhedron)

Given a polyhedron $Q \subseteq (\mathbb{R}^n \times \mathbb{R}^p)$, the projection of Q into the subspace $\mathbb{R}^n$, denoted $\text{proj}_x Q$, is defined as:

$$\text{proj}_x Q := \{x \in \mathbb{R}^n \mid (x, w) \in Q \text{ for some } w \in \mathbb{R}^p\}.$$

Hence, we can compare the formulation Q with another formulation $P \subseteq \mathbb{R}^n$, by comparing $\text{proj}_x Q$ with P.

Comparing Lot Sizing Formulations

Consider the two formulations

$$
\begin{aligned}
P_1 := \Big\{ & s_{t-1} + x_t = d_t + s_t && t = 1, \ldots, n \\
& x_t \leq M y_t && t = 1, \ldots, n \\
& s_0 = 0, s_t, x_t \geq 0, 0 \leq y_t \leq 1, && t = 1, \ldots, n \Big\}.
\end{aligned}
$$

$$
\begin{aligned}
Q := \Big\{ \sum_{i=1}^{t} w_{it} &= d_t, \quad t = 1, \ldots, n \\
w_{it} &\leq d_t y_i, \quad \text{for all } i, t, i \leq t \\
x_i &= \sum_{t=i}^{n} w_{it}, \quad t = 1, \ldots, n \\
w_{it} &\geq 0, \quad \text{for all } i, t, i \leq t \\
0 &\leq y_t \leq 1, \quad t = 1, \ldots, n \Big\}
\end{aligned}
$$

Comparing Lot Sizing Formulations

Consider the two formulations

$$P_1 := \Big\{ s_{t-1} + x_t = d_t + s_t \qquad t = 1, \ldots, n$$
$$x_t \leq M y_t \qquad t = 1, \ldots, n$$
$$s_0 = 0, s_t, x_t \geq 0, 0 \leq y_t \leq 1, \qquad t = 1, \ldots, n \Big\}.$$

$$Q := \Big\{ \sum_{i=1}^{t} w_{it} = d_t, \quad t = 1, \ldots, n$$
$$w_{it} \leq d_t y_i, \quad \text{for all } i, t, i \leq t$$
$$x_i = \sum_{t=i}^{n} w_{it}, \quad t = 1, \ldots, n$$
$$w_{it} \geq 0, \quad \text{for all } i, t, i \leq t$$
$$0 \leq y_t \leq 1, \quad t = 1, \ldots, n \Big\}$$

Given $P_2 := \text{proj}_x Q$,

Comparing Lot Sizing Formulations

Consider the two formulations

$$P_1 := \Big\{ s_{t-1} + x_t = d_t + s_t \qquad t = 1, \ldots, n$$
$$x_t \leq M y_t \qquad t = 1, \ldots, n$$
$$s_0 = 0, s_t, x_t \geq 0, 0 \leq y_t \leq 1, \qquad t = 1, \ldots, n \Big\}.$$

$$Q := \Big\{ \sum_{i=1}^{t} w_{it} = d_t, \quad t = 1, \ldots, n$$
$$w_{it} \leq d_t y_i, \quad \text{for all } i, t, i \leq t$$
$$x_i = \sum_{t=i}^{n} w_{it}, \quad t = 1, \ldots, n$$
$$w_{it} \geq 0, \quad \text{for all } i, t, i \leq t$$
$$0 \leq y_t \leq 1, \quad t = 1, \ldots, n \Big\}$$

Given $P_2 := \text{proj}_x Q$, we can compare P_1 with P_2.

Comparing Lot Sizing Formulations

Consider the two formulations

$$P_1 := \Big\{ s_{t-1} + x_t = d_t + s_t \qquad t = 1, \ldots, n$$
$$x_t \leq M y_t \qquad t = 1, \ldots, n$$
$$s_0 = 0, s_t, x_t \geq 0, 0 \leq y_t \leq 1, \qquad t = 1, \ldots, n \Big\}.$$

$$Q := \Big\{ \sum_{i=1}^{t} w_{it} = d_t, \quad t = 1, \ldots, n$$
$$w_{it} \leq d_t y_i, \quad \text{for all } i, t, i \leq t$$
$$x_i = \sum_{t=i}^{n} w_{it}, \quad t = 1, \ldots, n$$
$$w_{it} \geq 0, \quad \text{for all } i, t, i \leq t$$
$$0 \leq y_t \leq 1, \quad t = 1, \ldots, n \Big\}$$

Given $P_2 := \text{proj}_x Q$, we can compare P_1 with P_2.

We can show that $P_2 \subset P_1$. For instance, consider the point $x_t = d_t, y_t = \frac{d_t}{M}$ for all t. This is an extreme point of P_1 that does not belong to P_2.

Comparing Lot Sizing Formulations

Consider the two formulations

$$P_1 := \Big\{ s_{t-1} + x_t = d_t + s_t \qquad t = 1, \dots, n$$
$$x_t \leq M y_t \qquad t = 1, \dots, n$$
$$s_0 = 0, s_t, x_t \geq 0, 0 \leq y_t \leq 1, \qquad t = 1, \dots, n \Big\}.$$

$$Q := \Big\{ \sum_{i=1}^{t} w_{it} = d_t, \quad t = 1, \dots, n$$
$$w_{it} \leq d_t y_i, \quad \text{for all } i, t, i \leq t$$
$$x_i = \sum_{t=i}^{n} w_{it}, \quad t = 1, \dots, n$$
$$w_{it} \geq 0, \quad \text{for all } i, t, i \leq t$$
$$0 \leq y_t \leq 1, \quad t = 1, \dots, n \Big\}$$

Given $P_2 := \text{proj}_x Q$, we can compare P_1 with P_2.

We can show that $P_2 \subset P_1$. For instance, consider the point $x_t = d_t, y_t = \frac{d_t}{M}$ for all t. This is an extreme point of P_1 that does not belong to P_2.

It can be shown that P_2 is the convex hull of the original problem.

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection
4. Branching

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection
4. Branching
5. Node selection

Main Issues in solving IP by Branch-and-Cut

1. Formulation (so that the gap $z_0 - z^*$ is as small as possible)
2. Heuristic (to find a good lower bound $\underline{z}$)
3. Cutting Plan Generation and Selection
4. Branching
5. Node selection

How can we design efficient **Heuristics**?

Other Type of Relaxations

1. COMBINATORIAL RELAXATION
2. LAGRANGIAN RELAXATION

Incremental Exercise in Python

1. Enumeration
2. Incremental Enumeration
3. Naive Bounding
4. Optimistic Bounding
5. Combinatorial Bounding
6. Primal Heuristic
7. Local Search Heuristic (2-opt exchange)
8. Using Hash Tables
9. ... still a long story missing!

http://www.math.uwaterloo.ca/~bico/papers/comp_chapter1.pdf

https://www.math.uwaterloo.ca/~bico/co759/bentley/bentley_tsp.html

References up to now

- Conforti, Cornuéjols, Zambell. Integer Programming. Springer, 2014.. Chap. 1: Selected paragraphs.
- Laurence Wolsey. Integer Programming, Willey.. Chap. 1,2,3: Selected paragraphs.