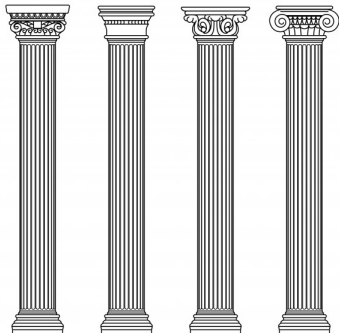


# Introduction to Column Generation

Eduardo Uchoa

Departamento de Engenharia de Produção  
Universidade Federal Fluminense, Brazil  
INRIA International Chair 2022-2026, Bordeaux



- 1 Revised Simplex Algorithm
- 2 Dantzig-Wolfe decomposition for LP
- 3 Dantzig-Wolfe decomposition for IP
- 4 DW decomposition with multiple subproblems
  - Example: Generalized Assignment Problem
- 5 DW decomposition with identical subproblems
  - Example: Cutting Stock Problem
- 6 Guidelines on when trying DW decomp for IP

- An LP has the following format:

$$\min z = cx$$

subject to

$$Ax = b$$

$$x \geq 0$$

$c$ :  $1 \times n$  vector, objective function coefficients

$A$ :  $m \times n$  matrix, constraint coefficients

$x$ :  $n \times 1$  vector, decision variables

$b$ :  $m \times 1$  vector, right-hand side constants

# Revised Simplex Algorithm for LP (Dantzig, 1953)

More efficient than the original Simplex (Dantzig, 1947)

Takes advantage of the fact that  $n$  is usually significantly larger than  $m$

## Definition: Basic solution, basic variables

Let  $(B \ N)$  be a partition of the columns in  $A$ , such that  $B$  has dimension  $m \times m$  and is invertible. Let  $x = (x_B \ x_N)$  and  $c = (c_B \ c_N)$  be the corresponding partitions of  $x$  and  $c$ . A feasible solution  $x = (x_B \ x_N)$  is said to be **basic** if  $x_N = 0$ . Variables in  $x_B$  are **basic variables**, those in  $x_N$  are **non-basic variables**.

## EXAMPLE

$$\begin{aligned} \min z = & 24x_1 + 29x_2 + 10x_3 + 38x_4 \\ \text{s.t.} \quad & x_1 + 4x_2 + 5x_3 = 60 \\ & \quad \quad 2x_2 + x_3 \leq 12 \\ & 2x_1 + x_2 - x_3 + 4x_4 \geq 10 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

## EXAMPLE

$$\begin{array}{rcll} \min z = & 24x_1 & + & 29x_2 & + & 10x_3 & + & 38x_4 \\ \text{s.t.} & x_1 & + & 4x_2 & + & 5x_3 & & & = & 60 \\ & & & 2x_2 & + & x_3 & & & \leq & 12 \\ & 2x_1 & + & x_2 & - & x_3 & + & 4x_4 & \geq & 10 \\ & x_1 & , & x_2 & , & x_3 & , & x_4 & \geq & 0 \end{array}$$

Simplex algorithms (both original and revised variants) require converting all inequalities to equalities, **slack/surplus variables** should be added.

## EXAMPLE

$$\begin{aligned} \min z = & 24x_1 + 29x_2 + 10x_3 + 38x_4 \\ \text{s.t.} & x_1 + 4x_2 + 5x_3 = 60 \\ & 2x_2 + x_3 + x_5 = 12 \\ & 2x_1 + x_2 - x_3 + 4x_4 - x_6 = 10 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$



## EXAMPLE

$$\begin{aligned}
 \min z = & 24x_1 + 29x_2 + 10x_3 + 38x_4 \\
 \text{s.t.} & \quad x_1 + 4x_2 + 5x_3 \qquad \qquad \qquad = 60 \\
 & \qquad \quad 2x_2 + x_3 \qquad \qquad \qquad + x_5 \qquad \qquad \qquad = 12 \\
 & 2x_1 + x_2 - x_3 + 4x_4 \qquad \qquad - x_6 = 10 \\
 & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0
 \end{aligned}$$

$$c = [24 \quad 29 \quad 10 \quad 38 \quad 0 \quad 0]$$

$$A = \begin{bmatrix} 1 & 4 & 5 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 1 & 0 \\ 2 & 1 & -1 & 4 & 0 & -1 \end{bmatrix}$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix}$$

$$b = \begin{bmatrix} 60 \\ 12 \\ 10 \end{bmatrix}.$$

Step 1: Find an initial basic feasible solution

Find  $m \times m$  submatrix  $B$  of  $A$  such that linear system  $Bx_B = b$  has solution  $\bar{x}_B \geq 0$ .

Suppose that  $x_1$ ,  $x_3$  and  $x_5$  are the variables chosen to be basic

$$B \cdot x_B = \begin{bmatrix} 1 & 5 & 0 \\ 0 & 1 & 1 \\ 2 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_3 \\ x_5 \end{bmatrix} = \begin{bmatrix} 60 \\ 12 \\ 10 \end{bmatrix} \Rightarrow \bar{x}_B = \begin{bmatrix} x_1 \\ x_3 \\ x_5 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \\ 2 \end{bmatrix}$$

$$z = 340$$

Step 1: Find an initial basic feasible solution

Find  $m \times m$  submatrix  $B$  of  $A$  such that linear system  $Bx_B = b$  has solution  $\bar{x}_B \geq 0$ .

Suppose that  $x_1, x_3$  and  $x_5$  are the variables chosen to be basic

$$B \cdot x_B = b \Leftrightarrow \begin{cases} x_1 + 5x_3 & = 60 \\ & x_3 + x_5 = 12 \\ 2x_1 - x_3 & = 10 \end{cases} \Rightarrow \begin{cases} x_1 = 10 \\ x_3 = 10 \\ x_5 = 2 \end{cases}$$

$$z = 340$$

## Step 2: Find the dual solution

The **reduced cost** of a variable  $x_j$  is given by  $\bar{c}_j = c_j - \pi A_j$ , where  $A_j$  is the  $j$ -th column of  $A$ . **Basic variables have zero reduced cost**. Dual solution  $\pi$  is the solution of linear system  $\pi B = c_B$ .

$$\pi B = \begin{bmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \end{bmatrix}^T \cdot \begin{bmatrix} 1 & 5 & 0 \\ 0 & 1 & 1 \\ 2 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 24 \\ 10 \\ 0 \end{bmatrix}^T \Rightarrow \begin{bmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \end{bmatrix}^T = \begin{bmatrix} 4 \\ 0 \\ 10 \end{bmatrix}^T$$

## Step 2: Find the dual solution

The **reduced cost** of a variable  $x_j$  is given by  $\bar{c}_j = c_j - \pi A_j$ , where  $A_j$  is the  $j$ -th column of  $A$ . **Basic variables have zero reduced cost**. Dual solution  $\pi$  is the solution of linear system  $\pi B = c_B$ .

$$\pi B = c_B \Leftrightarrow \begin{cases} \pi_1 & + 2\pi_3 = 24 \\ 5\pi_1 + \pi_2 - \pi_3 = 10 \\ & \pi_2 = 0 \end{cases} \Rightarrow \begin{cases} \pi_1 = 4 \\ \pi_2 = 0 \\ \pi_3 = 10 \end{cases}$$

## Step 3: Pricing (finding a variable to enter the basis)

Calculate the reduced cost ( $\bar{c}_j = c_j - \pi A_j$ ) of the non-basic variables. If no variable has negative reduced cost, the current solution is optimal. Otherwise, one of those variables can be chosen to enter the basis.

Non-basic variables are  $x_2$ ,  $x_4$ , and  $x_6$

$$\bar{c}_2 = 29 - 4\pi_1 - 2\pi_2 - \pi_3 = 3$$

$$\bar{c}_4 = 38 - 4\pi_3 = -2$$

$$\bar{c}_6 = 0 + \pi_3 = 10$$

## Step 3: Pricing (finding a variable to enter the basis)

Calculate the reduced cost ( $\bar{c}_j = c_j - \pi A_j$ ) of the non-basic variables. If no variable has negative reduced cost (positive reduced cost for maximization problems), the current solution is optimal. Otherwise, one of those variables can be chosen to enter the basis.

Non-basic variables are  $x_2$ ,  $x_4$ , and  $x_6$

$$\bar{c}_2 = 29 - 4\pi_1 - 2\pi_2 - \pi_3 = 3$$

$$\bar{c}_4 = 38 - 4\pi_3 = -2$$

$$\bar{c}_6 = 0 + \pi_3 = 10$$

$x_4$  is the only variable that can enter the basis

## Passo 4: Finding the direction of improvement

Calculate the direction vector  $d$  that leads the current basic solution into the next basic solution. Solve linear system  $Bd = A_j$ , where  $A_j$  is the column of  $A$  corresponding to the entering variable.

$$Bd = \begin{bmatrix} 1 & 5 & 0 \\ 0 & 1 & 1 \\ 2 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} d_1 \\ d_3 \\ d_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix} \Rightarrow d = \begin{bmatrix} 20/11 \\ -4/11 \\ 4/11 \end{bmatrix}.$$



## Step 5: Choose a variable to leave the basis

One would like to walk in direction  $d$  as much as possible.

Determine  $\theta^* = \max\{\theta \geq 0 : \bar{x}_B - \theta \cdot d \geq 0\}$ . One of the variables that most limited  $\theta^*$  should be chosen to leave the basis. If no variable limits  $\theta^*$ , the LP is unbounded.

## Variables $x_1$ and $x_5$ are eligible to leave the basis

$$\max \theta \text{ such that } \begin{bmatrix} 10 \\ 10 \\ 2 \end{bmatrix} - \theta \begin{bmatrix} 20/11 \\ -4/11 \\ 4/11 \end{bmatrix} \geq 0 \implies \theta^* = 11/2 = 5.5$$

Assume that  $x_1$  is chosen.

## Step 6: Update $B$ and $x_B$

Basis  $B$  is updated by replacing the column of the leaving variable by the column of the entering variable. Calculate new basic solution and go to Step 2.

New basis is formed by  $x_4$ ,  $x_3$  and  $x_5$

$$B = \begin{bmatrix} 0 & 5 & 0 \\ 0 & 1 & 1 \\ 4 & -1 & 0 \end{bmatrix} \quad x_B = \begin{bmatrix} x_4 \\ x_3 \\ x_5 \end{bmatrix} = \begin{bmatrix} 5.5 \\ 12 \\ 0 \end{bmatrix}$$

$$z = 329$$

Step 2 in the next RSA iteration would obtain dual variables  $\pi = [3.9 \ 0 \ 9.5]$ .

Step 3 would calculate the following reduced costs:  $\bar{c}_1 = 1.1$ ,  $\bar{c}_2 = 3.9$ , and  $\bar{c}_6 = 9.5$ .

So, the current solution is optimal.

## Step 6: Update $B$ and $x_B$

Basis  $B$  is updated by replacing the column of the leaving variable by the column of the entering variable. Calculate new basic solution and go to Step 2.

New basis is formed by  $x_4$ ,  $x_3$  and  $x_5$

$$B = \begin{bmatrix} 0 & 5 & 0 \\ 0 & 1 & 1 \\ 4 & -1 & 0 \end{bmatrix} \quad x_B = \begin{bmatrix} x_4 \\ x_3 \\ x_5 \end{bmatrix} = \begin{bmatrix} 5.5 \\ 12 \\ 0 \end{bmatrix}$$

$$z = 329$$

Step 2 in the next RSA iteration would obtain dual variables  $\pi = [3.9 \ 0 \ 9.5]$ .

Step 3 would calculate the following reduced costs:  $\bar{c}_1 = 1.1$ ,  $\bar{c}_2 = 3.9$ , and  $\bar{c}_6 = 9.5$ .

So, the current solution is optimal.

# Advantage of Revised Simplex

Why Revised Simplex is usually better than original Simplex?

- Only Step 3 (pricing) has complexity depending on  $n$
- All other steps have complexities that only depend on  $m$

Significant advantage when  $n$  is much larger than  $m$

# A fundamental insight

- It is possible to solve LPs with not so many constraints but with a HUGE number of variables, as long as those variables have a special structure that allows their efficient pricing
- Instead of calculating the reduced cost for each individual variable, the whole pricing step should be solved as another optimization problem!

# Dantzig-Wolfe Decomposition (1960)

- Consider an LP ( $\mathcal{O}$ ) in the following format:

$$(\mathcal{O}) \max z = cx$$

subject to

$$Ax = b$$

$$Dx = d$$

$$x \geq 0$$

LP ( $\mathcal{O}$ ) has  $m$  constraints (not counting the non-negativities) and  $n$  variables; submatrix  $A$  has  $p$  rows and submatrix  $D$  has  $q$  rows

# Dantzig-Wolfe Decomposition

- Defining polyhedron  $P = \{Dx = d, x \geq 0\}$ ,  $(\mathcal{O})$  is equivalent to:

$$(\mathcal{O}') \max z = cx$$

subject to

$$Ax = b$$

$$x \in P$$

Assuming that  $P$  is limited, any solution  $x \in P$  can be represented as a convex combination of points in the set  $R$  of the extreme points of  $P$ .

- Defining polyhedron  $P = \{Dx = d, x \geq 0\}$ ,  $(\mathcal{O})$  is equivalent to:

$$(\mathcal{O}') \max z = cx$$

subject to

$$Ax = b$$

$$x \in P$$

Assuming that  $P$  is limited, any solution  $x \in P$  can be represented as a convex combination of points in the set  $R$  of the extreme points of  $P$ .



- Defining polyhedron  $P = \{Dx = d, x \geq 0\}$ ,  $(\mathcal{O})$  is equivalent to:

$$(\mathcal{O}') \max z = cx$$

subject to

$$Ax = b$$

$$x \in P$$

Assuming that  $P$  is limited, any solution  $x \in P$  can be represented as a convex combination of points in the set  $R$  of the extreme points of  $P$ .

- Any  $x \in P$  can be represented by vectors of  $|R|$  variables  $\lambda$  that correspond to convex combinations of its extreme points:

$$x = \sum_{r \in R} r \lambda_r$$

$$\sum_{r \in R} \lambda_r = 1$$

$$\lambda \geq 0$$

# Dantzig-Wolfe Decomposition

- Replacing  $x$  variables in LP ( $\mathcal{O}'$ ) by their equivalent representations in  $\lambda$  variables, one gets:

$$(\mathcal{MP}) \max z = \sum_{r \in R} (cr)\lambda_r$$

subject to

$$\sum_{r \in R} (Ar)\lambda_r = b \quad (\pi)$$

$$\sum_{r \in R} \lambda_r = 1 \quad (\nu)$$

$$\lambda \geq 0$$

New LP ( $\mathcal{MP}$ ) is called a *Master Problem* and is equivalent to LP ( $\mathcal{O}$ ). Vector  $\pi$  and scalar  $\nu$  are the dual variables of the corresponding constraints.

## Consequences of the reformulation

- $(\mathcal{MP})$  has  $p + 1$  constraints, less than the  $p + q$  constraints of  $(\mathcal{O})$
- $(\mathcal{MP})$  has many more variables (columns) than  $(\mathcal{O})$ 
  - The number of columns in  $(\mathcal{MP})$  is  $|R|$ , the number of extreme points of  $P$ . That number can be exponentially large

No matter how large is  $|R|$ , the pricing step can be efficiently performed by solving the following LP:

$$\begin{aligned} (SP) \quad & \max \bar{c} = (c - \pi A)x - \nu \\ & \text{s.t. } Dx = d \\ & \quad x \geq 0 \end{aligned}$$

# Solving the Master Problem by Column Generation

## Step 1: Initialize the Restricted Master Problem ( $\mathcal{RMP}$ )

Use a small subset of the variables in ( $\mathcal{MP}$ ), only enough to provide a feasible basis, for creating ( $\mathcal{RMP}$ ). If necessary, use artificial variables.

## Step 2: Solve the current ( $\mathcal{RMP}$ )

Besides the primal solution, also get the dual variables  $\pi$  and  $\nu$ .

## Step 3: Pricing

Solve the pricing subproblem ( $\mathcal{SP}$ ), its objective function depends on the dual solution found in Step 2. The optimal solution  $x^*$  of ( $\mathcal{SP}$ ) is a point  $r \in R$ . If  $\bar{c}^* \geq 0$ , no variable in ( $\mathcal{MP}$ ) is suitable to enter in the current basis of ( $\mathcal{RMP}$ ), so **the current solution of ( $\mathcal{RMP}$ ) is also an optimal solution for ( $\mathcal{MP}$ )** and the column generation stops.

## Step 4: Update the RMP

Add the variable  $\lambda_r$  (and its corresponding column in the matrices), associated to  $x^*$ , to ( $\mathcal{RMP}$ ). Go to Step 2.

# Solving the Master Problem by Column Generation

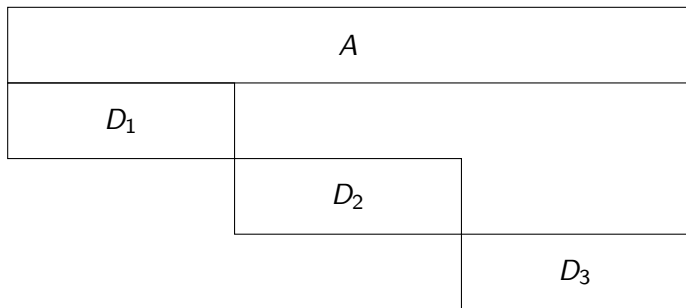
- Even if the number of variables in  $(\mathcal{MP})$  is huge, only a very small subset of its variables is likely to be added to  $(\mathcal{RMP})$ .

So, Column Generation can solve  $(\mathcal{MP})$



# Is it worthy to use DW decomposition for solving an LP?

- DW actually believed that it would work for problems with **block-diagonal** structure, where the pricing subproblem decomposes into many independent LPs. Those smaller LPs would be easier to solve, especially if they have a nice particular structure (like defining network flows, as in Ford Jr and Fulkerson (1958)).



## Usually not!

- Even when the LP has a block-diagonal structure, it is usually faster to solve ( $\mathcal{O}$ ) directly than to apply DW decomposition and solve ( $\mathcal{MP}$ ) by Column Generation
- In fact, top solvers like CPLEX, Gurobi and XPRESS do not even offer DW decomposition for LP

- An Integer Program (IP) has the following format:

$$\begin{aligned} (\mathcal{IP}) \quad & \max z = cx \\ & \text{s.t. } Ax = b \\ & x \in \mathbb{Z}_+^n \end{aligned}$$

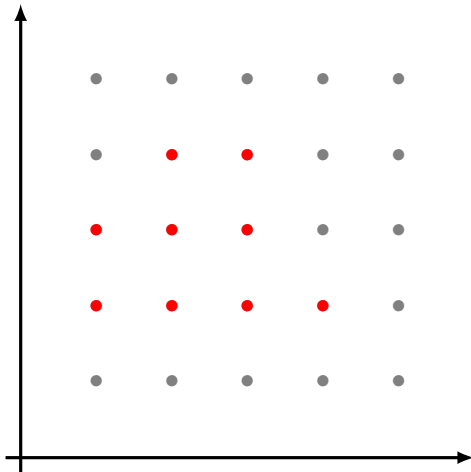
An IP is an LP with additional integrality constraints over the variables.

An IP often arises as a formulation for a Combinatorial Optimization Problem (COP)

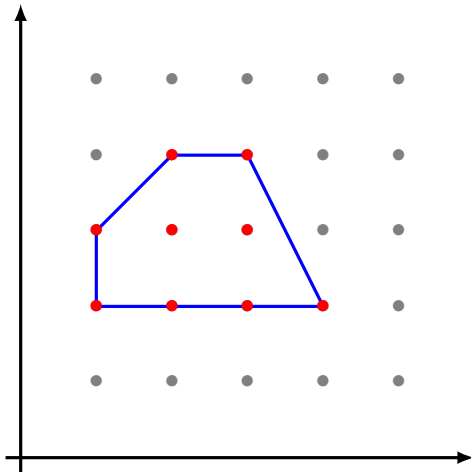
- Each solution of the COP is mapped into an integer point in the  $n$ -dimensional space
- Let  $X$  be the set of those points

If it was possible to know all inequalities that define  $\text{Conv}(X)$ , the COP could be solved as simple LP

# Set of integer points and its convex hull



# Set of integer points and its convex hull

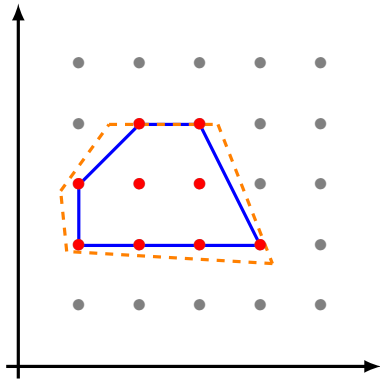
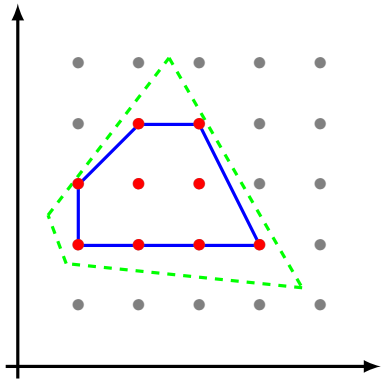


- If a COP is NP-hard, there is no efficient way to separate all inequalities that define  $\text{Conv}(X)$  (unless  $P = NP$ )
- In practice, one defines a **formulation**, a set of inequalities that contain all points in  $X$ , but no integer point not in  $X$
- The same COP can have many different formulations.

## Branch-and-Bound algorithm

- Work with the linear relaxation of an IP. Better formulations lead to smaller gaps (difference between the LP value to the IP value). That integrality gap has an exponential impact on the size of the B&B tree

# Two possible formulations for the same set $X$





$$\begin{array}{ll} (\mathcal{O}) \max cx & \\ \text{s.t. } Ax = b & \\ Dx = d & \implies \\ x \in \mathbb{Z}_+^n & \end{array} \quad \begin{array}{ll} (\mathcal{O}') \max cx & \\ \text{s.t. } Ax = b & \\ x \in P & \end{array}$$

$$P = \{Dx = d, x \in \mathbb{Z}_+^n\}$$

is assumed to be a finite set of points numbered from  $p_1$  to  $p_Q$ .

- A point  $x \in P$  can be (trivially) described as an integer convex combination of those  $Q$  points:

$$x = \sum_{j=1}^Q p_j \lambda_j$$

$$\sum_{j=1}^Q \lambda_j = 1$$

$$\lambda \in \{0, 1\}^Q$$

- Replacing  $x$  in  $(\mathcal{O}')$  by its equivalent, the following Integer Master Problem is obtained:

$$(\mathcal{IMP}) \max z = \sum_{j=1}^Q (cp_j)\lambda_j$$

subject to

$$\sum_{j=1}^Q (Ap_j)\lambda_j = b$$

$$\sum_{j=1}^Q \lambda_j = 1$$

$$\lambda \in \{0, 1\}^Q$$

- The linear relaxation of  $(\mathcal{IMP})$  is the following Master LP:

$$(\mathcal{MP}) \max z = \sum_{j=1}^Q (cp_j) \lambda_j$$

subject to

$$\sum_{j=1}^Q (Ap_j) \lambda_j = b \quad (\pi)$$

$$\sum_{j=1}^Q \lambda_j = 1 \quad (\nu)$$

$$\lambda \geq 0$$

## Consequences of the reformulation

- $(MP)$  usually has a huge number of variables. Yet, it can be solved by column generation
- **The value of  $(MP)$  can be better than the linear relaxation of  $(O)$ !**
- This may happen because the integrality is not relaxed in the subproblem:

$$\begin{aligned} (SP) \quad \max \quad & \bar{c} = (c - \pi A)x - \nu \\ \text{s.t.} \quad & Dx = d \\ & x \in \mathbb{Z}_+^n \end{aligned}$$

# Dantzig-Wolfe decomposition for IP

$$\max z = \sum_{j=1}^Q (cp_j) \lambda_j$$

$$\text{s.t. } \sum_{j=1}^Q (Ap_j) \lambda_j = b$$

$$\sum_{j=1}^Q \lambda_j = 1$$

$$\lambda \geq 0$$

$\Leftrightarrow$

$$\max cx$$

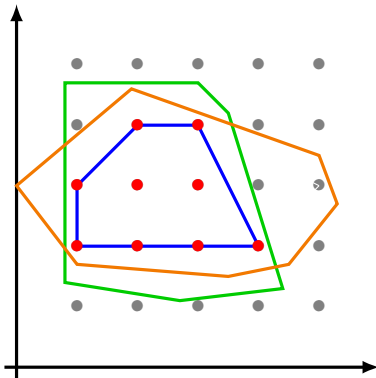
$$\text{s.t. } Ax = b$$

$$x \in \text{Conv}\{Dx = d, x \in \mathbb{Z}_+^n\}$$

The reformulation is equivalent to convexifying part of the constraints in  $(\mathcal{O})$

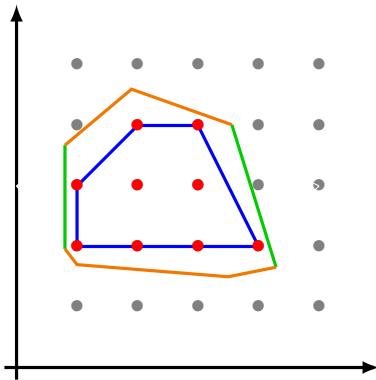
# How DW decomposition for IP improves a formulation

Suppose we partition the set of constraints of an Original Formulation into two sets: green and orange



# How DW decomposition for IP improves a formulation

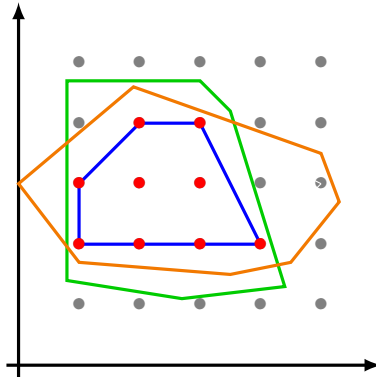
The Original Formulation is the intersection of the two sets





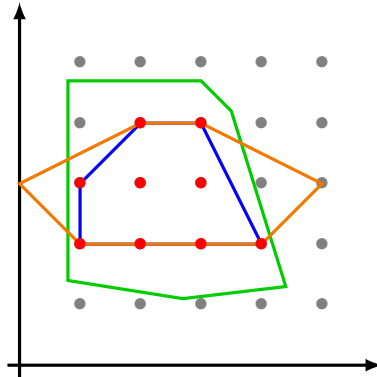
# How DW decomposition for IP improves a formulation

Convexifying the orange constraints: obtaining the convex hull of the integer points in that set



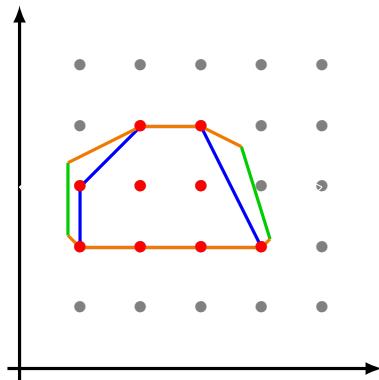
# How DW decomposition for IP improves a formulation

Convexifying the orange constraints: obtaining the convex hull of the integer points in that set



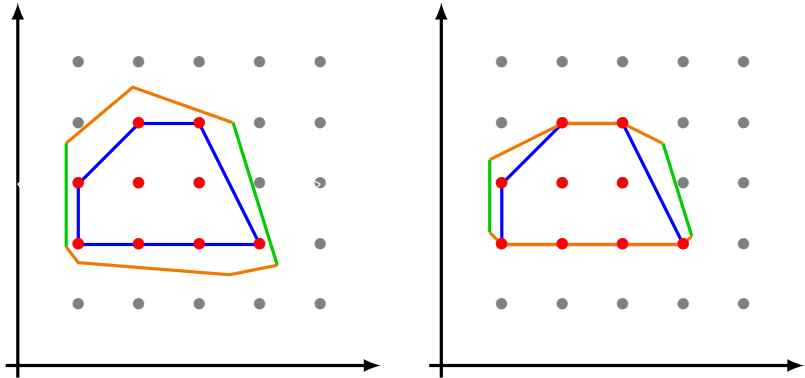
# How DW decomposition for IP improves a formulation

New Improved Formulation



# How DW decomposition for IP improves a formulation

Original formulation vs Improved formulation



# DW decomposition with multiple subproblems

- Consider an IP decomposable into  $K$  independent subproblems:

$$\min c^1 x^1 + c^2 x^2 + \dots + c^K x^K$$

subject to

$$A^1 x^1 + A^2 x^2 + \dots + A^K x^K = b$$

$$D^k x^k = d^k \quad k = 1, \dots, K$$

$$x^k \in Z_+^{n^k} \quad k = 1, \dots, K$$

For each  $k = 1, \dots, K$ ,  $A^k$  is a  $p \times n^k$  matrix,  $D^k$  is a  $q^k \times n^k$  matrix; the remaining vectors have compatible dimensions

# DW decomposition with multiple subproblems

- Consider an IP decomposable into  $K$  independent subproblems:

$$\min c^1 x^1 + c^2 x^2 + \dots + c^K x^K$$

subject to

$$A^1 x^1 + A^2 x^2 + \dots + A^K x^K = b$$

$$D^k x^k = d^k \quad k = 1, \dots, K$$

$$x^k \in Z_+^{n^k} \quad k = 1, \dots, K$$

When  $K = 1$  we have the case already considered

$$\min cx$$

$$\text{S.t. } Ax = b$$

$$Dx = d$$

$$x \in Z_+^n$$

# DW decomposition with multiple subproblems

- Consider an IP decomposable into  $K$  independent subproblems:

$$\min c^1 x^1 + c^2 x^2 + \dots + c^K x^K$$

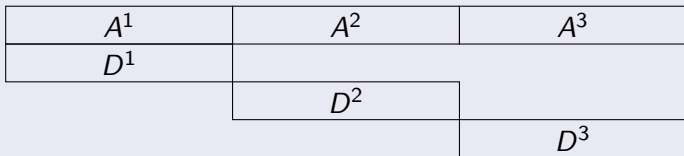
subject to

$$A^1 x^1 + A^2 x^2 + \dots + A^K x^K = b$$

$$D^k x^k = d^k \quad k = 1, \dots, K$$

$$x^k \in Z_+^{n^k} \quad k = 1, \dots, K$$

When  $K > 1$  the problem is said to have a block-diagonal structure



# DW with Multiple subproblems

$$\min c^1 x^1 + c^2 x^2 + \dots + c^K x^K$$

subject to

$$A^1 x^1 + A^2 x^2 + \dots + A^K x^K = b$$

$$x^k \in P^k \quad k = 1, \dots, K$$

$$P^k = \{D^k x^k = d^k, x \in \mathbb{Z}_+^{n^k}\}$$

is assumed a finite set of points numbered from  $p_1^k$  to  $p_{Q_k}^k$ .



# DW with Multiple subproblems

$$\min c^1 x^1 + c^2 x^2 + \dots + c^K x^K$$

subject to

$$A^1 x^1 + A^2 x^2 + \dots + A^K x^K = b$$

$$x^k \in P^k \quad k = 1, \dots, K$$

A point  $x^k \in P^k$  can be described as:

$$x^k = \sum_{j=1}^{Q_k} p_j^k \lambda_j^k$$

$$\sum_{j=1}^{Q_k} \lambda_j^k = 1$$

$$\lambda \in \{0, 1\}^{Q_k}$$

# Dantzig-Wolfe decomposition for IP

- Replacing every  $x^k$  by its equivalent, the following Integer Master Problem is obtained:

$$(\mathcal{IMP}) \min z = \sum_{k=1}^K \sum_{j=1}^{Q^k} (c^k p_j^k) \lambda_j^k$$

subject to

$$\sum_{k=1}^K \sum_{j=1}^{Q^k} (A^k p_j^k) \lambda_j^k = b$$

$$\sum_{j=1}^{Q^k} \lambda_j^k = 1 \quad k = 1, \dots, K$$

$$\lambda \in \{0, 1\}^{Q^k} \quad k = 1, \dots, K$$

- The linear relaxation of  $(\mathcal{IMP})$  is the following Master LP Problem:

$$(\mathcal{MP}) \min z = \sum_{k=1}^K \sum_{j=1}^{Q^k} (c^k p_j^k) \lambda_{j^k}$$

subject to

$$\sum_{k=1}^K \sum_{j=1}^{Q^k} (A^k p_j^k) \lambda_j^k = b \quad (\pi)$$

$$\sum_{j=1}^{Q^k} \lambda_j^k = 1 \quad k = 1, \dots, K \quad (\nu^k)$$

$$\lambda \geq 0 \quad k = 1, \dots, K$$

# A remark on the Master LP Problem

- For the subproblems where  $x^k = 0$  is a solution, it is possible to relax the corresponding convexity constraint to  $\leq 1$ . If all subproblems have that property, we can write:

$$(\mathcal{MP}) \min z = \sum_{k=1}^K \sum_{j=1}^{Q^k} (c^k p_j^k) \lambda_j^k$$

subject to

$$\sum_{k=1}^K \sum_{j=1}^{Q^k} (A^k p_j^k) \lambda_j^k = b \quad (\pi)$$

$$\sum_{j=1}^{Q^k} \lambda_j^k \leq 1 \quad k = 1, \dots, K \quad (\nu^k)$$

$$\lambda \geq 0 \quad k = 1, \dots, K$$

- For each  $k = 1, \dots, K$ , there is a pricing subproblem:

$$\begin{aligned} (\mathcal{SP}^k) \quad & \min \bar{c}^k = (c^k - \pi A^k)x^k - \nu^k \\ & \text{s.t. } D^k x^k = d^k \\ & \quad x^k \in \mathbb{Z}_+^{n^k} \end{aligned}$$

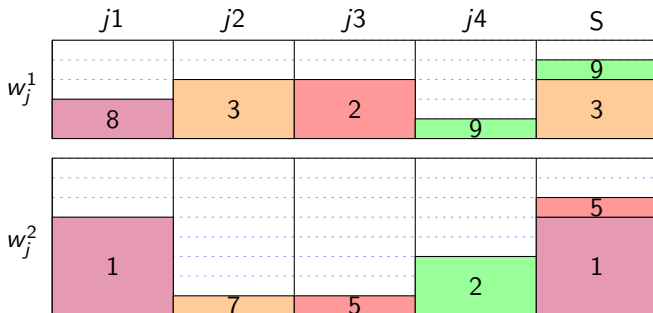
- The restricted master LP is optimal when  $\bar{c}^k \geq 0$ , for  $k = 1, \dots, K$

# Generalized Assignment Problem (GAP)

- Set  $J$  of tasks; set  $K$  of machines; capacity  $W^k$ ,  $k \in K$ ; assignment cost  $c_j^k$  and load  $w_j^k$ ,  $k \in K$ ,  $j \in J$
- Find an assignment of tasks to machines such that the total load in each machine does not exceed its capacity, with minimum total cost

# Example of instance

jobs	cost ( $c_j^k$ )				load ( $w_j^k$ )				$W_k$	
	1	2	3	4	1	2	3	4		
machines	<b>1</b>	8	3	2	9	2	3	3	1	5
	<b>2</b>	1	7	5	2	5	1	1	3	8



Optimal solution value: 18

# Generalized Assignment Problem (GAP)

- Set  $J$  of tasks; set  $K$  of machines; capacity  $W^k$ ,  $k \in K$ ; assignment cost  $c_j^k$  and load  $w_j^k$ ,  $k \in K$ ,  $j \in J$
- Find an assignment of tasks to machines such that the total load in each machine does not exceed its capacity, with minimum total cost

Original formulation ( $\mathcal{O}$ ):

$$\text{Min } z = \sum_{k \in K} \sum_{j \in J} c_j^k x_j^k \quad (1a)$$

$$\text{S.t. } \sum_{k \in K} x_j^k = 1, \quad j \in J; \quad (1b)$$

$$\sum_{j \in J} w_j^k x_j^k \leq W^k, \quad k \in K; \quad (1c)$$

$$x_j^k \in \{0, 1\}, \quad j \in J, k \in K. \quad (1d)$$



# Example: Generalized Assignment Problem (GAP)

		cost ( $c_j^k$ )				load ( $w_j^k$ )				$W^k$
jobs		<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	
machines	<b>1</b>	8	3	2	9	2	3	3	1	5
	<b>2</b>	1	7	5	2	5	1	1	3	8

# Example: Generalized Assignment Problem (GAP)

		cost ( $c_j^k$ )				load ( $w_j^k$ )				$W^k$
jobs		1	2	3	4	1	2	3	4	
machines	1	8	3	2	9	2	3	3	1	5
	2	1	7	5	2	5	1	1	3	8

Original formulation ( $\mathcal{O}$ ):

$$\begin{aligned}
 \text{Min } z_{IP} &= 8x_1^1 + 3x_2^1 + 2x_3^1 + 9x_4^1 + x_1^2 + 7x_2^2 + 5x_3^2 + 2x_4^2 \\
 \text{S.t.} \quad &x_1^1 + x_1^2 = 1 \\
 &x_2^1 + x_2^2 = 1 \\
 &x_3^1 + x_3^2 = 1 \\
 &x_4^1 + x_4^2 = 1 \\
 &2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\
 &5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\
 &0 \leq x \leq 1 \\
 &x \in \mathbb{Z}^8
 \end{aligned}$$

# Example: Generalized Assignment Problem (GAP)

Linear relaxation of the original formulation:

$$\begin{array}{rcl} \text{Min } z_{IP} & = & 8x_1^1 + 3x_2^1 + 2x_3^1 + 9x_4^1 + x_1^2 + 7x_2^2 + 5x_3^2 + 2x_4^2 \\ \text{S.t.} & & x_1^1 + x_1^2 = 1 \\ & & x_2^1 + x_2^2 = 1 \\ & & x_3^1 + x_3^2 = 1 \\ & & x_4^1 + x_4^2 = 1 \\ & & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & & x \geq 0 \end{array}$$

# Example: Generalized Assignment Problem (GAP)

Linear relaxation of the original formulation:

$$\begin{array}{rllllllll} \text{Min } z_{IP} = & 8x_1^1 & + & 3x_2^1 & + & 2x_3^1 & + & 9x_4^1 & + & x_1^2 & + & 7x_2^2 & + & 5x_3^2 & + & 2x_4^2 \\ \text{S.t.} & x_1^1 & & & & & & & & + & x_1^2 & & & & & & & & = & 1 \\ & & & x_2^1 & & & & & & & & + & x_2^2 & & & & & & & = & 1 \\ & & & & & x_3^1 & & & & & & & & + & x_3^2 & & & & & = & 1 \\ & & & & & & & x_4^1 & & & & & & & & + & x_4^2 & & & = & 1 \\ & 2x_1^1 & + & 3x_2^1 & + & 3x_3^1 & + & x_4^1 & & & & & & & & & & & & \leq & 5 \\ & & & & & & & & 5x_1^2 & + & x_2^2 & + & x_3^2 & + & 3x_4^2 & & & & \leq & 8 \\ & & & & & & & & & & & & & & & & & & & x & \geq & 0 \end{array}$$

$$\begin{array}{rccccc} z = 9.69 & x_1^1 = 0.077 & x_2^1 = 1 & x_3^1 = 0.615 & x_4^1 = 0 \\ & x_1^2 = 0.923 & x_2^2 = 0 & x_3^2 = 0.385 & x_4^2 = 1 \end{array}$$

# Applying DW decomposition to $(\mathcal{O})$

- Let  $P^k = \{p_1^k, p_2^k, \dots, p_{Q^k}^k\}$  be the set of all possible allocations of tasks to machine  $k$ .
- $p_q^k = (p_{q1}^k, p_{q2}^k, \dots, p_{q|J|}^k)$  is a feasible solution to:

$$\sum_{j \in J} w_j^k p_{qj}^k \leq W^k \quad (2a)$$

$$p_{qj}^k \in \{0, 1\}, \quad j \in J \quad (2b)$$

- Let  $\lambda_q^k, k \in K, q = 1, \dots, Q^k$ , be a binary variable indicating whether allocation  $p_q^k$  is selected to machine  $k$

Resulting Integer Master problem:

$$\text{Min } z = \sum_{k \in K} \sum_{q \in P^k} \left( \sum_{j \in J} c_j^k p_{qj}^k \right) \lambda_q^k \quad (3a)$$

$$\text{S.t. } \sum_{k \in K} \sum_{q \in P^k} p_{qj}^k \lambda_q^k = 1, \quad j \in J; \quad (3b)$$

$$\sum_{q \in P^k} \lambda_q^k \leq 1, \quad k \in K; \quad (3c)$$

$$\lambda_q^k \in \{0, 1\}, \quad k \in K, q = 1, \dots, Q^k \quad (3d)$$

Master LP:

$$\text{Min } z = \sum_{k \in K} \sum_{q \in P^k} \left( \sum_{j \in J} c_j^k p_{aj}^k \right) \lambda_q^k \quad (4a)$$

$$\text{S.t. } \sum_{k \in K} \sum_{q \in P^k} p_{aj}^k \lambda_q^k = 1, \quad j \in J; \quad (4b)$$

$$\sum_{q \in P^k} \lambda_q^k \leq 1, \quad k \in K; \quad (4c)$$

$$\lambda_q^k \geq 0, \quad k \in K, q = 1, \dots, Q^k \quad (4d)$$

The pricing subproblems are, for each machine  $k$ , solve the following binary knapsack problem:

$$\text{Min } \bar{c}^k = \sum_{j=1}^J (c_j^k - \pi_j) x_j^k - \nu^k \quad (5a)$$

$$\text{S.t. } \sum_{j \in J} w_j^k x_j^k \leq W^k, \quad (5b)$$

$$x_j^k \in \{0, 1\}, \quad j \in J, \quad (5c)$$

where  $\pi_j$  and  $\nu^k$  are the dual variables associated to Constraints (4b) and (4c), respectively.

- The Binary Knapsack Problem is (weakly) NP-hard, but extremely well solved in practice (see Pferschy et al. (2004))



# Example: Generalized Assignment Problem (GAP)

jobs	cost ( $c_j^k$ )				load ( $w_j^k$ )				$W^k$	
	1	2	3	4	1	2	3	4		
machines	1	8	3	2	9	2	3	3	1	5
	2	1	7	5	2	5	1	1	3	8

$p_q^k$	$P^1$									$P^2$										
	$p_1^1$	$p_2^1$	$p_3^1$	$p_4^1$	$p_5^1$	$p_6^1$	$p_7^1$	$p_8^1$	$p_9^1$	$p_1^2$	$p_2^2$	$p_3^2$	$p_4^2$	$p_5^2$	$p_6^2$	$p_7^2$	$p_8^2$	$p_9^2$	$p_{10}^2$	$p_{11}^2$
$p_{q1}^k$	1	0	0	0	1	1	1	0	0	1	0	0	0	1	1	1	0	0	0	1
$p_{q2}^k$	0	1	0	0	1	0	0	1	0	0	1	0	0	1	1	0	1	1	0	0
$p_{q3}^k$	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	1	1	1	1	0
$p_{q4}^k$	0	0	0	1	0	0	1	1	1	0	0	0	1	0	0	0	1	1	1	1
cost	8	3	2	9	11	10	17	12	11	1	7	5	2	8	13	6	12	14	7	3
$\lambda_q^k$	$\lambda_1^1$	$\lambda_2^1$	$\lambda_3^1$	$\lambda_4^1$	$\lambda_5^1$	$\lambda_6^1$	$\lambda_7^1$	$\lambda_8^1$	$\lambda_9^1$	$\lambda_1^2$	$\lambda_2^2$	$\lambda_3^2$	$\lambda_4^2$	$\lambda_5^2$	$\lambda_6^2$	$\lambda_7^2$	$\lambda_8^2$	$\lambda_9^2$	$\lambda_{10}^2$	$\lambda_{11}^2$

# Example: Generalized Assignment Problem (GAP)

jobs	cost ( $c_j^k$ )				load ( $w_j^k$ )				$W^k$	
	1	2	3	4	1	2	3	4		
machines	1	8	3	2	9	2	3	3	1	5
	2	1	7	5	2	5	1	1	3	8

## Integer Master Problem

$$\begin{aligned}
 \text{Min } z &= 8\lambda_1^1 + 3\lambda_2^1 + 2\lambda_3^1 + 9\lambda_4^1 + \dots + 13\lambda_6^2 + 6\lambda_7^2 + 12\lambda_8^2 + 14\lambda_9^2 + 7\lambda_{10}^2 \\
 &\quad \lambda_1^1 + \dots + \lambda_6^2 + \lambda_7^2 = 1 \\
 &\quad \lambda_2^1 + \dots + \lambda_6^2 + \lambda_8^2 + \lambda_9^2 = 1 \\
 &\quad \lambda_3^1 + \dots + \lambda_6^2 + \lambda_7^2 + \lambda_8^2 + \lambda_9^2 + \lambda_{10}^2 = 1 \\
 &\quad \lambda_4^1 + \dots + \lambda_9^2 + \lambda_{10}^2 = 1 \\
 &\quad \lambda_1^1 \quad \lambda_2^1 \quad \lambda_3^1 \quad \lambda_4^1 + \dots \leq 1 \\
 &\quad \dots + \lambda_6^2 + \lambda_7^2 + \lambda_8^2 + \lambda_9^2 + \lambda_{10}^2 \leq 1 \\
 &\quad \lambda \in \{0, 1\}
 \end{aligned}$$

# Example: Generalized Assignment Problem (GAP)

jobs	cost ( $c_j^k$ )				load ( $w_j^k$ )				$W^k$	
	1	2	3	4	1	2	3	4		
machines	1	8	3	2	9	2	3	3	1	5
	2	1	7	5	2	5	1	1	3	8

Relaxing the integrality  $\Rightarrow$  (MP)

$$\begin{aligned}
 \text{Min } z &= 8\lambda_1^1 + 3\lambda_2^1 + 2\lambda_3^1 + 9\lambda_4^1 + \dots + 13\lambda_6^2 + 6\lambda_7^2 + 12\lambda_8^2 + 14\lambda_9^2 + 7\lambda_{10}^2 \\
 \lambda_1^1 & \qquad \qquad \qquad + \dots + \lambda_6^2 + \lambda_7^2 &= 1 \\
 \lambda_2^1 & \qquad \qquad \qquad + \dots + \lambda_6^2 \quad + \quad \lambda_8^2 + \lambda_9^2 &= 1 \\
 \lambda_3^1 & \qquad \qquad \qquad + \dots + \lambda_6^2 + \lambda_7^2 + \lambda_8^2 + \lambda_9^2 + \lambda_{10}^2 &= 1 \\
 \lambda_4^1 & \qquad \qquad \qquad + \dots + \lambda_9^2 + \lambda_{10}^2 &= 1 \\
 \lambda_1^1 \quad \lambda_2^1 \quad \lambda_3^1 \quad \lambda_4^1 & + \dots &\leq 1 \\
 & \dots + \lambda_6^2 + \lambda_7^2 + \lambda_8^2 + \lambda_9^2 + \lambda_{10}^2 &\leq 1 \\
 & \lambda \geq 0
 \end{aligned}$$

# Solving the ( $\mathcal{MP}$ ) by Column Generation

Iteration 1: ( $\mathcal{RMP}$ ) initialized with artificial variables  
Restricted Master Problem

$$\begin{array}{rcll} \text{Min } z = 99\mu_1 + 99\mu_2 + 99\mu_3 + 99\mu_4 & & & \\ \mu_1 & & & = 1 \quad (\pi_1 = 99) \\ & \mu_2 & & = 1 \quad (\pi_2 = 99) \\ & & \mu_3 & = 1 \quad (\pi_3 = 99) \\ & & & \mu_4 = 1 \quad (\pi_4 = 99) \\ & & & \leq 1 \quad (\nu^1 = 0) \\ & & & \leq 1 \quad (\nu^2 = 0) \\ & \lambda & \geq 0 & \end{array}$$

$$z = 396; \mu_1 = 1, \mu_2 = 1, \mu_3 = 1, \mu_4 = 1$$

# Solving the ( $\mathcal{MP}$ ) by Column Generation

## Iteration 1: Restricted Master Problem

$$\begin{aligned} \text{Min } z &= 99\mu_1 + 99\mu_2 + 99\mu_3 + 99\mu_4 \\ \mu_1 &= 1 \quad (\pi_1 = 99) \\ \mu_2 &= 1 \quad (\pi_2 = 99) \\ \mu_3 &= 1 \quad (\pi_3 = 99) \\ \mu_4 &= 1 \quad (\pi_4 = 99) \\ &\leq 1 \quad (\nu^1 = 0) \\ &\leq 1 \quad (\nu^2 = 0) \\ \lambda &\geq 0 \end{aligned}$$

$$z = 396; \mu_1 = 1, \mu_2 = 1, \mu_3 = 1, \mu_4 = 1$$

$$\text{Subproblem 1: } \min \sum_{j \in J} (c_j^1 - \pi_j) x_j^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 &= -91x_1^1 - 96x_2^1 - 97x_3^1 - 90x_4^1 \\ \text{S.t. } &2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ &x \in \{0, 1\} \end{aligned}$$

$$S = (1, 0, 1, 0) \text{ and } \bar{c}^1 = -188; S \Leftrightarrow \lambda_6^1$$

$$\text{Subproblem 2: } \min \sum_{j \in J} (c_j^2 - \pi_j) x_j^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 &= -97x_1^2 - 92x_2^2 - 94x_3^2 - 97x_4^2 \\ \text{S.t. } &5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ &x \in \{0, 1\} \end{aligned}$$

$$S = (1, 1, 1, 0) \text{ and } \bar{c}^2 = -284; S \Leftrightarrow \lambda_6^2$$

# Solving the ( $\mathcal{MP}$ ) by Column Generation

## Iteration 2: Restricted Master Problem

$$\begin{aligned} \text{Min } z = & 99\mu_1 + 99\mu_2 + 99\mu_3 + 99\mu_4 + 10\lambda_1 + 13\lambda_2 \\ & \mu_1 \qquad \qquad \qquad + \lambda_1 + \lambda_2 = 1 \quad (\pi_1 = 0) \\ & \qquad \mu_2 \qquad \qquad \qquad + \lambda_2 = 1 \quad (\pi_2 = 3) \\ & \qquad \qquad \mu_3 \qquad + \lambda_1 + \lambda_2 = 1 \quad (\pi_3 = 10) \\ & \qquad \qquad \qquad \mu_4 \qquad \qquad = 1 \quad (\pi_4 = 99) \\ & \qquad \qquad \qquad \qquad \lambda_1 \qquad \leq 1 \quad (\nu^1 = 0) \\ & \qquad \qquad \qquad \qquad \lambda_2 \leq 1 \quad (\nu^2 = 0) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 112$$

$$\text{Subproblem 1: } \min \sum_{j \in J} (c_j^1 - \pi_j)x_j^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 = & 8x_1^1 - 8x_3^1 - 90x_4^1 \\ \text{S.t. } & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (0, 0, 1, 1) \text{ and } \bar{c}^1 = -98; S \Leftrightarrow \lambda_9^1$$

$$\text{Subproblem 2: } \min \sum_{j \in J} (c_j^2 - \pi_j)x_j^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 = & x_1^2 + 4x_2^2 - 5x_3^2 - 97x_4^2 \\ \text{S.t. } & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (0, 0, 1, 1) \text{ and } \bar{c}^2 = -102; S \Leftrightarrow \lambda_{10}^2$$

# Solving the ( $\mathcal{MP}$ ) by Column Generation

## Iteration 3: Restricted Master Problem

$$\begin{aligned} \text{Min } z = & 99\mu_1 + 99\mu_2 + 99\mu_3 + 99\mu_4 + 10\lambda_1 + 13\lambda_2 + 11\lambda_3 + 7\lambda_4 \\ & \mu_1 \qquad \qquad \qquad + \lambda_1 + \lambda_2 \qquad \qquad \qquad = 1 \quad (\pi_1 = 10) \\ & \qquad \mu_2 \qquad \qquad \qquad + \lambda_2 \qquad \qquad \qquad = 1 \quad (\pi_2 = 7) \\ & \qquad \qquad \mu_3 \qquad + \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1 \quad (\pi_3 = 0) \\ & \qquad \qquad \qquad \mu_4 \qquad \qquad + \lambda_3 + \lambda_4 = 1 \quad (\pi_4 = 11) \\ & \qquad \qquad \qquad \qquad \lambda_1 \qquad + \lambda_3 \qquad \leq 1 \quad (\nu^1 = 0) \\ & \qquad \qquad \qquad \qquad \lambda_2 \qquad + \lambda_4 \leq 1 \quad (\nu^2 = -4) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 24$$

$$\text{Subproblem 1: } \min \sum_{j \in J} (c_j^1 - \pi_j) x_j^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 &= -2x_1^1 - 4x_2^1 + 2x_3^1 - 2x_4^1 \\ \text{S.t. } & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (1, 1, 0, 0) \text{ and } \bar{c}^1 = -6; S \Leftrightarrow \lambda_1^1$$

$$\text{Subproblem 2: } \min \sum_{j \in J} (c_j^2 - \pi_j) x_j^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 &= -9x_1^2 + 5x_3^2 - 9x_4^2 + 4 \\ \text{S.t. } & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (1, 0, 0, 1) \text{ and } \bar{c}^2 = -14; S \Leftrightarrow \lambda_{11}^2$$

# Solving the ( $\mathcal{MP}$ ) by Column Generation

## Iteration 4: Restricted Master Problem

$$\begin{aligned} \text{Min } z = & \dots 10\lambda_1 + 13\lambda_2 + 11\lambda_3 + 7\lambda_4 + 11\lambda_5 + 3\lambda_6 \\ \dots & \lambda_1 + \lambda_2 + \lambda_5 + \lambda_6 = 1 \quad (\pi_1 = 2) \\ \dots & \quad + \lambda_2 + \lambda_5 = 1 \quad (\pi_2 = 9) \\ \dots & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1 \quad (\pi_3 = 6) \\ \dots & \quad + \lambda_3 + \lambda_4 + \lambda_6 = 1 \quad (\pi_4 = 5) \\ \dots & \lambda_1 + \lambda_3 + \lambda_5 \leq 1 \quad (\nu^1 = 0) \\ \dots & \quad \lambda_2 + \lambda_4 + \lambda_6 \leq 1 \quad (\nu^2 = -4) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 18$$

$$\text{Subproblem 1: } \min \sum_{j \in J} (c_j^1 - \pi_j) x_j^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 &= 6x_1^1 - 6x_2^1 - 4x_3^1 + 4x_4^1 \\ \text{S.t. } & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (0, 1, 0, 0) \text{ and } \bar{c}^1 = -6; S \Leftrightarrow \lambda_2^1$$

$$\text{Subproblem 2: } \min \sum_{j \in J} (c_j^2 - \pi_j) x_j^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 &= -x_1^2 - 2x_2^2 - x_3^2 - 3x_4^2 + 4 \\ \text{S.t. } & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (0, 1, 1, 1) \text{ and } \bar{c}^2 = -2; S \Leftrightarrow \lambda_9^2$$



# Solving the ( $\mathcal{MP}$ ) by Column Generation

## Iteration 5: Restricted Master Problem

$$\begin{aligned} \text{Min } z = & \dots 10\lambda_1 + 13\lambda_2 + 11\lambda_3 + 7\lambda_4 + 11\lambda_5 + 3\lambda_6 + 3\lambda_7 + 14\lambda_8 \\ \dots & \lambda_1 + \lambda_2 + \lambda_5 + \lambda_6 = 1 \quad (\pi_1 = 5) \\ \dots & \quad + \lambda_2 + \lambda_5 + \lambda_7 + \lambda_8 = 1 \quad (\pi_2 = 7) \\ \dots & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_8 = 1 \quad (\pi_3 = 9) \\ \dots & \quad + \lambda_3 + \lambda_4 + \lambda_6 + \lambda_8 = 1 \quad (\pi_4 = 6) \\ \dots & \lambda_1 + \lambda_3 + \lambda_5 + \lambda_7 \leq 1 \quad (\nu^1 = -4) \\ \dots & \quad \lambda_2 + \lambda_4 + \lambda_6 + \lambda_8 \leq 1 \quad (\nu^2 = -8) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 15$$

$$\text{Subproblem 1: } \min \sum_{j \in J} (c_j^1 - \pi_j)x_j^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 &= 3x_1^1 - 4x_2^1 - 7x_3^1 + 3x_4^1 + 4 \\ \text{S.t. } & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & x \in \{0, 1\} \end{aligned}$$

$$S = (0, 0, 1, 0) \text{ and } \bar{c}^1 = -3; S \Leftrightarrow \lambda_3^1$$

$$\text{Subproblem 2: } \min \sum_{j \in J} (c_j^2 - \pi_j)x_j^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 &= -4x_1^2 - 4x_3^2 - 4x_4^2 + 8 \\ \text{S.t. } & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & x \in \{0, 1\} \end{aligned}$$

$$\bar{c}^2 = 0$$

# Solving the ( $\mathcal{MP}$ ) by Column Generation

## Iteration 6: Restricted Master Problem

$$\begin{aligned} \text{Min } z = & \dots 10\lambda_1 + 13\lambda_2 + 11\lambda_3 + 7\lambda_4 + 11\lambda_5 + 3\lambda_6 + 3\lambda_7 + 14\lambda_8 + 2\lambda_9 \\ \dots & \lambda_1 + \lambda_2 + \lambda_5 + \lambda_6 = 1 \quad (\pi_1 = 8) \\ \dots & \quad + \lambda_2 + \lambda_5 + \lambda_7 + \lambda_8 = 1 \quad (\pi_2 = 10) \\ \dots & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_8 + \lambda_9 = 1 \quad (\pi_3 = 9) \\ \dots & \quad + \lambda_3 + \lambda_4 + \lambda_6 + \lambda_8 = 1 \quad (\pi_4 = 9) \\ \dots & \lambda_1 + \lambda_3 + \lambda_5 + \lambda_7 + \lambda_9 \leq 1 \quad (\nu^1 = -7) \\ \dots & \quad \lambda_2 + \lambda_4 + \lambda_6 + \lambda_8 \leq 1 \quad (\nu^2 = -14) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 15$$

$$\text{Subproblem 1: } \min \sum_{j \in J} (c_j^1 - \pi_j) x_j^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 = & -7x_2^1 - 7x_3^1 + 7 \\ \text{S.t. } & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & x \in \{0, 1\} \end{aligned}$$

$$\bar{c}^1 = 0$$

$$\text{Subproblem 2: } \min \sum_{j \in J} (c_j^2 - \pi_j) x_j^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 = & -7x_1^2 - 3x_2^2 - 4x_3^2 - 7x_4^2 + 14 \\ \text{S.t. } & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & x \in \{0, 1\} \end{aligned}$$

$$\bar{c}^2 = 0$$

$\bar{c} \geq 0$  for all subproblems  $\Rightarrow$  **Optimal ( $\mathcal{MP}$ ) Solution**

# Combining Column Generation with Cutting Planes: Robust vs Non-Robust

## Definition

**Robust Cut, Robust Branch-Cut-and-Price Algorithm (BCPA).** A cutting plane in a BCPA is *robust* if it does not force any change in the structure of the pricing subproblems in subsequent calls to the Column Generation algorithm. A cutting plane that does force changes in the pricing structure is *non-robust*. A BCPA that only performs robust branchings and only separates robust cuts is said to be *robust*, otherwise, it is *non-robust*.

A fractional solution  $\lambda^*$  to  $(\mathcal{MP})$  can be converted into a solution  $x^*$ , using:

$$x^* = \sum_{j=1}^Q p_j \lambda_j^*.$$

We may separate a valid inequality  $\alpha x \geq \alpha_0$  cutting that point. Then, it can be translated back to

$$\sum_{q=1}^Q \left( \sum_{j=1}^n \alpha_j p_{qj} \right) \lambda_q \geq \alpha_0. \quad (6)$$

The dual variable of the new cut is included in the  $\pi$  vector, its  $\alpha$  coefficients are included as an additional row in matrix  $A$ .

**Everything happens as if  $\alpha x \geq \alpha_0$  was part of the original IP.** So, there is no change in the pricing structure and the cut is robust.

# Example: Generalized Assignment Problem (GAP)

( $\mathcal{RMP}$ ) Solution:

$$\begin{array}{lcl} \lambda_1 & = & \lambda_{[1010]^\top}^1 = 0.5 \\ \lambda_7 & = & \lambda_{[0100]^\top}^1 = 0.5 \\ \lambda_6 & = & \lambda_{[1001]^\top}^2 = 0.5 \\ \lambda_8 & = & \lambda_{[0111]^\top}^2 = 0.5 \end{array} \quad \Longrightarrow \quad \begin{array}{l} x_1^1 = x_1^2 = x_2^1 = x_2^2 = x_3^1 = x_3^2 = 0.5 \\ x_4^2 = 1 \\ z = 15 \end{array}$$

# Example: Generalized Assignment Problem (GAP)

( $\mathcal{RMP}$ ) Solution:

$$\begin{aligned} \lambda_1 &= \lambda_{[1010]^\top}^1 = 0.5 \\ \lambda_7 &= \lambda_{[0100]^\top}^1 = 0.5 \\ \lambda_6 &= \lambda_{[1001]^\top}^2 = 0.5 \\ \lambda_8 &= \lambda_{[0111]^\top}^2 = 0.5 \end{aligned} \quad \Rightarrow \quad \begin{aligned} x_1^1 &= x_1^2 = x_2^1 = x_2^2 = x_3^1 = x_3^2 = 0.5 \\ x_4^2 &= 1 \\ z &= 15 \end{aligned}$$

Separate robust cut:

$$x_1^2 + 2x_2^1 + 2x_3^1 + x_4^2 \leq 3$$

Translating to  $\lambda$  variables:

$$2\lambda_1 + \lambda_2 + 2\lambda_3 + \lambda_4 + 2\lambda_5 + 2\lambda_6 + 2\lambda_7 + \lambda_8 + 2\lambda_9 < 3$$

# Example: Generalized Assignment Problem (GAP)

## Restricted Master Problem with the robust cut

$$\begin{aligned} \text{Min } z &= \dots 10\lambda_1 + 13\lambda_2 + 11\lambda_3 + 7\lambda_4 + 11\lambda_5 + 3\lambda_6 + 3\lambda_7 + 14\lambda_8 + 2\lambda_9 \\ \dots \quad \lambda_1 + \lambda_2 & \quad \quad \quad + \lambda_5 + \lambda_6 & = 1 \quad (\pi_1 = 8) \\ \dots \quad \quad + \lambda_2 & \quad \quad \quad + \lambda_5 \quad \quad + \lambda_7 + \lambda_8 & = 1 \quad (\pi_2 = 8.6) \\ \dots \quad \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 & \quad \quad \quad \quad \quad \quad + \lambda_8 + \lambda_9 & = 1 \quad (\pi_3 = 7.6) \\ \dots \quad \quad \quad + \lambda_3 + \lambda_4 & \quad \quad \quad + \lambda_6 \quad \quad + \lambda_8 & = 1 \quad (\pi_4 = 9) \\ \dots \quad 2\lambda_1 + \lambda_2 + 2\lambda_3 + \lambda_4 + 2\lambda_5 + 2\lambda_6 + 2\lambda_7 + \lambda_8 + 2\lambda_9 & < 3 \quad (\pi_5 = -2.8) \\ \dots \quad \lambda_1 \quad \quad + \lambda_3 \quad \quad + \lambda_5 \quad \quad \lambda_7 \quad \quad + \lambda_9 & \leq 1 \quad (\nu^1 = 0) \\ \dots \quad \quad \lambda_2 \quad \quad + \lambda_4 \quad \quad + \lambda_6 \quad \quad + \lambda_8 & \leq 1 \quad (\nu^2 = -8.4) \\ & \lambda \geq 0 \end{aligned}$$

# Example: Generalized Assignment Problem (GAP)

## Restricted Master Problem with the robust cut

$$\begin{aligned} \text{Min } z = & \dots 10\lambda_1 + 13\lambda_2 + 11\lambda_3 + 7\lambda_4 + 11\lambda_5 + 3\lambda_6 + 3\lambda_7 + 14\lambda_8 + 2\lambda_9 \\ \dots \quad & \lambda_1 + \lambda_2 + \lambda_5 + \lambda_6 = 1 \quad (\pi_1 = 8) \\ \dots \quad & + \lambda_2 + \lambda_5 + \lambda_7 + \lambda_8 = 1 \quad (\pi_2 = 8.6) \\ \dots \quad & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_8 + \lambda_9 = 1 \quad (\pi_3 = 7.6) \\ \dots \quad & + \lambda_3 + \lambda_4 + \lambda_6 + \lambda_8 = 1 \quad (\pi_4 = 9) \\ \dots \quad & 2\lambda_1 + \lambda_2 + 2\lambda_3 + \lambda_4 + 2\lambda_5 + 2\lambda_6 + 2\lambda_7 + \lambda_8 + 2\lambda_9 < 3 \quad (\pi_5 = -2.8) \\ \dots \quad & \lambda_1 + \lambda_3 + \lambda_5 + \lambda_7 + \lambda_9 \leq 1 \quad (\nu^1 = 0) \\ \dots \quad & \lambda_2 + \lambda_4 + \lambda_6 + \lambda_8 \leq 1 \quad (\nu^2 = -8.4) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 16.4$$

Subproblem 1:

$$\min \sum_{j \in J} (c_j^1 - \pi_j) x_j^1 - 2\pi_5 x_2^1 - 2\pi_5 x_3^1 - \nu^1$$

$$\begin{aligned} \text{Min } \bar{c}^1 = & 0x_1^1 + 0x_2^1 + 0x_3^1 + 0x_4^1 \\ \text{S.t. } & 2x_1^1 + 3x_2^1 + 3x_3^1 + x_4^1 \leq 5 \\ & x \in \{0, 1\} \end{aligned}$$

$$\bar{c}^1 = 0$$

Subproblem 2:

$$\min \sum_{j \in J} (c_j^2 - \pi_j) x_j^2 - \pi_5 x_1^2 - \pi_5 x_4^2 - \nu^2$$

$$\begin{aligned} \text{Min } \bar{c}^2 = & -4.2x_1^2 - 1.6x_2^2 - 2.6x_3^2 - 4.2x_4^2 + 8.4 \\ \text{S.t. } & 5x_1^2 + x_2^2 + x_3^2 + 3x_4^2 \leq 8 \\ & x \in \{0, 1\} \end{aligned}$$

$$\bar{c}^2 = 0$$

$\bar{c} \geq 0$  for all subproblems  $\Rightarrow$  **Optimal (MP) Solution**



# Example: Generalized Assignment Problem (GAP)

New ( $\mathcal{MP}$ ) Solution with the robust cut:

$$\begin{array}{rclcl} \lambda_1 & = & \lambda_{[1010]^\top}^1 & = & 0.4 & & x_1^1 & = & 0.6 \\ \lambda_5 & = & \lambda_{[1100]^\top}^1 & = & 0.2 & & x_1^2 & = & 0.4 \\ \lambda_7 & = & \lambda_{[0100]^\top}^1 & = & 0.2 & \implies & x_2^1 & = & 0.4 \\ \lambda_6 & = & \lambda_{[1001]^\top}^2 & = & 0.4 & & x_2^2 & = & 0.6 \\ \lambda_8 & = & \lambda_{[0111]^\top}^2 & = & 0.6 & & x_3^1 & = & 0.4 \\ & & & & & & x_3^2 & = & 0.6 \\ & & & & & & x_4^2 & = & 1 \\ & & & & & & z & = & 16.4 \end{array}$$

Then, a robust branching over variable  $x_3^1$  (implemented by adding cuts  $x_3^1 \leq 0$  or  $x_3^1 \geq 1$ ) finds the optimal integer solution with  $z = 18$  and solves the instance in 3 nodes.

A fractional solution  $\lambda^*$  can be cut directly by a cutting plane

$$\sum_{q=1}^Q \alpha(p_q) \lambda_q \geq \alpha_0,$$

where coeffs  $\alpha(p_q)$  are given by an arbitrary function. The new dual variable will be denoted by  $\sigma$ . Now, the new pricing subproblem is:

$$\begin{aligned} \min \bar{c} &= (c - \pi A)x - \sigma \alpha(x) - \nu \\ &\text{subject to } x \in P. \end{aligned}$$

The non-robust cut introduces a non-linear term  $-\sigma \alpha(x)$  in the objective function of the pricing, breaking its original structure and forcing algorithmic adaptations that may make it much less efficient.

# Non-robust Chvátal-Gomory Cuts (CGCs)

Given valid inequalities  $Ax \leq b$  and a set of multipliers  $\rho \geq 0$ , the following GCC is valid:

$$\lfloor \rho A \rfloor x \leq \lfloor \rho b \rfloor.$$

Relaxing the set partitioning constraints in GAP reformulation, and applying the CGC procedure we get:

$$\sum_{k \in K} \sum_{q \in P^k} \left\lfloor \sum_{j=1}^J \rho_j p_{qj}^k \right\rfloor \lambda_q^k \geq \left\lfloor \sum_{j=1}^J \rho_j \right\rfloor. \quad (8)$$

The cut is non-robust because the floor operator makes the coefficients non-linear with respect to the points  $p$ .

# Example: Generalized Assignment Problem (GAP)

RMP Solution:

$$\begin{aligned}\lambda_1 &= \lambda_{[1010]^\tau}^1 = 0.5 \\ \lambda_7 &= \lambda_{[0100]^\tau}^1 = 0.5 \\ \lambda_6 &= \lambda_{[1001]^\tau}^2 = 0.5 \\ \lambda_8 &= \lambda_{[0111]^\tau}^2 = 0.5\end{aligned}$$

Using  $\rho = [2/3 \ 1/3 \ 1/3 \ 1/3]$ , we obtain the following violated cut:

$$\lambda_1 + \lambda_2 + \lambda_5 + \lambda_6 + \lambda_8 \leq 1$$

By also separating a second CGC with  $\rho = [1/3 \ 1/3 \ 1/3 \ 2/3]$ , the instance is solved at the root node.

However, the pricing is not a pure binary knapsack problem anymore, each added CGC makes it significantly harder to solve.

# The identical subproblems case

$$\min c^1 x^1 + c^2 x^2 + \dots + c^K x^K$$

subject to

$$A^1 x^1 + A^2 x^2 + \dots + A^K x^K = b$$

$$D^k x^k = d^k \quad k = 1, \dots, K$$

$$x^k \in Z_+^{n^k} \quad k = 1, \dots, K$$

Suppose that there are matrices  $A'$ ,  $D'$ ,  $c'$  and  $d'$  such that  $A^k = A'$ ,  $D^k = D'$ ,  $c^k = c'$  and  $d^k = d'$ , for  $k = 1, \dots, K$ . In that case, the  $K$  pricing subproblems would be identical. Assume that this set has  $Q$  solutions.

$$P' = \{D'x' = d', x' \geq 0 \text{ and integer}\}$$

# The identical subproblems case

- In that case, we can have a simpler Master LP Problem:

$$(\mathcal{IMP}) \min z = \sum_{j=1}^Q (c' p_j) \lambda_j$$

subject to

$$\sum_{j=1}^Q (A' p_j) \lambda_j = b \quad (\pi)$$

$$\sum_{j=1}^Q \lambda_j = K \quad (\nu)$$

$$\lambda \geq 0$$

- There is a single pricing subproblem:

$$\begin{aligned} (SP) \quad & \min \bar{c} = (c' - \pi A')x' - \nu \\ & \text{s.t. } D'x' = d' \\ & \quad x' \geq 0 \text{ and integer} \end{aligned}$$

- The restricted master LP is optimal when  $\bar{c} \geq 0$

## 1 Instance

- Stocks of length  $W$
- Set  $J$  of items
  - Each item  $j \in J$  has length  $w_j$  and a demand of  $b_j$  copies

## 2 Problem

Obtain the demanded number of copies of each item by cutting the minimum possible number of stocks

The particular case where all demands  $b_j$  are unitary is known as the Bin Packing problem



Assumes the existence of a heuristic upper bound  $M$ . Number the potentially used stocks from 1 to  $M$ :

## Variables

- $x_{ij}$ : determines how many items  $j$  are cut from stock  $i$
- $y_i$ : indicates whether stock  $i$  is used or not

$$\min \quad z = \sum_{i=1}^M y_i \quad (9a)$$

$$\text{S.t.} \quad \sum_{i=1}^M x_{ij} = b_j, \quad j \in J; \quad (9b)$$

$$\sum_{j \in J} w_j x_{ij} \leq W y_i, \quad i = 1, \dots, M; \quad (9c)$$

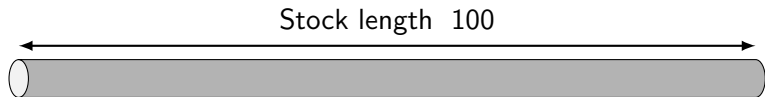
$$x_{ij} \in \mathbb{Z}_+^{M \times |J|}, \quad i = 1, \dots, M, j \in J; \quad (9d)$$





$$y_i \in \{0, 1\}, \quad i = 1, \dots, M. \quad (9e)$$

## Issues

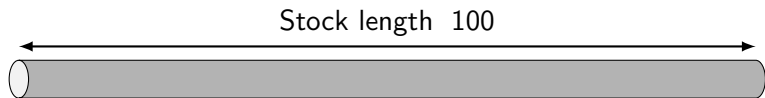
- A not-really compact formulation. The formulation size is pseudo-polynomial
- Even when  $M$  is not so large, branch-and-bound algorithms perform poorly on it
  - the linear relaxation lower bound is equal to the the trivial lower bound  $\frac{\sum_{j \in J} b_j w_j}{W}$
  - suffers from symmetry: the same fractional/integer solution can be represented in many different ways by only permutating the stock indices





# Example: Cutting Stock Problem



	Length	Demand
	40	4
	35	5
	31	5
	13	8

# Example: Cutting Stock Problem



	Length	Demand
	40	4
	35	5
	31	5
	13	8

Trivial lower bound =  $(4 \times 40 + 5 \times 35 + 5 \times 31 + 8 \times 13)/100 = 5.94$

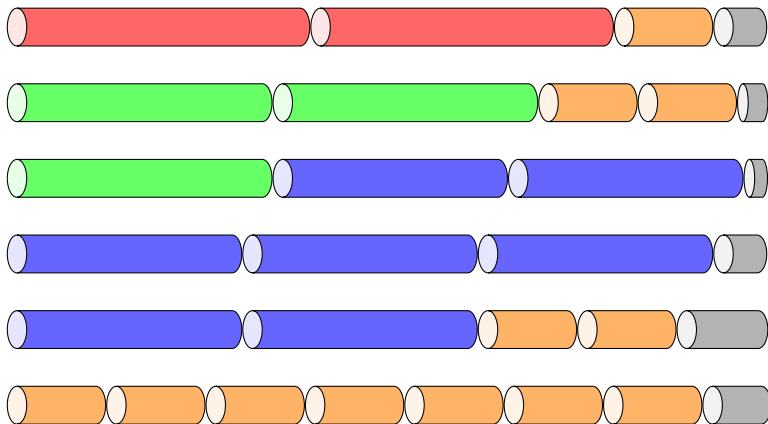
Rounding up  $\implies$  Lower Bound: 6 stocks

Based on the concept of *cutting patterns*

- A cutting pattern is a possible way of cutting a stock; it is defined by the number of copies of each item obtained from that stock

# Example: Cutting Stock Problem

## Some Possible Cutting Patterns



# Gilmore and Gomory (1961, 1963) Formulation

$$\min \quad z = \sum_{q=1}^Q \lambda_q \quad (10a)$$

$$\text{S.t.} \quad \sum_{q=1}^Q p_{qj} \lambda_q = b_j \quad j \in J \quad (\pi) \quad (10b)$$

$$\lambda_q \in \mathbb{Z} \quad q = 1, \dots, Q \quad (10c)$$

- Exponential number of  $\lambda$  variables, one for each cutting pattern (numbered from 1 to  $Q$ )
- $p_{qj}$  indicates how many copies of item  $j$  are obtained in the  $q$ -th cutting pattern
- Its linear relaxation can be efficiently solved by column generation

## Pricing subproblem

- At each iteration, the following Integer Knapsack Problem is solved:

$$\min \bar{c} = 1 - \sum_{j \in J} \pi_j x_j$$

$$\text{S.t.} \quad \sum_{j \in J} w_j x_j \leq W,$$

$$x_j \in \mathbb{Z}_+^{|J|} \quad \forall j \in J.$$

- Each subproblem solution is a cutting pattern
- The Integer Knapsack Problem is (weakly) NP-hard, but very well solved in practice (see Pferschy et al. (2004))



- Gilmore-Gomory Formulation can be obtained by a DW decomposition of the symmetric formulation.

GG Formulation is stronger (its linear relaxation yields better lower bounds) because knapsack constraints (9c) are convexified

## Remarkably strong linear relaxation bounds

- It is very hard to find an instance where the  $(\mathcal{MP})$  solution value rounded up is not equal to the value of an optimal integer solution!
- It is conjectured that the  $(\mathcal{MP})$  solution value rounded up is at most one unit away from the value of an optimal integer solution!!

# Example: Cutting Stock Problem

Iteration 1:

$$\begin{aligned} \min z = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 \\ 2\lambda_1 &= 4 \quad (\pi_1 = 0.50) \\ 2\lambda_2 &= 5 \quad (\pi_2 = 0.50) \\ 3\lambda_3 &= 5 \quad (\pi_3 = 0.33) \\ 7\lambda_4 &= 8 \quad (\pi_4 = 0.14) \\ \lambda &\geq 0 \end{aligned}$$

$$z = 7.31; \lambda_1 = 2, \lambda_2 = 2.5, \lambda_3 = 1.67, \lambda_4 = 1.14$$

# Example: Cutting Stock Problem

Iteration 1:

$$\begin{aligned} \min z = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 \\ 2\lambda_1 &= 4 \quad (\pi_1 = 0.50) \\ 2\lambda_2 &= 5 \quad (\pi_2 = 0.50) \\ 3\lambda_3 &= 5 \quad (\pi_3 = 0.33) \\ 7\lambda_4 &= 8 \quad (\pi_4 = 0.14) \\ \lambda &\geq 0 \end{aligned}$$

$$z = 7.31; \lambda_1 = 2, \lambda_2 = 2.5, \lambda_3 = 1.67, \lambda_4 = 1.14$$

Subproblem:  $\min 1 - \sum_{j=1}^n \pi_j x_j$

$$\begin{aligned} \min \bar{c} &= -0.5x_1 - 0.5x_2 - 0.33x_3 - 0.14x_4 + 1 \\ \text{S.t.} \quad &40x_1 - 35x_2 + 31x_3 + 13x_4 \leq 100 \\ &x \in Z_+^4 \end{aligned}$$

$$S = (0, 2, 0, 2) \text{ and } \bar{c} = -0.29$$

# Example: Cutting Stock Problem

Iteration 2:

$$\begin{aligned} \min z = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 \\ 2\lambda_1 &= 4 \quad (\pi_1 = 0.50) \\ 2\lambda_2 + 2\lambda_5 &= 5 \quad (\pi_2 = 0.36) \\ 3\lambda_3 &= 5 \quad (\pi_3 = 0.33) \\ 7\lambda_4 + 2\lambda_5 &= 8 \quad (\pi_4 = 0.14) \\ \lambda &\geq 0 \end{aligned}$$

$$z = 6.59; \lambda_1 = 2, \lambda_3 = 1.67, \lambda_4 = 0.43, \lambda_5 = 2.5$$

Subproblem:  $\min 1 - \sum_{j=1}^n \pi_j x_j$

$$\begin{aligned} \min \bar{c} &= -0.5x_1 - 0.36x_2 - 0.33x_3 - 0.14x_4 + 1 \\ \text{S.t.} \quad 40x_1 - 35x_2 + 31x_3 + 13x_4 &\leq 100 \\ x &\in Z_+^4 \end{aligned}$$

$$S = (2, 0, 0, 1) \text{ and } \bar{c} = -0.14$$

# Example: Cutting Stock Problem

Iteration 3:

$$\begin{aligned} \min z = & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 \\ & 2\lambda_1 + 2\lambda_6 = 4 \quad (\pi_1 = 0.43) \\ & \quad 2\lambda_2 + 2\lambda_5 = 5 \quad (\pi_2 = 0.36) \\ & \quad \quad 3\lambda_3 = 5 \quad (\pi_3 = 0.33) \\ & \quad \quad \quad 7\lambda_4 + 2\lambda_5 + \lambda_6 = 8 \quad (\pi_4 = 0.14) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 6.31; \lambda_3 = 1.67, \lambda_4 = 0.14, \lambda_5 = 2.5, \lambda_6 = 2$$

Subproblem:  $\min 1 - \sum_{j=1}^n \pi_j x_j$

$$\begin{aligned} \min \bar{c} = & -0.43x_1 - 0.36x_2 - 0.33x_3 - 0.14x_4 + 1 \\ \text{S.t.} \quad & 40x_1 - 35x_2 + 31x_3 + 13x_4 \leq 100 \\ & x \in Z_+^4 \end{aligned}$$

$$S = (0, 1, 0, 5) \text{ and } \bar{c} = -0.07$$

# Example: Cutting Stock Problem

Iteration 4:

$$\begin{aligned} \min z = & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 + \lambda_7 \\ & 2\lambda_1 \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad + 2\lambda_6 \qquad \qquad \qquad = 4 \quad (\pi_1 = 0.44) \\ & \qquad \qquad 2\lambda_2 \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad + 2\lambda_5 \qquad \qquad + \lambda_7 = 5 \quad (\pi_2 = 0.38) \\ & \qquad \qquad \qquad 3\lambda_3 \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad = 5 \quad (\pi_3 = 0.33) \\ & \qquad \qquad \qquad \qquad \qquad 7\lambda_4 + 2\lambda_5 + \lambda_6 + 5\lambda_7 = 8 \quad (\pi_4 = 0.12) \\ & \lambda \geq 0 \end{aligned}$$

$$z = 6.29; \lambda_3 = 1.67, \lambda_5 = 2.38, \lambda_6 = 2, \lambda_7 = 0.25$$

Subproblem:  $\min 1 - \sum_{j=1}^n \pi_j x_j$

$$\begin{aligned} \min \bar{c} = & -0.44x_1 - 0.38x_2 - 0.33x_3 - 0.12x_4 + 1 \\ \text{S.t.} \quad & 40x_1 - 35x_2 + 31x_3 + 13x_4 \leq 100 \\ & x \in Z_+^4 \end{aligned}$$

$$S = (0, 1, 2, 0) \text{ and } \bar{c} = -0.04$$

# Example: Cutting Stock Problem

Iteration 5:

$$\begin{array}{rll} \min z = & \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 + \lambda_7 + \lambda_8 & \\ & 2\lambda_1 & + 2\lambda_6 = 4 \quad (\pi_1 = 0.44) \\ & 2\lambda_2 & + 2\lambda_5 + \lambda_7 + \lambda_8 = 5 \quad (\pi_2 = 0.38) \\ & 3\lambda_3 & + 2\lambda_8 = 5 \quad (\pi_3 = 0.31) \\ & 7\lambda_4 + 2\lambda_5 + \lambda_6 + 5\lambda_7 & = 8 \quad (\pi_4 = 0.12) \\ & \lambda \geq 0 & \end{array}$$

$$z = 6.19; \lambda_5 = 0.81, \lambda_6 = 2, \lambda_7 = 0.88, \lambda_8 = 2.5$$

Subproblem:  $\min 1 - \sum_{j=1}^n \pi_j x_j$

$$\begin{array}{ll} \min \bar{c} = & -0.44x_1 - 0.38x_2 - 0.31x_3 - 0.12x_4 + 1 \\ \text{s.t.} & 40x_1 - 35x_2 + 31x_3 + 13x_4 \leq 100 \\ & x \in Z_+^4 \end{array}$$

$S = (2, 0, 0, 1)$  and  $\bar{c} = 0$

New lower bound: 7 stocks. But no integer solution. **What to do?**



# Getting an (heuristic) integer solution

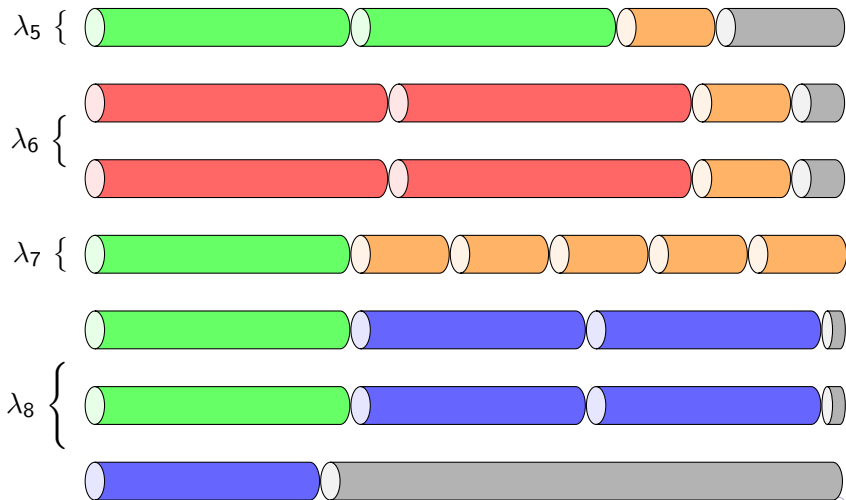
Change constraints to  $\geq$  and solve the restricted master as an IP:

$$\begin{array}{rcccccccc} \min z = & \lambda_1 & + & \lambda_2 & + & \lambda_3 & + & \lambda_4 & + & \lambda_5 & + & \lambda_6 & + & \lambda_7 & + & \lambda_8 \\ & 2\lambda_1 & & & & & & & & & + & 2\lambda_6 & & & & \geq 4 \\ & & & 2\lambda_2 & & & & & + & 2\lambda_5 & & & + & \lambda_7 & + & \lambda_8 \geq 5 \\ & & & & & 3\lambda_3 & & & & & & & & + & 2\lambda_8 \geq 5 \\ & & & & & & & 7\lambda_4 & + & 2\lambda_5 & + & \lambda_6 & + & 5\lambda_7 & & \geq 8 \\ & & & & & & & & & & & & & & & \lambda \in \mathbb{Z}_+^8 \end{array}$$

$$\lambda_5 = 1, \lambda_6 = 2, \lambda_7 = 1, \lambda_8 = 3, z = 7$$

# Example: Cutting Stock Problem

The obtained optimal solution with 7 stocks (after trimming the surplus copies)



# Getting an optimal integer solution for CSP

- Somehow rounding the ( $\mathcal{RMP}$ ) solution, the idea originally proposed by Gilmore and Gomory, works reasonably well in practice, but it is not guaranteed to find an optimal integer
- Of course, a solution obtained by any heuristic that matches the GG lower bound is proved to be optimal
- Yet, only much later (mid-1990s) Branch-and-Price algorithms for the CSP and the BPP were created
  - The original symmetric variables are useless for branching or cutting. More complex non-robust schemes should be devised

# When DW decomposition for IP should be tried?

Some guidelines:

When DW decomp improves linear relaxation significantly, but the subproblems are still tractable

- Solving a ( $\mathcal{MP}$ ) by CG is time-consuming. This is usually only worthy if the resulting bounds are **a lot better**
  - This means that the *subproblems should not be easy polynomial problems*. The big gains are obtained exactly by convexifying constraints that define NP-hard problems
  - “Tractable” NP-hard problems often includes those solvable in pseudo-polynomial time or those where some exact algorithm still performs well on instances of reasonable size
- A delicate balance

# When DW decomposition for IP should be tried?

When DW decomp leads to many small subproblems, better if many of them are identical

- Gains in pricing time and also in CG convergence

When DW decomp removes the symmetry of a bad formulation

- CSP and BPP are good examples

# Robust vs Non-Robust BCPs

Robust BCPAs can improve a lot over BP algorithms:

- Robust BCP is easier to implement, if you already know families of cuts for the problem and have their separation algorithms
  - Yet, sometimes even families of facets are useless, because they are already implied by the column generation!
- No worries about destroying the pricing structure

Non-robust BCPAs can possibly do better:

- In some problems all robust cuts are useless (due to a symmetric original formulation)
- **Each master variable carries much more information than an original variable.** It is much easier to find strong non-robust cuts over them!
- Yet, non-robust cuts are indeed “non-robust”: a few dozen bad cuts can make your pricing 1000x slower!

The most advanced state-of-the-art BCPAs are exactly those where the **non-robust cuts and the pricing algorithm are jointly and symbiotically designed**, in such a way that the pricing can handle a large number of very tailored non-robust cuts without becoming too inefficient.

The most advanced state-of-the-art BCPAs are exactly those where the **non-robust cuts and the pricing algorithm are jointly and symbiotically designed**, in such a way that the pricing can handle a large number of very tailored non-robust cuts without becoming too inefficient.

Spoiler: BCPs for Vehicle Routing Problems



- Dantzig, G. B. and Wolfe, P. (1960). Decomposition principle for linear programs. *Operations research*, 8(1):101–111.
- Ford Jr, L. R. and Fulkerson, D. R. (1958). A suggested computation for maximal multi-commodity network flows. *Management Science*, 5(1):97–101.
- Gilmore, P. C. and Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859.
- Gilmore, P. C. and Gomory, R. E. (1963). A linear programming approach to the cutting stock problem—part ii. *Operations research*, 11(6):863–888.
- Pferschy, U., Kellerer, H., and Pisinger, D. (2004). *Knapsack Problems*. Springer.