Operations Research Exercises

Mathematical Modeling

Management Engineering

University of Palermo

simona.mancini@unipa.it
alessio.sclafani@unipa.it

Lesson Objectives

What we will learn:

- Translate a real-world problem into a mathematical model
- Identify decision variables, constraints, and objective functions
- Implement and solve models with Gurobi (Python)
- Interpret and validate results

Slide available at: `https://github.com/alessioscl/OR25-26`

Example 1: Logistics Network Optimization

Context:
A company has 3 production plants and must supply 4 distribution centers.
Production plants: $I = \{1, 2, 3\}$, Distribution center: $J = \{A, B, C, D\}$

Production capacity (units/month): $b_i \quad \forall\, i \in I$

- Plant 1: 200 units
- Plant 2: 300 units
- Plant 3: 250 units

Distribution center demand (units/month): $d_j \quad \forall\, j \in J$

- Center A: 150 units
- Center B: 200 units
- Center C: 180 units
- Center D: 170 units

Objective: Minimize total transportation costs.

Example 1: Transportation Costs

Unit transportation costs (euros/unit): $c_{ij} \quad \forall\, i \in I, j \in J$

	Center A	**Center B**	**Center C**	**Center D**
Plant 1	8	6	10	9
Plant 2	9	12	7	8
Plant 3	14	9	11	5

Example: transporting 1 unit from Plant 1 to Center A costs 8 euros.

Example 1: Modeling - Step 1

STEP 1: Identify the decision variables

Example 1: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

Example 1: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *How much product to transport from each plant to each center*

Example 1: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *How much product to transport from each plant to each center*

Decision variables:

$$x_{ij} = \text{units transported from plant } i \text{ to center } j$$

where $i \in I(= \{1, 2, 3\})$ and $j \in J(= \{A, B, C, D\})$

Example 1: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *How much product to transport from each plant to each center*

Decision variables:

$$x_{ij} = \text{units transported from plant } i \text{ to center } j$$

where $i \in I(= \{1, 2, 3\})$ and $j \in J(= \{A, B, C, D\})$

Example: $x_{1A} =$ units from Plant 1 to Center A

Example 1: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *How much product to transport from each plant to each center*

Decision variables:

$$x_{ij} = \text{units transported from plant } i \text{ to center } j$$

where $i \in I(= \{1, 2, 3\})$ and $j \in J(= \{A, B, C, D\})$

Example: $x_{1A} =$ units from Plant 1 to Center A

Domain: $x_{ij} \geq 0$ for each pair (i, j)

Example 1: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

Example 1: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

$\Rightarrow$ *Minimize the total transportation cost*

Example 1: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

$\Rightarrow$ *Minimize the total transportation cost*

Objective function:

$$\begin{aligned} \min \quad z = & 8x_{1A} + 6x_{1B} + 10x_{1C} + 9x_{1D} + \\ & 9x_{2A} + 12x_{2B} + 7x_{2C} + 8x_{2D} + \\ & 14x_{3A} + 9x_{3B} + 11x_{3C} + 5x_{3D} \end{aligned}$$

Example 1: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

$\Rightarrow$ *Minimize the total transportation cost*

Objective function:

$$\begin{aligned}\min \quad z = & 8x_{1A} + 6x_{1B} + 10x_{1C} + 9x_{1D} + \\ & 9x_{2A} + 12x_{2B} + 7x_{2C} + 8x_{2D} + \\ & 14x_{3A} + 9x_{3B} + 11x_{3C} + 5x_{3D}\end{aligned}$$

Or in compact form:

$$\min \quad z = \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

Example 1: Modeling - Step 3a

STEP 3: Formulate constraints

A. Plant capacity constraints

Each plant i cannot ship more than its capacity b_i.

Example 1: Modeling - Step 3a

STEP 3: Formulate constraints

A. Plant capacity constraints

Each plant i cannot ship more than its capacity b_i.

Plant 1 (capacity 200):

$$x_{1A} + x_{1B} + x_{1C} + x_{1D} \leq 200$$

Example 1: Modeling - Step 3a

STEP 3: Formulate constraints

A. Plant capacity constraints

Each plant i cannot ship more than its capacity b_i.

Plant 1 (capacity 200):

$$x_{1A} + x_{1B} + x_{1C} + x_{1D} \leq 200$$

Plant 2 (capacity 300):

$$x_{2A} + x_{2B} + x_{2C} + x_{2D} \leq 300$$

Example 1: Modeling - Step 3a

STEP 3: Formulate constraints

A. Plant capacity constraints

Each plant i cannot ship more than its capacity b_i.

Plant 1 (capacity 200):

$$x_{1A} + x_{1B} + x_{1C} + x_{1D} \leq 200$$

Plant 2 (capacity 300):

$$x_{2A} + x_{2B} + x_{2C} + x_{2D} \leq 300$$

Plant 3 (capacity 250):

$$x_{3A} + x_{3B} + x_{3C} + x_{3D} \leq 250$$

Example 1: Modeling - Step 3a

STEP 3: Formulate constraints

A. Plant capacity constraints

Each plant i cannot ship more than its capacity b_i.

Plant 1 (capacity 200):

$$x_{1A} + x_{1B} + x_{1C} + x_{1D} \leq 200$$

Plant 2 (capacity 300):

$$x_{2A} + x_{2B} + x_{2C} + x_{2D} \leq 300$$

Plant 3 (capacity 250):

$$x_{3A} + x_{3B} + x_{3C} + x_{3D} \leq 250$$

Or in compact form:

$$\sum_{j \in J} x_{ij} \leq b_i \quad \forall\, i \in I$$

Example 1: Modeling - Step 3b

B. Demand constraints of distribution centers

Each center must receive exactly the requested demand.

Example 1: Modeling - Step 3b

B. Demand constraints of distribution centers
Each center must receive exactly the requested demand.

Center A (demand 150):

$$x_{1A} + x_{2A} + x_{3A} = 150$$

Example 1: Modeling - Step 3b

B. Demand constraints of distribution centers
Each center must receive exactly the requested demand.

Center A (demand 150):

$$x_{1A} + x_{2A} + x_{3A} = 150$$

Center B (demand 200):

$$x_{1B} + x_{2B} + x_{3B} = 200$$

Example 1: Modeling - Step 3b

B. Demand constraints of distribution centers
Each center must receive exactly the requested demand.

Center A (demand 150):

$$x_{1A} + x_{2A} + x_{3A} = 150$$

Center B (demand 200):

$$x_{1B} + x_{2B} + x_{3B} = 200$$

$$\dots$$

Or in compact form:

$$\sum_{i \in I} x_{ij} \leq d_j \quad \forall\, j \in J$$

Example 1: Complete Model

Mathematical formulation:

$$\min \quad z = \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij}$$

$$\text{s.t.} \quad \sum_{j \in J} x_{ij} \leq b_i \quad \forall\, i \in I \qquad \text{(capacity)}$$

$$\sum_{i \in I} x_{ij} \leq d_j \quad \forall\, j \in J \qquad \text{(demand)}$$

$$x_{ij} \geq 0 \quad \forall i, j$$

Example 1: Solution

Implementation in python

https://github.com/alessioscl/OR25-26/blob/main/Esempio_1_Rete_Logistica.ipynb

Example 1 Extended: Limitations of Transportation Model

What the basic model doesn't consider:

In the transportation problem, we assume:

- Direct shipments from each plant to each center
- No routing decisions (no intermediate stops)
- Unlimited vehicles available
- No vehicle capacity constraints
- No travel distances/times considered

Example 1 Extended: Limitations of Transportation Model

What the basic model doesn't consider:

In the transportation problem, we assume:

- Direct shipments from each plant to each center
- No routing decisions (no intermediate stops)
- Unlimited vehicles available
- No vehicle capacity constraints
- No travel distances/times considered

Real-world scenario:

A company has:

- A **central depot** (Plant 1)
- Multiple **customers** to deliver to (Centers A, B, C, D)
- A **fleet of vehicles** with limited capacity
- Need to design **delivery routes** (sequences of visits)
- Each customer visited exactly once
- All vehicles return to depot

Example 2: Project Selection with Limited Budget

Context:

A company must select which innovation projects to finance.

Available data:

Project	NPV (k)	Investment (k)	Resources (FTE)	Duration (months)
P1	250	120	3	12
P2	180	80	2	8
P3	320	150	4	15
P4	200	100	2	10
P5	150	70	2	6
P6	280	130	3	14

Available constraints:

- Total available budget: 300 k
- Available human resources: 7 FTE (Full-Time Equivalent)
- Projects are indivisible (they are either done completely or not done at all)

Objective: Select the project portfolio that maximizes the total NPV.

Example 2: Problem Characteristics

Knapsack Problem:

Analogy: You have a backpack with limited capacity and must choose which items to carry to maximize the total value.

In our case:

- **Backpack** = Budget and available resources
- **Items** = Projects to be evaluated
- **Weight** = Costs and necessary resources
- **Value** = NPV (Net Present Value)

* NPV allows the value created by an investment project to be measured in monetary terms and also identifies an objective criterion for determining whether or not it is worthwhile undertaking the project.

Complexity:

- We cannot make "partial" projects
- Two capacity constraints (budget AND resources)
- Combinatorial optimization problem

Example 2: Modeling - Step 1

STEP 1: Identify the decision variables

Example 2: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

Example 2: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *Which projects to select (yes/no for each project)*

Example 2: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *Which projects to select (yes/no for each project)*

Binary decision variables:

$$x_i \in \{0,1\} = \begin{cases} 1 & \text{if we select project } i \\ 0 & \text{otherwise} \end{cases}$$

where $i \in \{1,2,3,4,5,6\}$ (set of projects)

Example 2: Modeling - Step 1

STEP 1: Identify the decision variables

What do we need to decide?

$\Rightarrow$ *Which projects to select (yes/no for each project)*

Binary decision variables:

$$x_i \in \{0,1\} = \begin{cases} 1 & \text{if we select project } i \\ 0 & \text{otherwise} \end{cases}$$

where $i \in \{1,2,3,4,5,6\}$ (set of projects)

Example:

- $x_1 = 1$ means that we implement Project 1
- $x_3 = 0$ means that we do NOT implement Project 3

Example 2: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

Example 2: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

$\Rightarrow$ *Maximize the total NPV of the selected projects*

Example 2: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

$\Rightarrow$ *Maximize the total NPV of the selected projects*

Example 2: Modeling - Step 2

STEP 2: Define the objective function

What do we want to optimize?

$\Rightarrow$ *Maximize the total NPV of the selected projects*

Objective function:

$$\max z = 250x_1 + 180x_2 + 320x_3 + 200x_4 + 150x_5 + 280x_6$$

Or in compact form:

$$\max z = \sum_{i=1}^{6} \mathsf{NPV}_i \cdot x_i$$

Example 2: Modeling - Step 3

STEP 3: Formulate the constraints

1. **Budget constraint** (max 300 k):

Example 2: Modeling - Step 3

STEP 3: Formulate the constraints

1. **Budget constraint** (max 300 k):

$$120x_1 + 80x_2 + 150x_3 + 100x_4 + 70x_5 + 130x_6 \leq 300$$

Example 2: Modeling - Step 3

STEP 3: Formulate the constraints

1. **Budget constraint** (max 300 k):

$$120x_1 + 80x_2 + 150x_3 + 100x_4 + 70x_5 + 130x_6 \leq 300$$

Or in compact form:

$$\sum_{i=1}^{6} \text{Cost}_i \cdot x_i \leq 300$$

Example 2: Modeling - Step 3

STEP 3: Formulate the constraints

1. **Budget constraint** (max 300 k):

$$120x_1 + 80x_2 + 150x_3 + 100x_4 + 70x_5 + 130x_6 \leq 300$$

Or in compact form:

$$\sum_{i=1}^{6} \text{Cost}_i \cdot x_i \leq 300$$

2. **Human resources constraint** (max 7 FTE):

Example 2: Modeling - Step 3

STEP 3: Formulate the constraints

1. **Budget constraint** (max 300 k):

$$120x_1 + 80x_2 + 150x_3 + 100x_4 + 70x_5 + 130x_6 \leq 300$$

Or in compact form:

$$\sum_{i=1}^{6} \text{Cost}_i \cdot x_i \leq 300$$

2. **Human resources constraint** (max 7 FTE):

$$3x_1 + 2x_2 + 4x_3 + 2x_4 + 2x_5 + 3x_6 \leq 7$$

Example 2: Modeling - Step 3

STEP 3: Formulate the constraints

1. **Budget constraint** (max 300 k):

$$120x_1 + 80x_2 + 150x_3 + 100x_4 + 70x_5 + 130x_6 \leq 300$$

Or in compact form:

$$\sum_{i=1}^{6} \text{Cost}_i \cdot x_i \leq 300$$

2. **Human resources constraint** (max 7 FTE):

$$3x_1 + 2x_2 + 4x_3 + 2x_4 + 2x_5 + 3x_6 \leq 7$$

Or in compact form:

$$\sum_{i=1}^{6} \text{FTE}_i \cdot x_i \leq 7$$

Example 2: Modeling - Step 3 (continued)

3. Binary constraints:

$$x_i \in \{0, 1\} \quad \forall i \in \{1, 2, 3, 4, 5, 6\}$$

Example 2: Modeling - Step 3 (continued)

3. Binary constraints:

$$x_i \in \{0, 1\} \quad \forall i \in \{1, 2, 3, 4, 5, 6\}$$

Interpretation of constraints:

- **Budget**: The sum of the costs of the selected projects cannot exceed 300 k
- **Resources**: The sum of the necessary resources cannot exceed 7 FTE
- **Binary**: Each project is "all or nothing" (we cannot do half a project)

Example 2: Complete Model

Complete Mathematical Formulation:

Given the set of projects $P = \{1, 2, 3, 4, 5, 6\}$:

$$\max \quad z = \sum_{i \in P} \text{NPV}_i \cdot x_i$$

$$\text{s.t.} \quad \sum_{i \in P} \text{Cost}_i \cdot x_i \leq 300 \qquad \text{(budget constraint)}$$

$$\sum_{i \in P} \text{FTE}_i \cdot x_i \leq 7 \qquad \text{(resource constraint)}$$

$$x_i \in \{0, 1\} \qquad \forall i \in P$$

Example 2: Complete Model

Complete Mathematical Formulation:
Given the set of projects $P = \{1, 2, 3, 4, 5, 6\}$:

$$\begin{aligned}
\max \quad & z = \sum_{i \in P} \text{NPV}_i \cdot x_i \\
\text{s.t.} \quad & \sum_{i \in P} \text{Cost}_i \cdot x_i \leq 300 && \text{(budget constraint)} \\
& \sum_{i \in P} \text{FTE}_i \cdot x_i \leq 7 && \text{(resource constraint)} \\
& x_i \in \{0, 1\} && \forall i \in P
\end{aligned}$$

Model type: Binary Integer Programming (BIP)

- Only binary variables (0-1 problem)
- NP-hard problem (difficult from a computational point of view)
- Special case of the multidimensional knapsack problem

Example 2: Preliminary Analysis

Considerations before solving:

How many possible solutions?

Example 2: Preliminary Analysis

Considerations before solving:

How many possible solutions?

- 6 projects $\rightarrow 2^6 = 64$ possible combinations
- Some violate the constraints (not admissible)

Example 2: Preliminary Analysis

Considerations before solving:

How many possible solutions?

- 6 projects $\rightarrow 2^6 = 64$ possible combinations
- Some violate the constraints (not admissible)

Heuristics solutions to compare:

- **Greedy for NPV**: Take projects with the highest NPV
- **Greedy for efficiency**: Maximum NPV/Cost ratio
- **Optimal solution**: Found by the solver

Example 2: Preliminary Analysis

Considerations before solving:

How many possible solutions?

- 6 projects $\rightarrow 2^6 = 64$ possible combinations
- Some violate the constraints (not admissible)

Heuristics solutions to compare:

- **Greedy for NPV**: Take projects with the highest NPV
- **Greedy for efficiency**: Maximum NPV/Cost ratio
- **Optimal solution**: Found by the solver

Question: Will the project with the highest NPV (P3: €320k) definitely be part of the optimal solution?

Example 2: Preliminary Analysis

Considerations before solving:

How many possible solutions?

- 6 projects $\rightarrow 2^6 = 64$ possible combinations
- Some violate the constraints (not admissible)

Heuristics solutions to compare:

- **Greedy for NPV**: Take projects with the highest NPV
- **Greedy for efficiency**: Maximum NPV/Cost ratio
- **Optimal solution**: Found by the solver

Question: Will the project with the highest NPV (P3: €320k) definitely be part of the optimal solution?
Not necessarily! It could "consume" too many resources...

Example 2 Extended: Adding Compatibility Constraints

Real-world complexity:

In practice, project selection involves additional **logical relationships**:

- **Dependencies**: Some projects require others to be done first
- **Mutual exclusivity**: Some projects cannot be done together
- **Synergies**: Some projects work better together
- **Cardinality**: Constraints on number of projects from certain categories

Example 2 Extended: Adding Compatibility Constraints

Real-world complexity:
In practice, project selection involves additional **logical relationships**:

- **Dependencies**: Some projects require others to be done first
- **Mutual exclusivity**: Some projects cannot be done together
- **Synergies**: Some projects work better together
- **Cardinality**: Constraints on number of projects from certain categories

Example scenario:

- P1 is an infrastructure project (platform development)
- P3 requires P1 infrastructure to work (dependency)
- P2 and P4 target the same market (mutual exclusivity)
- P5 and P6 are R&D projects (at least one must be selected)
- Projects are categorized: Core (P1,P3) vs Market (P2,P4) vs R&D (P5,P6)

Challenge: How to model these logical constraints mathematically?

Compatibility Constraint Types

Type 1: Prerequisite/Dependency

Project i can only be done if project j is also done:

$$x_i \leq x_j$$

Interpretation:

- If $x_i = 1$ (we do project i), then x_j must be 1
- If $x_j = 0$ (we don't do j), then x_i must be 0
- If $x_j = 1$, project i is optional

Compatibility Constraint Types

Type 1: Prerequisite/Dependency

Project i can only be done if project j is also done:

$$x_i \leq x_j$$

Interpretation:

- If $x_i = 1$ (we do project i), then x_j must be 1
- If $x_j = 0$ (we don't do j), then x_i must be 0
- If $x_j = 1$, project i is optional

Example: P3 requires P1 infrastructure

$$x_3 \leq x_1$$

Logic table:

x_1	x_3	Feasible?
0	0	ok
0	1	no (violates $x_3 \leq x_1$)
1	0	ok
1	1	ok

Compatibility Constraint Types (continued)

Type 2: Mutual Exclusivity

At most one of two projects can be selected:

$$x_i + x_j \leq 1$$

Interpretation:

- Both can be 0 (neither selected)
- One can be 1 and the other 0
- Both cannot be 1 simultaneously

Compatibility Constraint Types (continued)

Type 2: Mutual Exclusivity

At most one of two projects can be selected:

$$x_i + x_j \leq 1$$

Interpretation:

- Both can be 0 (neither selected)
- One can be 1 and the other 0
- Both cannot be 1 simultaneously

Example: P2 and P4 are mutually exclusive (same market)

$$x_2 + x_4 \leq 1$$

Logic table:

x_2	x_4	Feasible?
0	0	ok
0	1	ok
1	0	ok
1	1	no (violates $x_2 + x_4 \leq 1$)

Compatibility Constraint Types (continued)

Type 3: Conditional Implication

If project i is selected, then project j must also be selected:

$$x_i \leq x_j \quad \text{or equivalently} \quad x_i - x_j \leq 0$$

This is the same as Type 1 (prerequisite), but viewed from different perspective.

Compatibility Constraint Types (continued)

Type 3: Conditional Implication

If project i is selected, then project j must also be selected:

$$x_i \leq x_j \quad \text{or equivalently} \quad x_i - x_j \leq 0$$

This is the same as Type 1 (prerequisite), but viewed from different perspective.

Type 4: Bi-directional Implication

Projects i and j must be selected together (or not at all):

$$x_i = x_j$$

Can be modeled as two constraints:

$$x_i \leq x_j$$
$$x_j \leq x_i$$

Example: P5 and P6 are complementary R&D projects (package deal)

Compatibility Constraint Types (continued)

Type 5: At Least One from Set

At least k projects from a set S must be selected:

$$\sum_{i \in S} x_i \geq k$$

Example: At least 1 R&D project must be selected

$$x_5 + x_6 \geq 1$$

Compatibility Constraint Types (continued)

Type 5: At Least One from Set

At least k projects from a set S must be selected:

$$\sum_{i \in S} x_i \geq k$$

Example: At least 1 R&D project must be selected

$$x_5 + x_6 \geq 1$$

Type 6: At Most One from Set

At most k projects from a set S can be selected:

$$\sum_{i \in S} x_i \leq k$$

Example: At most 1 market project can be selected

$$x_2 + x_4 \leq 1$$

Compatibility Constraint Types (continued)

Type 7: Exactly One from Set

Exactly k projects from a set S must be selected:

$$\sum_{i \in S} x_i = k$$

Example: Exactly 1 infrastructure project must be selected

$$x_1 + x_3 = 1$$

Compatibility Constraint Types (continued)

Type 7: Exactly One from Set

Exactly k projects from a set S must be selected:

$$\sum_{i \in S} x_i = k$$

Example: Exactly 1 infrastructure project must be selected

$$x_1 + x_3 = 1$$

Type 8: Conditional Selection

If project i is selected, then at least k projects from set S must be selected:

$$\sum_{j \in S} x_j \geq k \cdot x_i$$

Example: If P1 (infrastructure) is selected, at least 2 projects that use it must be selected

$$x_3 + x_4 \geq 2 \cdot x_1$$

Extended Model with Compatibility Constraints

Complete Mathematical Formulation:

$$\max \quad z = \sum_{i=1}^{6} \text{NPV}_i \cdot x_i$$

$$\text{s.t.} \quad \sum_{i=1}^{6} \text{Cost}_i \cdot x_i \leq 300 \qquad \text{(budget)}$$

$$\sum_{i=1}^{6} \text{FTE}_i \cdot x_i \leq 7 \qquad \text{(resources)}$$

$$x_3 \leq x_1 \qquad \text{(P3 requires P1)}$$

$$x_2 + x_4 \leq 1 \qquad \text{(P2 and P4 mutually exclusive)}$$

$$x_5 + x_6 \geq 1 \qquad \text{(at least 1 R\&D project)}$$

$$x_i \in \{0, 1\} \qquad \forall i \in \{1, 2, 3, 4, 5, 6\}$$

Compatibility Matrix Visualization

Visual representation of project relationships:

	P1	**P2**	**P3**	**P4**	**P5**	**P6**
P1	-	1	$\rightarrow$	1	1	1
P2	1	-	1	0	1	1
P3	$\leftarrow$	1	-	1	1	1
P4	1	0	1	-	1	1
P5	1	1	1	1	-	$\leftrightarrow$
P6	1	1	1	1	$\leftrightarrow$	-

Legend:

- 1 = Compatible (can be selected together)
- 0 = Mutually exclusive (cannot both be selected)
- $\rightarrow$ = Row requires column (P1 $\rightarrow$ P3)
- $\leftarrow$ = Column requires row (P3 $\rightarrow$ P1)
- $\leftrightarrow$ = Must be selected together (P5 $\leftrightarrow$ P6)

Project Categories and Group Constraints

Strategic categorization:

Category	Projects	Strategic Goal
Core Infrastructure	P1, P3	Foundation for growth
Market Expansion	P2, P4	Revenue generation
R&D Innovation	P5, P6	Long-term capability

Project Categories and Group Constraints

Strategic categorization:

Category	Projects	Strategic Goal
Core Infrastructure	P1, P3	Foundation for growth
Market Expansion	P2, P4	Revenue generation
R&D Innovation	P5, P6	Long-term capability

Additional strategic constraints:

1. **Diversification requirement:**

   At least 1 project from each category

2. **Category budget allocation:**

   At least 100k€ for Core Infrastructure

$$120x_1 + 150x_3 \geq 100 \cdot (x_1 + x_3)$$

Complex Compatibility Scenario

Real business case:

Technology dependencies:

- P1 = Cloud infrastructure platform
- P3 = AI-powered analytics (needs P1 cloud)
- P5 = Machine learning R&D (needs P1 cloud)

Constraints:

$$x_3 \leq x_1 \quad \text{and} \quad x_5 \leq x_1$$

Complex Compatibility Scenario

Real business case:

Technology dependencies:

- P1 = Cloud infrastructure platform
- P3 = AI-powered analytics (needs P1 cloud)
- P5 = Machine learning R&D (needs P1 cloud)

Constraints:

$$x_3 \leq x_1 \quad \text{and} \quad x_5 \leq x_1$$

Market cannibalization:

- P2 = Mobile app for retail
- P4 = Web platform for retail
- Both target same customers $\rightarrow$ choose one strategy

Constraint:

$$x_2 + x_4 \leq 1$$

Complex Compatibility Scenario (continued)

Synergy bonus:
If both P5 (ML R&D) and P6 (Data Science R&D) are selected, there's a synergy bonus of 50k€:

Introduce auxiliary variable:

$$y_{56} \in \{0,1\} = \begin{cases} 1 & \text{if both P5 and P6 are selected} \\ 0 & \text{otherwise} \end{cases}$$

Linearization:

$$y_{56} \leq x_5$$
$$y_{56} \leq x_6$$
$$y_{56} \geq x_5 + x_6 - 1$$

Modified objective:

$$\max \quad z = \sum_{i=1}^{6} \text{NPV}_i \cdot x_i + 50 \cdot y_{56}$$

Complete Model with All Constraints

$$\max \quad z = \sum_{i=1}^{6} \text{NPV}_i \cdot x_i + 50 \cdot y_{56}$$

s.t.

$$\sum_{i=1}^{6} \text{Cost}_i \cdot x_i \leq 300 \qquad \text{(budget)}$$

$$\sum_{i=1}^{6} \text{FTE}_i \cdot x_i \leq 7 \qquad \text{(resources)}$$

$$x_3 \leq x_1 \qquad \text{(P3 requires P1)}$$

$$x_5 \leq x_1 \qquad \text{(P5 requires P1)}$$

$$x_2 + x_4 \leq 1 \qquad \text{(P2, P4 mutually exclusive)}$$

$$x_5 + x_6 \geq 1 \qquad \text{(at least 1 R\&D)}$$

$$x_1 + x_3 \geq 1 \qquad \text{(at least 1 Core)}$$

$$x_2 + x_4 \geq x_5 + x_6 \qquad \text{(balance Market/R\&D)}$$

$$y_{56} \leq x_5, \quad y_{56} \leq x_6, \quad y_{56} \geq x_5 + x_6 - 1 \qquad \text{(synergy)}$$

$$x_i \in \{0,1\}, \quad y_{56} \in \{0,1\}$$

Example 2: Solution

Implementation in python

https://github.com/alessioscl/OR25-26/blob/main/Esempio_2_Knapsack_Selezione_Progetti.ipynb

Example 3: Production Planning with Setup

Context:
A company must plan the production of 3 products over 4 weeks.
Products: $P = \{1, 2, 3\}$, Time slots: $T = \{1, 2, 3, 4\}$

Data:

Product	Production cost (€/unit)	Setup cost (€)	Capacity (units/week)
P1	20	500	100
P2	25	700	80
P3	30	600	90

Warehouse cost: 2€/unit/week

Important: If you decide to produce a product in a week, you must bear the setup cost (even if you produce only 1 unit).

Example 3: Question

Expected demand for each product (unit): $d_{pt} \quad \forall\, p \in P, t \in T$

	Week 1	Week 2	Week 3	Week 4
Product 1	50	60	40	70
Product 2	40	50	30	50
Product 3	30	40	50	45

Additional constraints:

- Initial inventory $= 0$ for all products
- Desired final inventory $= 0$ for all products
- Each week's demand must be met

Objective: Minimize total costs (production + setup + inventory).

Example 3: Modeling - Step 1

STEP 1: Identify decision variables

Example 3: Modeling - Step 1

STEP 1: Identify decision variables

Two types of decisions:

1. How much of each product should be produced each week?
2. Should a product be produced in a given week or not?

Example 3: Modeling - Step 1

STEP 1: Identify decision variables

Two types of decisions:

1. How much of each product should be produced each week?
2. Should a product be produced in a given week or not?

Decision variables:
Continuous variables:

- x_{pt} = units of product p produced in week t
- I_{pt} = inventory of product p at the end of week t

Example 3: Modeling - Step 1

STEP 1: Identify decision variables

Two types of decisions:

1. How much of each product should be produced each week?
2. Should a product be produced in a given week or not?

Decision variables:
Continuous variables:

- x_{pt} = units of product p produced in week t
- I_{pt} = inventory of product p at the end of week t

Binary variables (NEW!):

- $y_{pt} \in \{0, 1\} = 1$ if we produce product p in week t, 0 otherwise

REMINDER: where $p \in \{1, 2, 3\}$ and $t \in \{1, 2, 3, 4\}$

Example 3: Modeling - Step 2

STEP 2: Define the objective function

Example 3: Modeling - Step 2

STEP 2: Define the objective function

Three cost components:

1. Production cost c_p^{prod}
2. Setup cost (if we produce) c_p^{setup}
3. Inventory cost c^{inv}

Example 3: Modeling - Step 2

STEP 2: Define the objective function

Three cost components:

1. Production cost c_p^{prod}
2. Setup cost (if we produce) c_p^{setup}
3. Inventory cost c^{inv}

Objective function:

$$\min \quad z = \sum_{p=1}^{3} \sum_{t=1}^{4} c_p^{prod} \cdot x_{pt} \qquad \text{(production cost)}$$

$$+ \sum_{p=1}^{3} \sum_{t=1}^{4} c_p^{setup} \cdot y_{pt} \qquad \text{(setup cost)}$$

$$+ \sum_{p=1}^{3} \sum_{t=1}^{4} c^{inv} \cdot I_{pt} \qquad \text{(inventory cost)}$$

$$\min \quad z = \sum_{p \in P} \sum_{t \in T} c_p^{prod} \cdot x_{pt} + c_p^{setup} \cdot y_{pt} + c^{inv} \cdot I_{pt}$$

Example 3: Modeling - Step 3a

STEP 3: Formulate constraints

A. Inventory balancing

For each product and each week:

$$I_{p,t-1} + x_{pt} = d_{pt} + I_{pt}$$

where d_{pt} is the demand for product p in week t

Example 3: Modeling - Step 3a

STEP 3: Formulate constraints

A. Inventory balancing

For each product and each week:

$$I_{p,t-1} + x_{pt} = d_{pt} + I_{pt}$$

where d_{pt} is the demand for product p in week t

B. Initial and final inventory

$$I_{p,0} = 0 \quad \forall p \qquad \text{(no initial inventory)}$$

$$I_{p,4} = 0 \quad \forall p \qquad \text{(no final inventory)}$$

Example 3: Modeling - Step 3b

C. Production capacity constraint

If we produce, we cannot exceed capacity:

$$x_{pt} \leq \text{cap}_p \quad \forall p, t$$

Example 3: Modeling - Step 3b

C. Production capacity constraint

If we produce, we cannot exceed capacity:

$$x_{pt} \leq \text{cap}_p \quad \forall p, t$$

D. Production-setup link (Big-M)

Idea: If we do not produce $(y_{pt} = 0)$, then x_{pt} must be 0.

Example 3: Modeling - Step 3b

C. Production capacity constraint

If we produce, we cannot exceed capacity:

$$x_{pt} \leq \text{cap}_p \quad \forall p, t$$

D. Production-setup link (Big-M)

Idea: If we do not produce $(y_{pt} = 0)$, then x_{pt} must be 0.

$$x_{pt} \leq \text{cap}_p \cdot y_{pt} \quad \forall p, t$$

Example 3: Modeling - Step 3b

C. Production capacity constraint

If we produce, we cannot exceed capacity:

$$x_{pt} \leq \text{cap}_p \quad \forall p, t$$

D. Production-setup link (Big-M)

Idea: If we do not produce $(y_{pt} = 0)$, then x_{pt} must be 0.

$$x_{pt} \leq \text{cap}_p \cdot y_{pt} \quad \forall p, t$$

Explanation:

- If $y_{pt} = 0 \rightarrow x_{pt} \leq 0 \rightarrow x_{pt} = 0$
- If $y_{pt} = 1 \rightarrow x_{pt} \leq \text{cap}_p$ (normal capacity)

Example 3: Modeling - Step 3b

C. Production capacity constraint

If we produce, we cannot exceed capacity:

$$x_{pt} \leq \text{cap}_p \quad \forall p, t$$

D. Production-setup link (Big-M)

Idea: If we do not produce $(y_{pt} = 0)$, then x_{pt} must be 0.

$$x_{pt} \leq \text{cap}_p \cdot y_{pt} \quad \forall p, t$$

Explanation:

- If $y_{pt} = 0 \rightarrow x_{pt} \leq 0 \rightarrow x_{pt} = 0$
- If $y_{pt} = 1 \rightarrow x_{pt} \leq \text{cap}_p$ (normal capacity)

This is the classic "Big-M" constraint

Example 3: Complete Model

Complete Mathematical Formulation:

$$\min \quad z = \sum_{p \in P} \sum_{t \in T} c_p^{prod} \cdot x_{pt} + c_p^{setup} \cdot y_{pt} + c^{inv} \cdot I_{pt}$$

$$\begin{aligned}
\text{s.t.} \quad & I_{p,t-1} + x_{pt} = d_{pt} + I_{pt} && \forall p \in P, t \in T \\
& I_{p0} = 0 && \forall p \in P \\
& I_{p4} = 0 && \forall p \in P \\
& x_{pt} \leq \mathsf{cap}_p \cdot y_{pt} && \forall p \in P, t \in T \\
& x_{pt}, I_{pt} \geq 0 && \forall p \in P, t \in T \\
& y_{pt} \in \{0, 1\} && \forall p \in P, t \in T
\end{aligned}$$

Example 3: Complete Model

Complete Mathematical Formulation:

$$\min \quad z = \sum_{p \in P} \sum_{t \in T} c_p^{prod} \cdot x_{pt} + c_p^{setup} \cdot y_{pt} + c^{inv} \cdot I_{pt}$$

$$\begin{aligned}
\text{s.t.} \quad & I_{p,t-1} + x_{pt} = d_{pt} + I_{pt} && \forall p \in P, t \in T \\
& I_{p0} = 0 && \forall p \in P \\
& I_{p4} = 0 && \forall p \in P \\
& x_{pt} \leq \mathsf{cap}_p \cdot y_{pt} && \forall p \in P, t \in T \\
& x_{pt}, I_{pt} \geq 0 && \forall p \in P, t \in T \\
& y_{pt} \in \{0, 1\} && \forall p \in P, t \in T
\end{aligned}$$

Model type: Mixed-Integer Linear Programming (MILP)

- Continuous (x_{pt}, I_{pt}) and binary (y_{pt}) variables
- Big-M constraints for decision logic

Example 3: Solution

Implementation in python

`https://github.com/alessioscl/OR25-26/blob/main/Esempio_3_Pianificazione_Setup.ipynb`

Recap: Modeling Techniques

General methodology:

1. Identify decision variables
2. Define objective function
3. Formulate all constraints
4. Implement in Gurobi
5. Interpret and validate results

Best Practices:

- **Start with the real problem**, not the math
- **Use clear notation**: define indices and variables well
- **Validate the model**: check that constraints make sense
- **Test with simple data**: verify obvious solutions
- **Document the code**: your future self will thank you!

Further Resources

Gurobi Documentation:

- `gurobi.com/documentation`
- Quick Start Guide for Python
- Examples library

Recommended Books:

- H. Paul Williams - "Model Building in Mathematical Programming"
- Winston - "Operations Research: Applications and Algorithms"

Vehicle Routing Problem (VRP) - Introduction

Definition:
The VRP determines optimal routes for a fleet of vehicles to serve a set of customers from a central depot, minimizing total distance/cost.

Key characteristics:

- **Routing**: Determine sequence of visits (not just quantities)
- **Vehicle constraints**: Limited capacity, limited vehicles
- **Topology**: Each customer visited exactly once
- **Closed routes**: Vehicles start and end at depot

Vehicle Routing Problem (VRP) - Introduction

Definition:
The VRP determines optimal routes for a fleet of vehicles to serve a set of customers from a central depot, minimizing total distance/cost.

Key characteristics:

- **Routing**: Determine sequence of visits (not just quantities)
- **Vehicle constraints**: Limited capacity, limited vehicles
- **Topology**: Each customer visited exactly once
- **Closed routes**: Vehicles start and end at depot

Complexity:

- Transportation Problem: **Polynomial** time (easy)
- VRP: **NP-hard** (very difficult!)
- TSP (Traveling Salesman) is a special case of VRP with 1 vehicle

VRP Network Representation

Graph structure:

- **Nodes**:
  - Node 0: Depot (starting/ending point)
  - Nodes 1,2,3,4: Customers (A, B, C, D)
- **Arcs**: (i, j) represents travel from node i to node j
- **Arc costs**: c_{ij} = distance or travel time from i to j
- **Demands**: q_i = demand at customer i
- **Vehicle capacity**: Q = maximum load per vehicle

VRP Network Representation

Graph structure:

- **Nodes**:
  - Node 0: Depot (starting/ending point)
  - Nodes 1,2,3,4: Customers (A, B, C, D)
- **Arcs**: (i, j) represents travel from node i to node j
- **Arc costs**: c_{ij} = distance or travel time from i to j
- **Demands**: q_i = demand at customer i
- **Vehicle capacity**: Q = maximum load per vehicle

Example route for vehicle 1:

$$0 \to A \to C \to 0$$

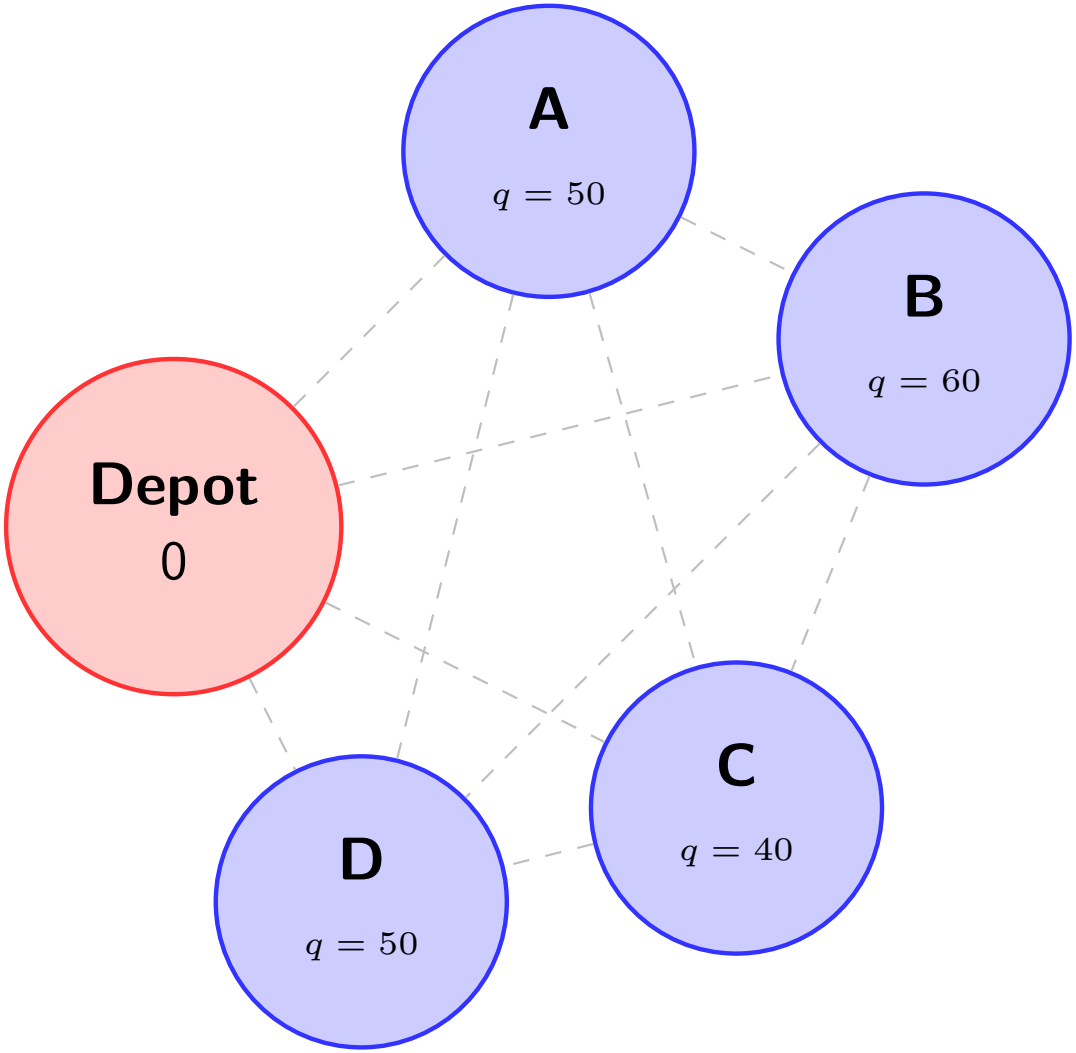
Depot $\to$ Customer A $\to$ Customer C $\to$ Depot

Example route for vehicle 2:

$$0 \to B \to D \to 0$$

Depot $\to$ Customer B $\to$ Customer D $\to$ Depot

VRP Network Visualization



Depot: Starting point

Customers: A, B, C, D

q_i: Demand (units)

Capacity: $Q = 120$

CVRP: Sets and Parameters

Capacitated Vehicle Routing Problem (CVRP)

SETS:

$V = \{0, 1, 2, \ldots, n\}$	nodes (0 = depot, $1..n$ = customers)
$K = \{1, 2, \ldots, m\}$	vehicles
$A = \{(i, j) : i, j \in V, i \neq j\}$	arcs (directed)

PARAMETERS:

$c_{ij} \geq 0$	distance/cost from node i to node j
$q_i \geq 0$	demand at customer i (units)
$Q > 0$	vehicle capacity (units)
$n > 0$	number of customers
$m > 0$	number of vehicles available

Assumption: $q_0 = 0$ (depot has no demand)

CVRP: Decision Variables

PRIMARY VARIABLES (routing):

$$x_{ij}^k \in \{0,1\} \qquad \begin{cases} 1 & \text{if vehicle } k \text{ travels from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

Number of variables: $|V|^2 \times |K|$
For 5 nodes and 3 vehicles: $5^2 \times 3 = 75$ binary variables

CVRP: Decision Variables

PRIMARY VARIABLES (routing):

$$x_{ij}^k \in \{0,1\} \qquad \begin{cases} 1 & \text{if vehicle } k \text{ travels from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

Number of variables: $|V|^2 \times |K|$
For 5 nodes and 3 vehicles: $5^2 \times 3 = 75$ binary variables

AUXILIARY VARIABLES (for subtour elimination):

$$u_i \geq 0 \qquad \text{position of node } i \text{ in tour (MTZ variable)}$$

Interpretation:

- $u_0 = 0$ (depot is position 0)
- $u_i \in [1, n]$ for customers
- If vehicle visits i before j, then $u_i < u_j$
- Used to eliminate subtours

CVRP: Objective Function

OBJECTIVE: Minimize total distance traveled by all vehicles

$$\min \quad z = \sum_{k \in K} \sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij} \cdot x_{ij}^{k}$$

CVRP: Objective Function

OBJECTIVE: Minimize total distance traveled by all vehicles

$$\min \quad z = \sum_{k \in K} \sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij} \cdot x_{ij}^k$$

Interpretation:

- Sum over all vehicles k
- Sum over all possible arcs (i, j)
- c_{ij}: Distance from i to j
- $x_{ij}^k = 1$: If arc is used by vehicle k
- Only used arcs contribute to objective

CVRP: Objective Function

OBJECTIVE: Minimize total distance traveled by all vehicles

$$\min \quad z = \sum_{k \in K} \sum_{i \in V} \sum_{j \in V, j \neq i} c_{ij} \cdot x_{ij}^k$$

Interpretation:

- Sum over all vehicles k
- Sum over all possible arcs (i, j)
- c_{ij}: Distance from i to j
- $x_{ij}^k = 1$: If arc is used by vehicle k
- Only used arcs contribute to objective

Alternative objectives:

- Minimize number of vehicles used
- Minimize maximum route length (balanced workload)
- Minimize total time (with time-dependent travel times)

CVRP: Constraints - Part 1 (Coverage)

1. Each customer visited exactly once:

$$\sum_{k \in K} \sum_{i \in V, i \neq j} x_{ij}^k = 1 \quad \forall j \in V \setminus \{0\}$$

Meaning: Sum of all incoming arcs to customer j across all vehicles $= 1$

CVRP: Constraints - Part 1 (Coverage)

1. Each customer visited exactly once:

$$\sum_{k \in K} \sum_{i \in V, i \neq j} x_{ij}^k = 1 \quad \forall j \in V \setminus \{0\}$$

Meaning: Sum of all incoming arcs to customer j across all vehicles $= 1$

2. Flow conservation (what enters must exit):

$$\sum_{i \in V, i \neq h} x_{ih}^k = \sum_{j \in V, j \neq h} x_{hj}^k \quad \forall h \in V, \forall k \in K$$

Meaning: For each node h and vehicle k:

- Number of arcs entering h = Number of arcs leaving h
- If vehicle arrives at h, it must also leave h
- Ensures continuous routes

CVRP: Constraints - Part 2 (Capacity)

3. Vehicle capacity constraint:

$$\sum_{j \in V \setminus \{0\}} q_j \cdot \sum_{i \in V, i \neq j} x_{ij}^k \leq Q \quad \forall k \in K$$

Meaning: Total demand served by vehicle k cannot exceed capacity Q

CVRP: Constraints - Part 2 (Capacity)

3. Vehicle capacity constraint:

$$\sum_{j \in V \setminus \{0\}} q_j \cdot \sum_{i \in V, i \neq j} x_{ij}^k \leq Q \quad \forall k \in K$$

Meaning: Total demand served by vehicle k cannot exceed capacity Q

4. Depot constraints:

$$\sum_{j \in V \setminus \{0\}} x_{0j}^k \leq 1 \qquad \text{(each vehicle leaves depot at most once)}$$

$$\sum_{i \in V \setminus \{0\}} x_{i0}^k \leq 1 \qquad \text{(each vehicle returns to depot at most once)}$$

Meaning: Vehicles either stay at depot or make exactly one trip

CVRP: Constraints - Part 3 (Subtour Elimination)

Problem: Without additional constraints, we could get **subtours**

Example of infeasible solution:

- Route 1: $0 \to A \to 0$
- Route 2: $B \to C \to D \to B$ (subtour!)

Route 2 doesn't include depot $\to$ not a valid vehicle route!

CVRP: Constraints - Part 3 (Subtour Elimination)

Problem: Without additional constraints, we could get **subtours**

Example of infeasible solution:

- Route 1: $0 \to A \to 0$
- Route 2: $B \to C \to D \to B$ (subtour!)

Route 2 doesn't include depot $\to$ not a valid vehicle route!

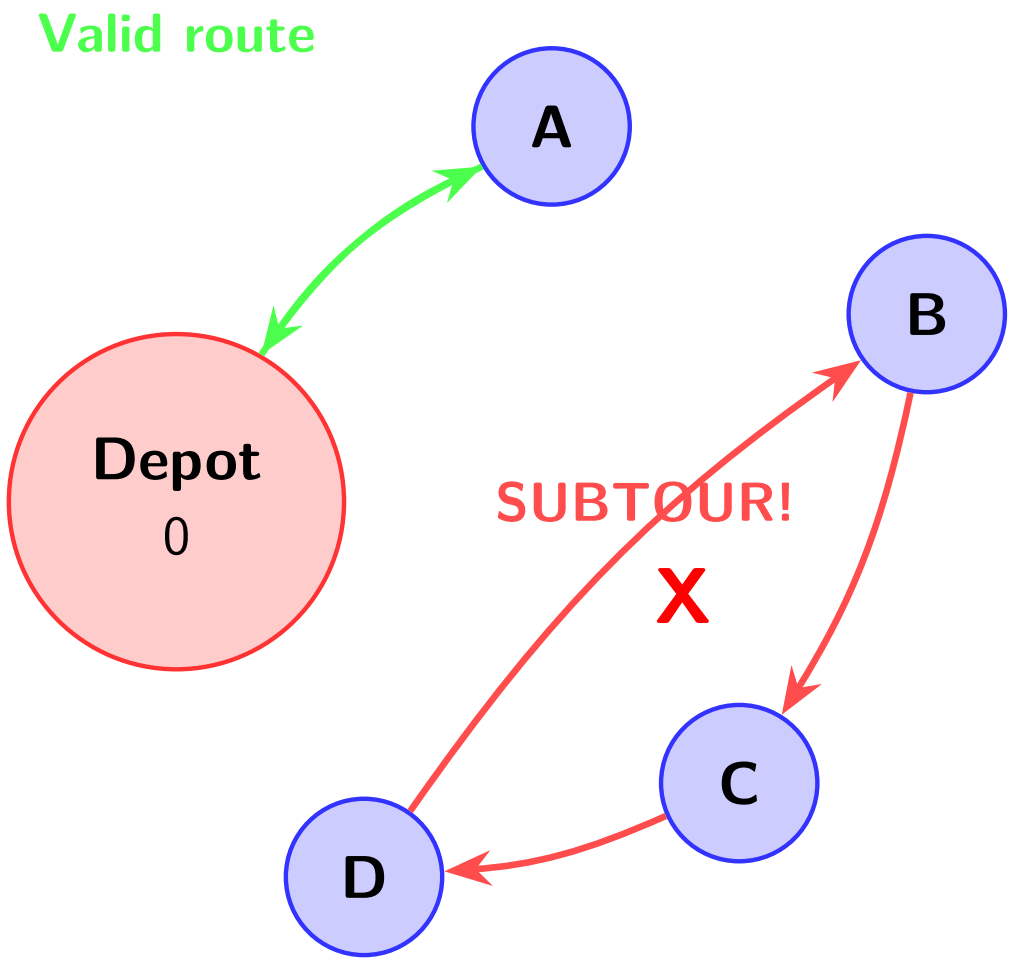
Miller-Tucker-Zemlin (MTZ) constraints:

$$u_i - u_j + n \sum_{k \in K} x_{ij}^k \leq n - 1 \quad \forall i, j \in V \setminus \{0\}, i \neq j$$

How it works:

- u_i = position of customer i in route
- If $x_{ij}^k = 1$ (arc used), then $u_i < u_j$ (enforces order)
- Prevents cycles that don't include depot
- $u_0 = 0$ (depot is first), $1 \leq u_i \leq n$ for customers

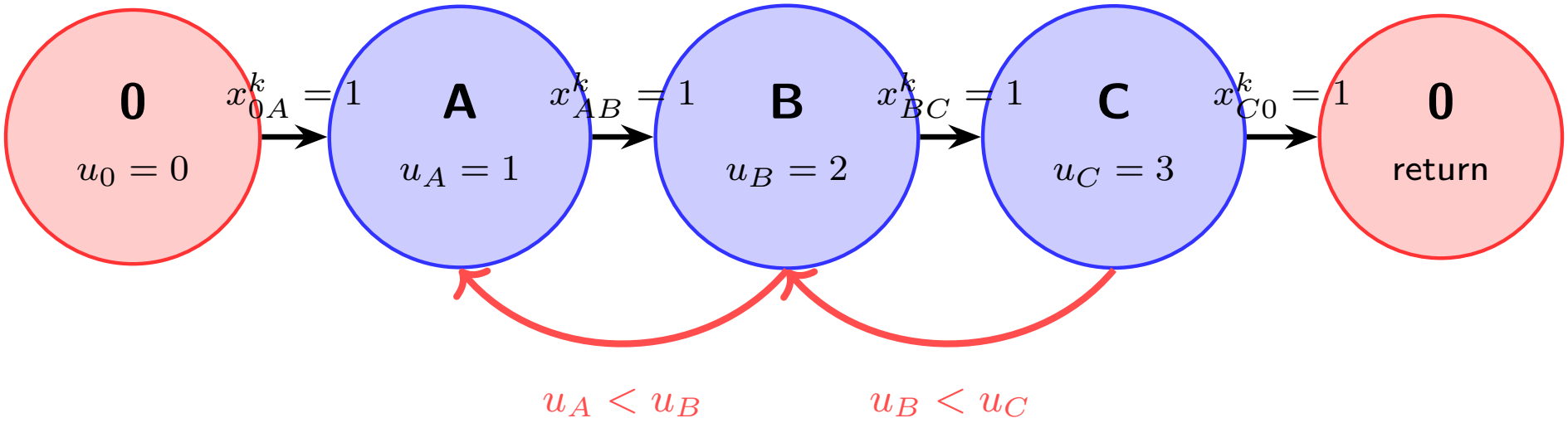
Problem: Subtour Example (Invalid Solution)



Problem:
Route B $\rightarrow$ C $\rightarrow$ D $\rightarrow$ B doesn't include the depot!

Not a valid vehicle route

MTZ Constraint Visualization



Variables enforce visiting order:

- $u_0 = 0$ (depot is starting position)
- $u_A = 1$ (customer A visited first)
- $u_B = 2$ (customer B visited second)
- $u_C = 3$ (customer C visited third)

Effect: If arc (i, j) is used ($x^k_{ij} = 1$), then $u_i < u_j$ (ensures proper sequence)

Miller-Tucker-Zemlin (MTZ) Constraint

Case 1: Arc is USED ($x_{ij}^k = 1$)

Mathematical simplification: Interpretation:
If vehicle uses arc (i, j), then customer i MUST be visited before customer j

Numerical example with $n = 4$:
Suppose $u_A = 2$ (A is 2nd customer)
Then for arc A $\to$ B with $x_{AB}^k = 1$:

$$\begin{aligned} 2 - u_B + 4 \cdot 1 &\leq 4 - 1 \\ 2 - u_B + 4 &\leq 3 \\ 6 - u_B &\leq 3 \\ -u_B &\leq -3 \\ u_B &\geq 3 \end{aligned}$$

Since $u_B \leq n = 4$: $u_B \in \{3, 4\}$
$\Rightarrow$ B must be visited after A (positions 3 or 4)

Case 2: Arc is NOT USED ($x_{ij}^k = 0$)

Mathematical simplification: Interpretation:
Constraint becomes very weak (almost always satisfied)

Numerical example with $n = 4$:
Suppose $u_A = 2$ and $u_C = 3$
For arc A $\to$ C with $x_{AC}^k = 0$:

$$\begin{aligned} 2 - 3 + 4 \cdot 0 &\leq 4 - 1 \\ -1 &\leq 3 \quad \checkmark \end{aligned}$$

Even if $u_C = 1$ and $u_A = 4$:

$$\begin{aligned} 4 - 1 + 4 \cdot 0 &\leq 4 - 1 \\ 3 &\leq 3 \quad \checkmark \end{aligned}$$

CVRP: Complete Mathematical Model

$$\min \quad z = \sum_{k \in K} \sum_{i \in V} \sum_{j \in V \setminus \{i\}} c_{ij} \cdot x_{ij}^k$$

s.t.

$$\sum_{k \in K} \sum_{i \in V \setminus \{j\}} x_{ij}^k = 1 \qquad \forall j \in V \setminus \{0\} \qquad \text{(each customer visited once)}$$

$$\sum_{i \in V \setminus \{h\}} x_{ih}^k = \sum_{j \in V \setminus \{h\}} x_{hj}^k \qquad \forall h \in V, k \in K \qquad \text{(flow conservation)}$$

$$\sum_{j \in V \setminus \{0\}} q_j \sum_{j \in V \setminus \{i\}} x_{ij}^k \leq Q \qquad \forall k \in K \qquad \text{(capacity)}$$

$$\sum_{j \in V \setminus \{0\}} x_{0j}^k \leq 1 \qquad \forall k \in K \qquad \text{(leave depot once)}$$

$$\sum_{i \in V \setminus \{0\}} x_{i0}^k \leq 1 \qquad \forall k \in K \qquad \text{(return to depot once)}$$

$$u_i - u_j + n \sum_{k \in K} x_{ij}^k \leq n - 1 \qquad \forall i, j \in V \setminus \{0\}, i \neq j \qquad \text{(MTZ subtour)}$$

$$u_0 = 0, \quad 1 \leq u_i \leq |V| \qquad \forall i \in V \setminus \{0\} \qquad \text{(MTZ bounds)}$$

$$x_{ij}^k \in \{0, 1\} \qquad \forall i, j \in V, k \in K$$

Example: 4 Customers, 2 Vehicles

Data:

- Depot: Node 0
- Customers: A(1), B(2), C(3), D(4)
- Demands: $q_1 = 50, q_2 = 60, q_3 = 40, q_4 = 50$
- Vehicle capacity: $Q = 120$ units
- Number of vehicles: $m = 2$

Distance matrix (simplified):

	0	**A(1)**	**B(2)**	**C(3)**	**D(4)**
0	-	10	15	20	25
A(1)	10	-	8	18	22
B(2)	15	8	-	12	16
C(3)	20	18	12	-	10
D(4)	25	22	16	10	-

Example: 4 Customers, 2 Vehicles

Data:

- Depot: Node 0
- Customers: A(1), B(2), C(3), D(4)
- Demands: $q_1 = 50, q_2 = 60, q_3 = 40, q_4 = 50$
- Vehicle capacity: $Q = 120$ units
- Number of vehicles: $m = 2$

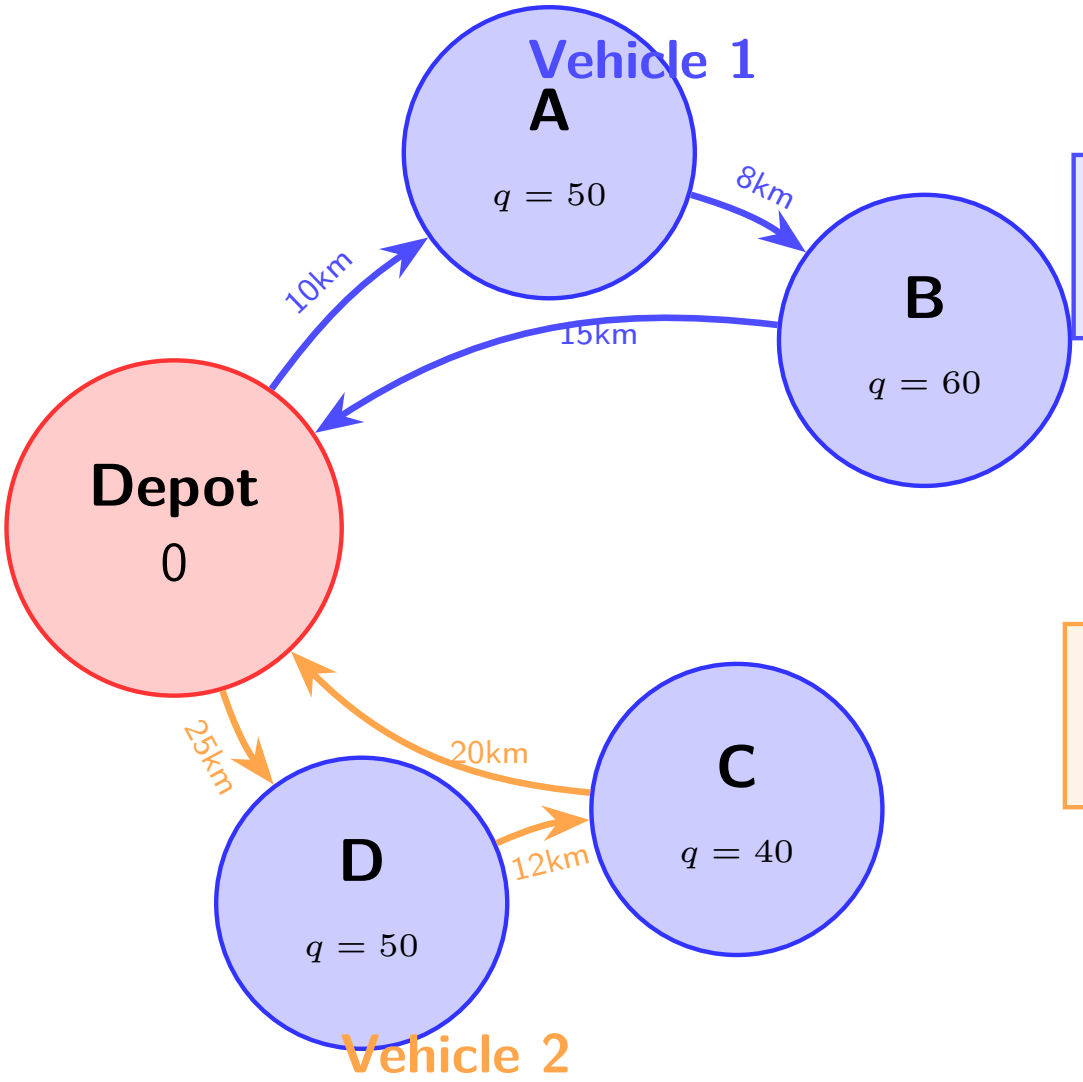
Distance matrix (simplified):

	0	**A(1)**	**B(2)**	**C(3)**	**D(4)**
0	-	10	15	20	25
A(1)	10	-	8	18	22
B(2)	15	8	-	12	16
C(3)	20	18	12	-	10
D(4)	25	22	16	10	-

Feasible solution example:

- Vehicle 1: $0 \to A \to B \to 0$ (load: 50+60=110)
- Vehicle 2: $0 \to D \to C \to 0$ (load: 40+50=90)
- Total distance: 10+8+15 + 25+10+20 = 88 km

Example Solution: Two Vehicle Routes



Vehicle 1 (blue):
Route: 0→A→B→0
Load: 50+60=110 $\leq$ 120
Distance: 10+8+15=33 km

Vehicle 2 (orange):
Route: 0→ D → C → 0
Load: 50+40=90 $\leq$ 120
Distance: 25+10+20=57 km

Total: 88 km

Model Complexity Analysis

Problem size growth:

Customers	Vehicles	Binary vars	Constraints
4	2	$5^2 \times 2 = 50$	$\sim$80
10	3	$11^2 \times 3 = 363$	$\sim$500
20	5	$21^2 \times 5 = 2,205$	$\sim$3,000
50	10	$51^2 \times 10 = 26,010$	$\sim$30,000

Computational challenge:

- Binary variables: $2^{50} \approx 10^{15}$ combinations!
- MTZ constraints: $O(n^2)$ growth
- For 50 customers: Often need heuristics or decomposition

Key Takeaways

Main lessons about VRP:

1. **Complexity jump**: Transportation (easy) - VRP (hard)
2. **Routing adds difficulty**: Sequence matters, not just quantities
3. **Subtours are tricky**: Need careful constraint formulation
4. **MTZ constraints**: Classic technique but adds many constraints
5. **Many variants**: Real world has many different VRP types
6. **Solution methods matter**: Exact vs heuristic trade-offs
7. **Practical impact**: Billion-dollar savings for logistics companies

VRP is one of the most studied problems in Operations Research!

Discussion Questions

1. Why can't we solve VRP with 100 customers to optimality in reasonable time?
2. What happens if we remove MTZ constraints? What kind of invalid solutions might we get?
3. How would you modify the model if vehicles have different capacities?
4. A customer wants delivery between 9am-11am. How to add time windows?
5. If we allow split deliveries (customer visited by 2 vehicles), which constraint changes?
6. How would you model a scenario where vehicles can refuel at certain nodes?

Advanced Example: Operating Room Scheduling with Surgeon Fatigue

Context:
A hospital must schedule surgical interventions across multiple operating rooms and surgeons, considering:

- Surgeon skill levels and availability
- Operating room capacity
- Intervention difficulty and duration
- **Surgeon fatigue** (increases with workload)
- Temporal precedence constraints

Complexity:

- Assignment problem (intervention $\rightarrow$ surgeon, intervention $\rightarrow$ room)
- Scheduling problem (determine start times and order)

Goal: Maximize surgical quality considering skill matching and fatigue minimization

Problem Characteristics

Key Features:

1. **Multi-dimensional assignment:**
   - Each intervention must be assigned to exactly ONE surgeon
   - Each intervention must be assigned to exactly ONE operating room

2. **Temporal scheduling:**
   - Determine start time for each intervention
   - Respect precedence relationships
   - No overlaps for same surgeon or same room

3. **Fatigue modeling:**
   - Surgeon fatigue accumulates based on difficulty and duration of previous interventions
   - Fatigue reduces effective surgical quality

Sets and Input Parameters

SETS:

$K = \{1, 2, ...\}$	set of operating rooms
$J = \{1, 2, ...\}$	set of surgeons
$I = \{1, 2, ...\}$	set of surgical interventions

PARAMETERS:

$t_i \in \mathbb{R}^+$	duration of intervention i (hours)
$d_i \in [0, 1]$	difficulty level of intervention i
$s_j \in [0, 1]$	skill level of surgeon j
$a_j \in [0, 1]$	normalized age of surgeon j
$W^j_{\max} \in \mathbb{R}^+$	max working time for surgeon j
$T^k_{\max} \in \mathbb{R}^+$	max usage time for room k
$H \in \mathbb{R}^+$	global scheduling horizon (e.g., 16h shift)
$M \in \mathbb{R}^+$	big-M constant for disjunctive constraints

Decision Variables

BINARY VARIABLES (assignment and precedence):

$x_{ij} \in \{0,1\}$	1 if intervention i assigned to surgeon j
$y_{ik} \in \{0,1\}$	1 if intervention i assigned to room k
$z_{ih} \in \{0,1\}$	1 if intervention i precedes intervention h
$p_{hij} \in \{0,1\}$	auxiliary variable for linearization

CONTINUOUS VARIABLES (timing and fatigue):

$v_i \geq 0$	start time of intervention i
$f_{ij} \geq 0$	fatigue level of surgeon j before intervention i

Objective Function - Conceptual

MAXIMIZE surgical quality adjusted for fatigue:

Original (nonlinear) formulation:

$$\max \sum_{i \in I} \sum_{j \in J} d_i \cdot s_j \cdot (1 - f_{ij}) \cdot x_{ij}$$

Components:

- d_i: Difficulty of intervention i (higher difficulty = more value)
- s_j: Skill level of surgeon j (higher skill = better outcomes)
- f_{ij}: Fatigue factor (higher fatigue = reduced quality)
- $(1 - f_{ij})$: Quality reduction due to fatigue
- x_{ij}: Assignment variable

Intuition: Match difficult interventions with skilled surgeons, while minimizing the impact of fatigue

Fatigue Model

Fatigue accumulation:

Fatigue of surgeon j before performing intervention i depends on all previous interventions:

$$f_{ij} = \frac{1}{W^j_{\max}} \sum_{h \in I \setminus \{i\}} d_h \cdot t_h \cdot x_{hj} \cdot z_{hi}$$

Interpretation:

- $x_{hj} \cdot z_{hi}$: surgeon j performed intervention h BEFORE intervention i
- $d_h \cdot t_h$: workload of intervention h (difficulty x duration)
- Sum accumulates all previous workload
- Normalized by $W^j_{\max}$ (relative to surgeon's capacity)

Issue: Product $x_{hj} \cdot z_{hi}$ is **nonlinear**!
$\Rightarrow$ Need linearization technique

Linearization Technique

Solution: Introduce auxiliary binary variable $p_{hij} = x_{hj} \cdot z_{hi}$

Add linearization constraints:

$$p_{hij} \leq x_{hj} \qquad \forall i, h \in I, i \neq h, \forall j \in J$$
$$p_{hij} \leq z_{hi} \qquad \forall i, h \in I, i \neq h, \forall j \in J$$
$$p_{hij} \geq x_{hj} + z_{hi} - 1 \qquad \forall i, h \in I, i \neq h, \forall j \in J$$

Logic:

- If $x_{hj} = 1$ AND $z_{hi} = 1 \Rightarrow p_{hij} = 1$
- If $x_{hj} = 0$ OR $z_{hi} = 0 \Rightarrow p_{hij} = 0$

Then substitute in fatigue:

$$f_{ij} = \frac{1}{W^j_{\max}} \sum_{h \in I \setminus \{i\}} d_h \cdot t_h \cdot p_{hij}$$

Constraints - Part 1: Assignment

1. **Each intervention assigned to exactly ONE surgeon:**

$$\sum_{j \in J} x_{ij} = 1 \quad \forall i \in I$$

2. **Each intervention assigned to exactly ONE room:**

$$\sum_{k \in K} y_{ik} = 1 \quad \forall i \in I$$

3. **Surgeon working time limits:**

$$\sum_{i \in I} t_i \cdot x_{ij} \leq W^j_{\max} \quad \forall j \in J$$

4. **Operating room usage limits:**

$$\sum_{i \in I} t_i \cdot y_{ik} \leq T^k_{\max} \quad \forall k \in K$$

Constraints - Part 2: Scheduling

5. Fatigue definition:

$$f_{ij} = \frac{1}{W^j_{\max}} \sum_{h \in I \setminus \{i\}} d_h \cdot t_h \cdot p_{hij} \quad \forall i \in I, j \in J$$

6. Precedence for same surgeon (no overlap):

$$v_i + t_i \leq v_h + M \cdot (1 - z_{ih}) + M \cdot (2 - x_{ij} - x_{hj})$$
$$\forall i, h \in I, i \neq h, \forall j \in J$$

Interpretation:

- If $x_{ij} = x_{hj} = 1$ (same surgeon) AND $z_{ih} = 1$ (i before h):
- Then: $v_i + t_i \leq v_h$ (intervention i must finish before h starts)
- Otherwise: constraint relaxed by big-M

Constraints - Part 3: Room Scheduling

7. Precedence for same room (no overlap):

$$v_i + t_i \leq v_h + M \cdot (1 - z_{ih}) + M \cdot (2 - y_{ik} - y_{hk})$$
$$\forall i, h \in I, i \neq h, \forall k \in K$$

8. Antisymmetry of precedence (one must precede the other):

$$z_{ih} + z_{hi} = 1 \quad \forall i, h \in I, i \neq h$$

9. Global horizon constraint:

$$v_i + t_i \leq H \quad \forall i \in I$$

All interventions must finish within the shift horizon

Complete Mathematical Model

$$\max \quad \frac{\sum_{i \in I} \sum_{j \in J} d_i \cdot s_j \cdot x_{ij} - a_j \cdot f_{ij}}{\sum_{i \in I} d_i}$$

$$\text{s.t.} \quad \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I$$

$$\sum_{k \in K} y_{ik} = 1 \qquad \forall i \in I$$

$$\sum_{i \in I} t_i \cdot x_{ij} \leq W^j_{\max} \qquad \forall j \in J$$

$$\sum_{i \in I} t_i \cdot y_{ik} \leq T^k_{\max} \qquad \forall k \in K$$

$$f_{ij} = \frac{1}{W^j_{\max}} \sum_{h \in I \setminus \{i\}} d_h \cdot t_h \cdot p_{hij} \qquad \forall i, j$$

$$p_{hij} \leq x_{hj}, \quad p_{hij} \leq z_{hi}, \quad p_{hij} \geq x_{hj} + z_{hi} - 1 \qquad \forall i, h, j$$

$$v_i + t_i \leq v_h + M(1 - z_{ih}) + M(2 - x_{ij} - x_{hj}) \qquad \forall i, h, j$$

$$v_i + t_i \leq v_h + M(1 - z_{ih}) + M(2 - y_{ik} - y_{hk}) \qquad \forall i, h, k$$

$$z_{ih} + z_{hi} = 1 \qquad \forall i \neq h$$

Advanced Example: Multi-Level Electric Grid Network

Context:

An electric utility company must design and operate a distribution network with multiple voltage levels:

- **High voltage (H)**: Main substation
- **Medium voltage (M)**: Distribution transformers
- **Low voltage (L)**: Local transformers
- **Demand nodes (R)**: End customers

Decisions:

- Which connections (arcs) to build (budget constraint)
- How to route power flow to meet customer demand

Challenges:

- Network topology constraints (flow can only go from high to low voltage)
- Arc capacity limits
- Budget constraint for building connections

Part 1: Deterministic Model - Sets and Parameters

SETS:

$H = \{h_1\}$	high voltage nodes
$M = \{m_1, m_2, \ldots\}$	medium voltage nodes
$L = \{l_1, l_2, \ldots\}$	low voltage (transformer) nodes
$R = \{r_1, r_2, \ldots\}$	demand (customer) nodes
$N = H \cup M \cup L \cup R$	all nodes

PARAMETERS:

$c_{ij} \geq 0$	capacity of arc (i, j) (MW)
$b_{ij} \geq 0$	cost to build/open arc (i, j) (€)
$q_r \geq 0$	power demand at node r (MW)
$B \geq 0$	total budget for building arcs (€)

Deterministic Model - Decision Variables

CONTINUOUS VARIABLES (flow):

$$f_{ij} \geq 0 \qquad \text{power flow on arc } (i,j) \text{ (MW)}$$

BINARY VARIABLES (network design):

$$z_{ij} \in \{0,1\} \qquad 1 \text{ if arc } (i,j) \text{ is built/open}$$
$$y_r \in \{0,1\} \qquad 1 \text{ if demand at node } r \text{ is satisfied}$$
$$u_{rl} \in \{0,1\} \qquad 1 \text{ if demand node } r \text{ is served by transformer } l$$

Interpretation:

- $z_{ij} = 1$: We invest to build connection (i,j)
- $y_r = 1$: Customer r receives full power demand q_r
- $u_{rl} = 1$: Customer r is connected to transformer l

Deterministic Model - Objective Function

OBJECTIVE: Maximize total satisfied demand

$$\max \sum_{r \in R} q_r \cdot y_r$$

Alternative objectives:

- Minimize cost while satisfying all demand
- Maximize profit (revenue from satisfied demand - building cost)
- Minimize load shedding (unsatisfied demand)

Current formulation:

- Prioritize satisfying demand
- Subject to budget constraint
- Cannot build unlimited connections

Deterministic Model - Constraints (1/3)

1. Flow capacity constraint:

$$f_{ij} \leq z_{ij} \cdot c_{ij} \quad \forall i, j \in N$$

Meaning: Flow on arc (i, j) is only possible if arc is open ($z_{ij} = 1$) and cannot exceed capacity.

Deterministic Model - Constraints (1/3)

1. Flow capacity constraint:

$$f_{ij} \leq z_{ij} \cdot c_{ij} \quad \forall i, j \in N$$

Meaning: Flow on arc (i, j) is only possible if arc is open ($z_{ij} = 1$) and cannot exceed capacity.

2. Flow conservation at medium voltage nodes:

$$\sum_{i \in H \cup M} f_{im} = \sum_{j \in M \cup L} f_{mj} \quad \forall m \in M$$

Meaning: Inflow = Outflow (no generation or consumption at M nodes)

Deterministic Model - Constraints (1/3)

1. **Flow capacity constraint:**

$$f_{ij} \leq z_{ij} \cdot c_{ij} \quad \forall i, j \in N$$

Meaning: Flow on arc (i, j) is only possible if arc is open ($z_{ij} = 1$) and cannot exceed capacity.

2. **Flow conservation at medium voltage nodes:**

$$\sum_{i \in H \cup M} f_{im} = \sum_{j \in M \cup L} f_{mj} \quad \forall m \in M$$

Meaning: Inflow = Outflow (no generation or consumption at M nodes)

3. **Flow conservation at low voltage nodes:**

$$\sum_{i \in M \cup L} f_{il} = \sum_{j \in L \cup R} f_{lj} \quad \forall l \in L$$

Meaning: Inflow = Outflow (no generation or consumption at L nodes)

Deterministic Model - Constraints (2/3)

4. Demand satisfaction:

$$\sum_{l \in L} f_{lr} = q_r \cdot y_r \quad \forall r \in R$$

Meaning: If $y_r = 1$ (demand satisfied), then total inflow to r must equal demand q_r

Deterministic Model - Constraints (2/3)

4. Demand satisfaction:

$$\sum_{l \in L} f_{lr} = q_r \cdot y_r \quad \forall r \in R$$

Meaning: If $y_r = 1$ (demand satisfied), then total inflow to r must equal demand q_r

5. Single transformer assignment:

$$\sum_{l \in L} u_{rl} = y_r \quad \forall r \in R$$

Meaning: If demand r is served ($y_r = 1$), it must be connected to exactly one transformer

Deterministic Model - Constraints (2/3)

4. Demand satisfaction:

$$\sum_{l \in L} f_{lr} = q_r \cdot y_r \quad \forall r \in R$$

Meaning: If $y_r = 1$ (demand satisfied), then total inflow to r must equal demand q_r

5. Single transformer assignment:

$$\sum_{l \in L} u_{rl} = y_r \quad \forall r \in R$$

Meaning: If demand r is served ($y_r = 1$), it must be connected to exactly one transformer

6. Link flow to transformer assignment:

$$f_{lr} \leq c_{lr} \cdot u_{rl} \quad \forall r \in R, l \in L$$

Meaning: Flow from l to r only possible if $u_{rl} = 1$

Deterministic Model - Constraints (3/3)

7. Budget constraint:

$$\sum_{i \in N} \sum_{j \in N} b_{ij} \cdot z_{ij} \leq B$$

Meaning: Total cost of building arcs cannot exceed budget

Deterministic Model - Constraints (3/3)

7. Budget constraint:

$$\sum_{i \in N} \sum_{j \in N} b_{ij} \cdot z_{ij} \leq B$$

Meaning: Total cost of building arcs cannot exceed budget

8. Arc utilization (optional strengthening):

$$f_{ij} \geq \epsilon \cdot z_{ij} \quad \forall i, j \in N$$

Meaning: If we build an arc ($z_{ij} = 1$), we should use it (avoid building unused arcs)

Alternative formulation to ensure built arcs carry flow:

$$\sum_{i,j} f_{ij} \geq \sum_{i,j} z_{ij}$$

Complete Deterministic Model

$$\max \quad \sum_{r \in R} q_r \cdot y_r$$

$$\text{s.t.} \quad f_{ij} \leq z_{ij} \cdot c_{ij} \qquad \forall i,j \in N$$

$$\sum_{i \in H \cup M} f_{im} = \sum_{j \in M \cup L} f_{mj} \qquad \forall m \in M$$

$$\sum_{i \in M \cup L} f_{il} = \sum_{j \in L \cup R} f_{lj} \qquad \forall l \in L$$

$$\sum_{l \in L} f_{lr} = q_r \cdot y_r \qquad \forall r \in R$$

$$\sum_{l \in L} u_{rl} = y_r \qquad \forall r \in R$$

$$f_{lr} \leq c_{lr} \cdot u_{rl} \qquad \forall r \in R, l \in L$$

$$\sum_{i,j \in N} b_{ij} \cdot z_{ij} \leq B$$

$$f_{ij} \geq 0, \quad z_{ij}, y_r, u_{rl} \in \{0,1\}$$

Thank you very much for your attention!

Domande?

alessio.sclafani@unipa.it