DA 507 - Modeling and Optimization

Lecture 4
Fundamental Problems in Operations Research

Birol Yüceoğlu

Migros T.A.Ş.

December 7, 2021

Knapsack Problem (Capacity Budgeting)

Given a set of $N = \{1, \ldots, n\}$ objects, where object j has profit $p_j > 0$ and weight $w_j > 0$, and a container (knapsack) of capacity W, we want to select a set of objects to be inserted in the container so that:

- the total weight does not exceed capacity,
- the profit of the selected objects is maximum.

Knapsack Problem (Capacity Budgeting)

Given a set of $N = \{1, \ldots, n\}$ objects, where object j has profit $p_j > 0$ and weight $w_j > 0$, and a container (knapsack) of capacity W, we want to select a set of objects to be inserted in the container so that:

- the total weight does not exceed capacity,
- the profit of the selected objects is maximum.

$$x_j = \begin{cases} 1, & \text{if we select object } j, \\ 0, & \text{otherwise.} \end{cases}$$

Knapsack Problem (Capacity Budgeting)

Given a set of $N = \{1, \ldots, n\}$ objects, where object j has profit $p_j > 0$ and weight $w_j > 0$, and a container (knapsack) of capacity W, we want to select a set of objects to be inserted in the container so that:

- the total weight does not exceed capacity,
- the profit of the selected objects is maximum.

$$x_j = \begin{cases} 1, & \text{if we select object } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{maximize} \sum_{j \in N} x_j p_j, \tag{1}$$

$$\text{subject to} \sum_{j \in N} w_j x_j \leq W, \qquad \text{(capacity)} \tag{2}$$

$$x_j \in \{0, 1\}, \qquad \forall j \in N. \tag{3}$$

Knapsack Problem

- The knapsack problem consists of a single constraint (in addition to the domain constraints.
- However, it is a difficult problem to solve theoretically.
- The problem appears in the subproblem for several other problems.
- The problem has other variants such as multiple knapsack or change-making problem.
  - In the multiple knapsack problem, we are given a set of knapsacks, each having a capacity, and we decide in which knapsack we place the items in addition to item selection.
  - In the change-making problem, we are given a set of items with a limited number. We want to fulfill the exact capacity with the least number of items.

Traveling Salesperson Problem (TSP)

In the traveling salesperson problem (TSP), we are given a set of $V = \{1, \ldots, n\}$ cities and distances between the cities. This is usually represented by a directed graph $G = (V, A)$. Note that in the undirected version distances between the cities are the same in the opposite direction (i.e. $d_{ij} = d_{ji}$). We want to find a tour that visits every city exactly once (Hamiltonian tour) with a minimum cost.
The problem appears when we want to distribute a set of goods to a set of locations.

Traveling Salesperson Problem (TSP)

In the traveling salesperson problem (TSP), we are given a set of $V = \{1, \ldots, n\}$ cities and distances between the cities. This is usually represented by a directed graph $G = (V, A)$. Note that in the undirected version distances between the cities are the same in the opposite direction (i.e. $d_{ij} = d_{ji}$). We want to find a tour that visits every city exactly once (Hamiltonian tour) with a minimum cost.
The problem appears when we want to distribute a set of goods to a set of locations.

$$x_{ij} = \begin{cases} 1, & \text{the path goes from city } i \text{ to city } j, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Traveling Salesperson Problem (TSP)

In the traveling salesperson problem (TSP), we are given a set of $V = \{1, \ldots, n\}$ cities and distances between the cities. This is usually represented by a directed graph $G = (V, A)$. Note that in the undirected version distances between the cities are the same in the opposite direction (i.e. $d_{ij} = d_{ji}$). We want to find a tour that visits every city exactly once (Hamiltonian tour) with a minimum cost.
The problem appears when we want to distribute a set of goods to a set of locations.

$$x_{ij} = \begin{cases} 1, & \text{the path goes from city } i \text{ to city } j, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

$$x_{ij} = \begin{cases} 1, & \text{if arc } (i,j) \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Traveling Salesperson Problem (TSP)

$$\textit{minimize} \sum_{(i,j)\in A} c_{ij}x_{ij}, \tag{6}$$

$$\tag{7}$$

$$\text{subject to} \sum_{i\in V} x_{ij} = 1, \qquad \forall j \in V, (\text{incoming}) \tag{8}$$

$$\sum_{j\in V} x_{ij} = 1, \qquad \forall i \in V, (\text{outgoing}) \tag{9}$$

$$x_{ij} \in \{0,1\}, \qquad \forall (i,j) \in A, (\text{domain}) \tag{10}$$

$$\sum_{(i,j)\in A: i\in S, j\in V\setminus S} x_{ij} \geq 2, \qquad \forall S \subset V, (\text{subtour}). \tag{11}$$

Traveling Salesperson Problem

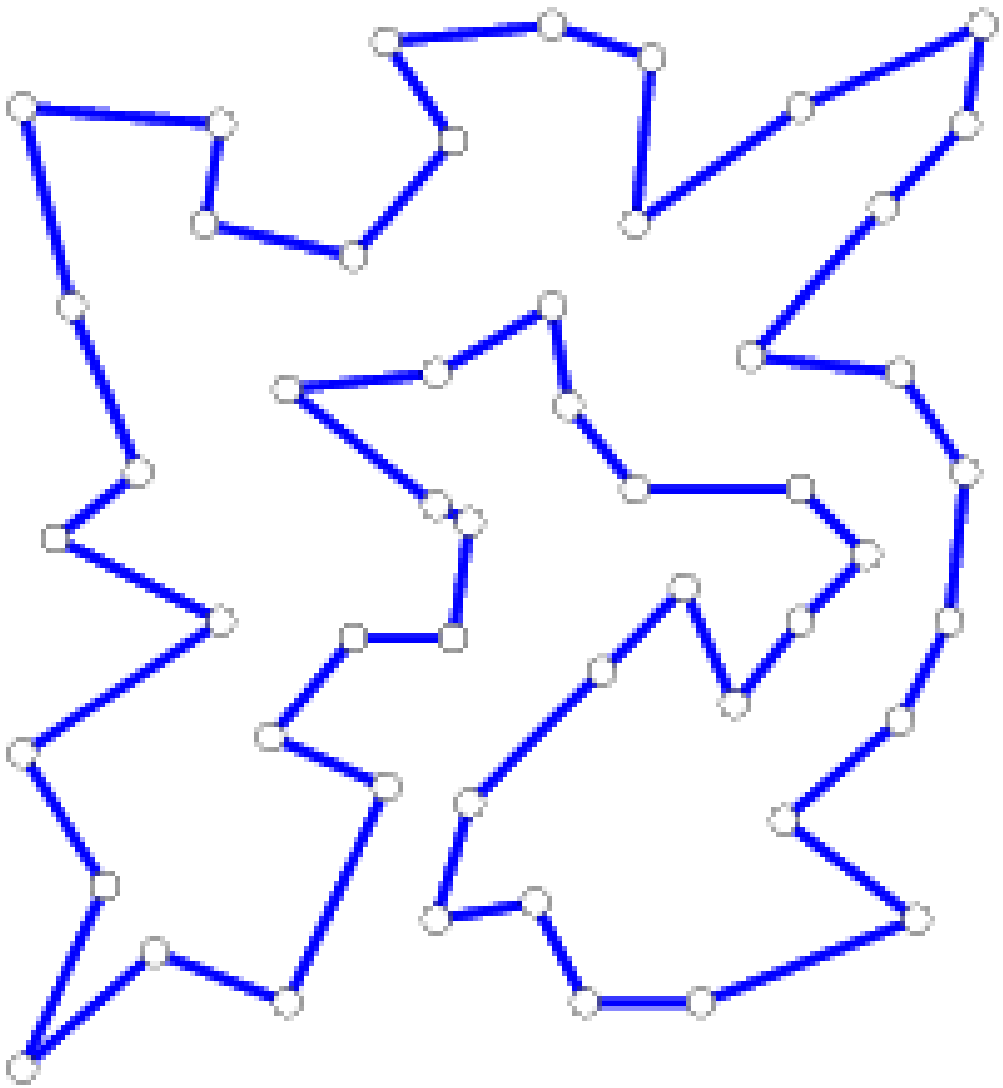
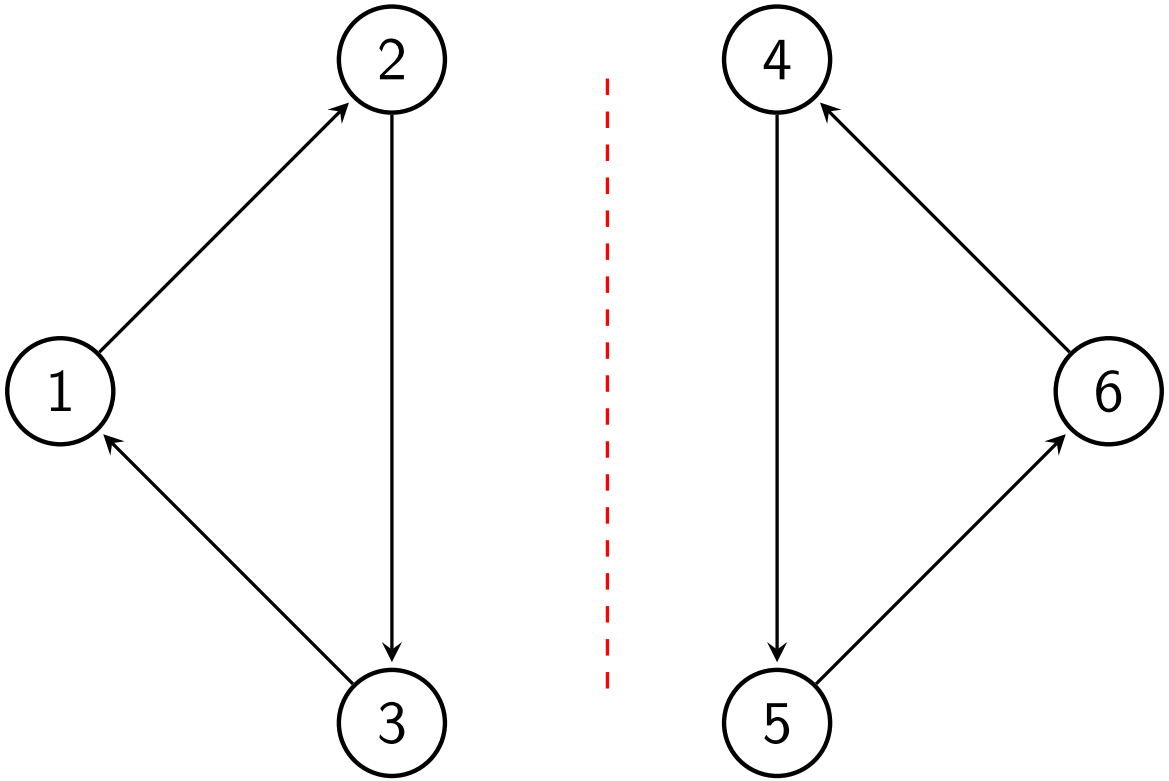


Figure: http://lemon.cs.elte.hu/pub/doc/1.3.1/a00618.html

Subtour Elimination Constraints

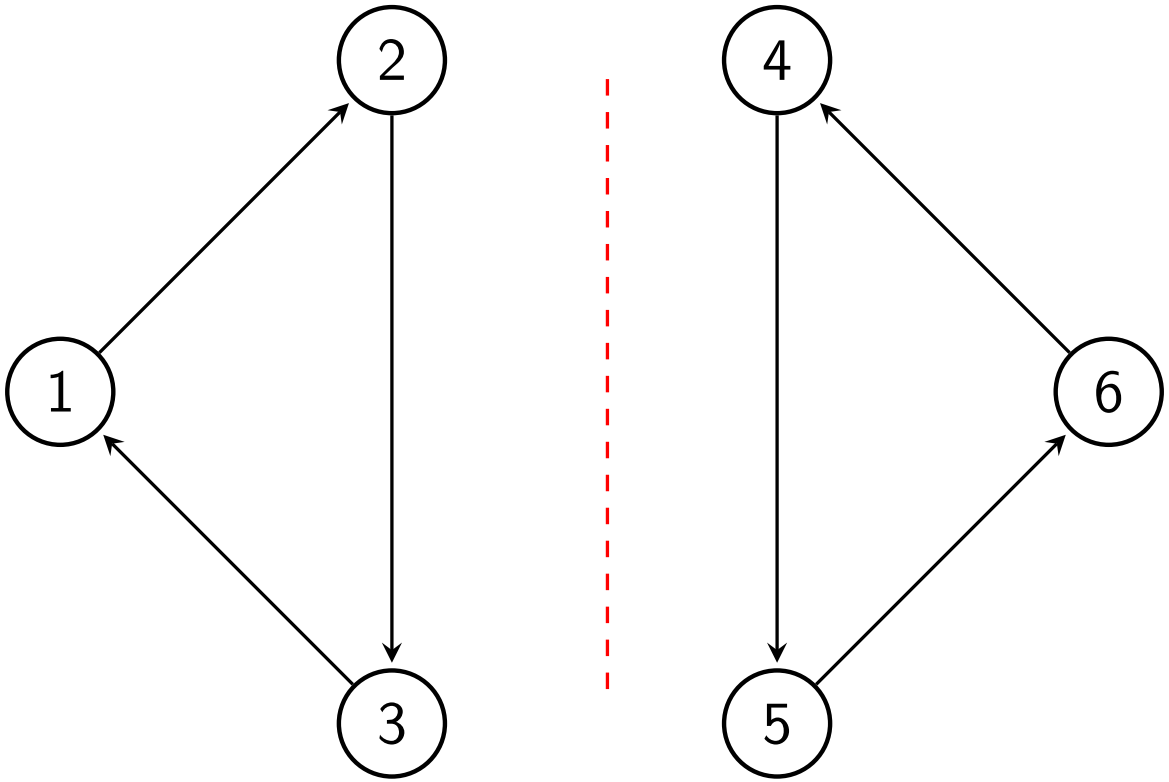


Let $S = \{1, 2, 3\}$, then we have the following subtour inequality that is violated by this solution.

$$x_{14} + x_{15} + x_{16} + x_{24} + x_{25} + x_{26} + x_{34} + x_{35} + x_{36} +$$

$$x_{41} + x_{51} + x_{61} + x_{42} + x_{52} + x_{62} + x_{43} + x_{53} + x_{63} \geq 2$$

Subtour Elimination Constraints



An alternative violated inequality based on forbidding directed cycles is written as:

$$x_{12} + x_{23} + x_{31} \leq 2$$

Subtour Elimination Constraints

- What happens when $S = \{4, 5, 6\}$?

Subtour Elimination Constraints

- What happens when $S = \{4, 5, 6\}$?
  - There should be an arc leaving S, i.e.
    $x_{41} + x_{51} + x_{61} + x_{42} + x_{52} + x_{62} + x_{43} + x_{53} + x_{63} \geq 1$
  - There should be an arc entering S, i.e.
    $x_{14} + x_{15} + x_{16} + x_{24} + x_{25} + x_{26} + x_{34} + x_{35} + x_{36} \geq 1$
- The subtour elimination constraints has to be written for every $S \subset V$. In other words we have to write $O(2^n)$ constraints.
- Usually, these constraints are omitted from the formulation. Whenever there is an integer solution in the branch-and-bound tree, we check whether a subtour elimination constraint is violated. If there is a violated constraint it is added to the formulation and we carry on with the branch-and-bound.

Optimization in Clustering

- In clustering we approximate or group a large set by a smaller set of representatives.
- Cluster analysis involves the grouping of data entities in a way that maximizes the homogeneity of entities within a group and, at the same time the heterogeneity of entities between groups.
- It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics.
- We are going to study the K-Center and the K-Median problems using integer programming.

Setup & Notation

- Entities (observations in data science) are numbered from 1 to n, i.e. $i \in N = \{1, \ldots, n\}$.
- Suppose there are M attributes (features) that describe each entity as a_{ik} (numeric value) corresponding to your features.
- We define dissimilarity between i and j as the distance, such as Euclidean distance. You can use w_i as the importance/weight of entity i. We do not consider weights in this example.
- We assign at most k cluster centers.
- Each point is assigned to exactly one cluster center.
- A point can only be assigned to a selected cluster center.
- K-Median: Minimize the total distance for assignment.
- K-Center: Minimize the maximum distance for assignment.

Decision Variables

$$y_j = \begin{cases} 1, & \text{if observation } j \text{ is a cluster center,} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

$$x_{ij} = \begin{cases} 1, & \text{if observation } i \text{ is assigned to cluster center } j, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

K-Median Problem

$$\text{minimize} \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ij}, \tag{14}$$

$$\text{subject to} \sum_{j \in N} y_j = k, \qquad \text{(centers)} \tag{15}$$

$$\sum_{j \in N} x_{ij} = 1, \qquad \forall i \in N \text{ (assignment)} \tag{16}$$

$$\sum_{i \in N} x_{ij} \leq M y_j, \qquad \forall j \in N \text{ (center assignment)} \tag{17}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in N, \tag{18}$$

$$y_j \in \{0, 1\}, \qquad \forall j \in N. \tag{19}$$

K-Center Problem

r: the maximum distance between a cluster center and an observation.

$$\textit{minimize } r, \tag{20}$$

$$\text{subject to } \sum_{j \in N} y_j = k, \qquad \text{(centers)} \tag{21}$$

$$\sum_{j \in N} x_{ij} = 1, \qquad \forall i \in N \text{(assignment)} \tag{22}$$

$$\sum_{i \in N} x_{ij} \leq M y_j, \qquad \forall j \in N \text{ (center assignment)} \tag{23}$$

$$r \geq d_{ij} x_{ij}, \qquad \forall i, j \in N \text{ (distance)} \tag{24}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in N, \tag{25}$$

$$y_j \in \{0, 1\}, \qquad \forall j \in N. \tag{26}$$

Cutting Stock Problem

- A manufacturer of metal sheets produces rolls of standard fixed width w and of length of 10 meters.
- A large order is placed by a customer who needs sheets of width w and varying lengths such as 1.5, 2.5, 3.0, and 4.0. Demand for each is 10, 12, 15 and 7 sheets, respectively.
- The manufacturer would like to cut the standard rolls in such a way as to satisfy the order and to minimize the waste. Since scrap pieces are useless to the manufacturer, the objective is to minimize the number of rolls needed to satisfy the order (or total scrapped metal sheets).

Cutting Stock Problem

Given a standard sheet of 10 meters, there are many ways of cutting it. We consider the following five patterns, where each number corresponds to the number of sheets of length 1.5, 2.5, 3.0, and 4.0.

$$a_1 = \begin{bmatrix} 3 \\ 2 \\ 0 \\ 0 \end{bmatrix}, a_2 = \begin{bmatrix} 0 \\ 4 \\ 0 \\ 0 \end{bmatrix}, a_3 = \begin{bmatrix} 0 \\ 0 \\ 3 \\ 0 \end{bmatrix}, a_4 = \begin{bmatrix} 0 \\ 0 \\ 2 \\ 1 \end{bmatrix}, a_5 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Note that when the patterns are not given (i.e. any pattern respecting the length of the sheet is possible) the problem becomes notoriously difficult.

Cutting Stock Problem

$x_1 =$ the number of sheets cut using pattern a_1. $x_2, \ldots, x_5$ are defined analogously.

Cutting Stock Problem

x_1 = the number of sheets cut using pattern a_1. $x_2, \ldots, x_5$ are defined analogously.

$$\textit{minimize} \sum_{i=1}^{5} x_i, \tag{27}$$

$$\begin{aligned}
\text{subject to } & 3x_1 + x_5 \geq 10, && \text{(demand (1.5))} && (28)\\
& 2x_1 + 4x_2 + x_5 \geq 12, && \text{(demand (2.5))} && (29)\\
& 3x_3 + 2x_4 \geq 15, && \text{(demand (3.0))} && (30)\\
& x_4 + x_5 \geq 7, && \text{(demand (4.0))} && (31)\\
& x_i \geq 0, x_i \in \mathbb{Z}, && \forall i \in \{1, \ldots, 5\} \text{(domain)} && (32)
\end{aligned}$$

Thanks

Heuristics

Please read about decision trees

Job Scheduling Problem

Given a set of $N = \{1, \ldots, n\}$ job, where job j has processing time $p_j > 0$, and a set of $M = \{1, \ldots, m\}$ identical machines, we want to assign the jobs to the machines so that:

- each job is assigned to exactly one machine,
- each machine processes one job at a time,
- the makespan (i.e. the maximum uptime of all the machines) is minimum.

Job Scheduling Problem

$$x_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to machine } i, \\ 0, & \text{otherwise.} \end{cases}$$

$z =$ completion time of the last job (makespan).

Job Scheduling Problem

$$x_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to machine } i, \\ 0, & \text{otherwise.} \end{cases}$$

$z =$ completion time of the last job (makespan).

$$\textit{minimize } z, \tag{33}$$

$$\text{subject to } z \geq \sum_{j \in N} p_j x_{ij}, \qquad \forall i \in M(\text{makespan}) \tag{34}$$

$$\sum_{i \in M} x_{ij} = 1, \qquad \forall j \in N(\text{assignment}) \tag{35}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i \in M, \forall j \in N. \tag{36}$$

Assignment Problem

Given a matrix the cost to assign row i to column j the assignment problem consists in assigning the rows to the columns so that:

- each row is assigned to exactly one column,
- each columns is assigned to exactly one row,
- the total assignment cost is minimum.

Assignment Problem

$$x_{ij} = \begin{cases} 1, & \text{if row } i \text{ is assigned to column } j, \\ 0, & \text{otherwise.} \end{cases}$$

Assignment Problem

$$x_{ij} = \begin{cases} 1, & \text{if row } i \text{ is assigned to column } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\textit{minimize} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}, \tag{37}$$

$$\text{subject to} \sum_{j \in N} x_{ij} = 1, \qquad \forall i \in N (\text{assignment}) \tag{38}$$

$$\sum_{i \in N} x_{ij} = 1, \qquad \forall j \in N (\text{assignment}) \tag{39}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i \in N, \forall j \in N. \tag{40}$$

Set Covering Problem

Consider a binary matrix A with m rows ($M = \{1, \ldots, m\}$) and n columns ($N = \{1, \ldots, n\}$), where column j has cost c_j. If $a_{ij} = 1$, we say that column j covers row i. The set covering problem consists in selecting a set of columns so that:

- each row is covered by **at least** one column,
- the total cost of selection is minimum.

Examples

Consider you have to locate ambulances in the eight district of İstanbul. The traveling times between neighborhoods are given below.

	A	K	B	K	B	B	Ü	T
Ataşehir	0	15	30	20	50	40	20	30
Kadıköy	15	0	20	25	40	25	15	30
Beşiktaş	30	20	0	45	25	15	20	55
Kartal	20	25	45	0	75	60	45	15
Bakırköy	50	40	25	75	0	20	45	90
Beyoğlu	40	25	15	60	20	0	40	75
Üsküdar	20	15	20	45	45	40	40	80
Tuzla	30	30	55	15	90	75	80	0

We want to minimize the number of ambulances, while making sure that each district is covered. An ambulance can only serve a district if the traveling distance is less than 25 minutes.

Set Covering Problem

$$x_j = \begin{cases} 1, & \text{if column } j \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

Set Covering Problem

$$x_j = \begin{cases} 1, & \text{if column } j \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

$$\textit{minimize} \sum_{j \in N} c_j x_j, \tag{41}$$

$$\text{subject to} \sum_{j \in N} a_{ij} x_{ij} \geq 1, \qquad \forall i \in M (\text{coverage}) \tag{42}$$

$$x_j \in \{0, 1\}, \qquad \forall j \in N. \tag{43}$$

Set Partitioning Problem

Consider a binary matrix A with m rows and n columns, where column j has cost c_j. If $a_{ij} = 1$, we say that column j covers row i. The set partitioning problem consists in selecting a set of columns so that:

- each row is covered **exactly** by one column,
- the total cost of selection is minimum.

Set Partitioning Problem

$$x_j = \begin{cases} 1, & \text{if column } j \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

Set Partitioning Problem

$$x_j = \begin{cases} 1, & \text{if column } j \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

$$\textit{minimize} \sum_{j \in N} c_j x_j, \tag{44}$$

$$\text{subject to} \sum_{j \in N} a_{ij} x_{ij} = 1, \qquad \forall i \in M (\text{coverage}) \tag{45}$$

$$x_j \in \{0, 1\}, \qquad \forall j \in N. \tag{46}$$

Set Packing Problem

Consider a binary matrix A with m rows and n columns, where column j has profit p_j. If $a_{ij} = 1$, we say that column j covers row i. The set packing problem consists in selecting a set of columns so that:

- each row is covered by **at most** one column,
- the total **profit** of selection is maximum.

Set Packing Problem

$$x_j = \begin{cases} 1, & \text{if column } j \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

Set Packing Problem

$$x_j = \begin{cases} 1, & \text{if column } j \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

$$\textit{maximize} \sum_{j \in N} c_j x_j, \tag{47}$$

$$\text{subject to} \sum_{j \in N} a_{ij} x_{ij} \leq 1, \qquad \forall i \in M (\text{coverage}) \tag{48}$$

$$x_j \in \{0, 1\}, \qquad \forall j \in N. \tag{49}$$

Examples

Consider you have to locate ambulances in the eight district of İstanbul. The traveling times between neighborhoods are given below.

	A	K	B	K	B	B	Ü	T
Ataşehir	0	15	30	20	50	40	20	30
Kadıköy	15	0	20	25	40	25	15	30
Beşiktaş	30	20	0	45	25	15	20	55
Kartal	20	25	45	0	75	60	45	15
Bakırköy	50	40	25	75	0	20	45	90
Beyoğlu	40	25	15	60	20	0	40	75
Üsküdar	20	15	20	45	45	40	40	80
Tuzla	30	30	55	15	90	75	80	0

We want to minimize the number of ambulances, while making sure that each district is covered. An ambulance can only serve a district if the traveling distance is less than 25 minutes.

Examples

Below, the distance matrix is turned into the coverage matrix. $a_{ij} = 1$ if distance between i and j is less than 25 minutes.

	A	K	B	K	B	B	Ü	T
Ataşehir	1	1	0	1	0	0	1	0
Kadıköy	1	1	1	1	0	1	1	0
Beşiktaş	0	1	1	0	1	1	1	0
Kartal	1	1	0	1	0	0	0	1
Bakırköy	0	0	0	0	1	1	0	0
Beyoğlu	0	1	1	0	1	1	0	0
Üsküdar	1	1	1	0	0	0	1	0
Tuzla	0	0	1	1	0	0	0	1

We can formulate a set covering problem based on this data..

Examples

The set covering/partitioning/packing problems are essential as they allow us to model more difficult problems in a concise way.
Consider you have to replenish the cash inventory in your ATM machines in İstanbul. In the following matrix you already have the cities that a truck may cover (so you already modeled this aspect of your problem in some way).

$$
\begin{array}{c} \\ \text{Ataşehir} \\ \text{Kadıköy} \\ \text{Beşiktaş} \\ \text{Kartal} \\ \text{Bakırköy} \\ \text{Beyoğlu} \\ \text{Üsküdar} \\ \text{Tuzla} \end{array}
\begin{array}{c}
\begin{array}{cccccccc} \multicolumn{8}{c}{\textit{TruckRoute}} \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\
\begin{pmatrix}
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{pmatrix}
\end{array}
$$

Examples

The set covering/partitioning/packing problems are essential as they allow us to model more difficult problems in a concise way.
Consider you have to replenish the cash inventory in your ATM machines in İstanbul. In the following matrix you already have the cities that a truck may cover (so you already modeled this aspect of your problem in some way).

$$
\begin{array}{l} \\ \text{Ataşehir} \\ \text{Kadıköy} \\ \text{Beşiktaş} \\ \text{Kartal} \\ \text{Bakırköy} \\ \text{Beyoğlu} \\ \text{Üsküdar} \\ \text{Tuzla} \end{array}
\begin{array}{c}
\begin{array}{cccccccc} \multicolumn{8}{c}{\textit{TruckRoute}} \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\
\left(\begin{array}{cccccccc}
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{array} \right)
\end{array}
$$

Examples

	TruckRoute							
	1	2	3	4	5	6	7	8
Ataşehir	1	0	1	0	0	0	1	1
Kadıköy	1	0	0	1	0	1	0	0
Beşiktaş	0	1	0	1	0	1	0	1
Kartal	0	0	0	0	1	0	1	0
Bakırköy	0	1	1	0	0	0	0	0
Beyoğlu	0	0	1	0	0	0	0	1
Üsküdar	0	0	0	1	1	1	1	0
Tuzla	0	0	0	0	1	0	0	0

- Each location is covered by at least one vehicle (Covering)
- Each location is covered by exactly one vehicle (Partitioning)
- Each location is covered by at most one vehicle (Packing)

Examples

$$
\begin{array}{c} \\ \text{Customer 1} \\ \text{Customer 2} \\ \ldots \end{array}
\begin{array}{c}
\begin{array}{cccccccc} \multicolumn{8}{c}{\textit{Offers}} \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\
\left(\begin{array}{cccccccc} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array} \right)
\end{array}
$$

- As the CRM unit of a company you have a set of different offers that you can propose to your customers. Each offer is suitable to a set of customers and vice versa. You want to minimize the cost of offers while giving each customer at least one offer.

Examples

$$
\begin{array}{l}
 \\
\text{Customer 1} \\
\text{Customer 2} \\
\ldots
\end{array}
\begin{array}{c}
\begin{array}{cccccccc}
\multicolumn{8}{c}{\textit{Offers}} \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8
\end{array} \\
\left(\begin{array}{cccccccc}
1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 1
\end{array}\right)
\end{array}
$$

- As the CRM unit of a company you have a set of different offers that you can propose to your customers. Each offer is suitable to a set of customers and vice versa. You want to minimize the cost of offers while giving each customer at least one offer.
- Same problem but you make exactly one offer to each customer

Examples

$$
\begin{array}{c} \\ \text{Customer 1} \\ \text{Customer 2} \\ \ldots \end{array}
\begin{array}{c}
\begin{array}{cccccccc} \multicolumn{8}{c}{\textit{Offers}} \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\
\left(\begin{array}{cccccccc} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array} \right)
\end{array}
$$

- As the CRM unit of a company you have a set of different offers that you can propose to your customers. Each offer is suitable to a set of customers and vice versa. You want to minimize the cost of offers while giving each customer at least one offer.
- Same problem but you make exactly one offer to each customer
- You want to maximize your revenues from the offers but you can at most give one offer to each customer.

Facility Location Problem

Given a set of M customers to be served, where i^{th} customer has demand d_i and a set of N possible locations for facilities with investment cost $f_j > 0$ and production capacity $b_j > 0$, and transportaton cost c_{ij} from plant j to customer i, we want to decide which facilities to build so that each customer is completely served by one and only one facility, each facility (if activated) serves a subset of customers whose total demand does not exceed the capacity of the facility and the overall cost is minimum.

Facility Location Problem

Given a set of M customers to be served, where i^{th} customer has demand d_i and a set of N possible locations for facilities with investment cost $f_j > 0$ and production capacity $b_j > 0$, and transportaton cost c_{ij} from plant j to customer i, we want to decide which facilities to build so that each customer is completely served by one and only one facility, each facility (if activated) serves a subset of customers whose total demand does not exceed the capacity of the facility and the overall cost is minimum.

$$y_j = \begin{cases} 1, & \text{if facility } j \text{ is activated,} \\ 0, & \text{otherwise.} \end{cases}$$

Facility Location Problem

Given a set of M customers to be served, where i^{th} customer has demand d_i and a set of N possible locations for facilities with investment cost $f_j > 0$ and production capacity $b_j > 0$, and transportaton cost c_{ij} from plant j to customer i, we want to decide which facilities to build so that each customer is completely served by one and only one facility, each facility (if activated) serves a subset of customers whose total demand does not exceed the capacity of the facility and the overall cost is minimum.

$$y_j = \begin{cases} 1, & \text{if facility } j \text{ is activated,} \\ 0, & \text{otherwise.} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{if customer } i \text{ is served by facility } j, \\ 0, & \text{otherwise.} \end{cases}$$

Facility Location Problem

$$\text{minimize} \sum_{j \in N} f_j y_j + \sum_{j \in N} \sum_{i \in M} c_{ij} x_{ij}, \tag{50}$$

$$\text{subject to} \sum_{j \in N} x_{ij} = 1, \qquad \forall i \in M (\text{assignment}) \tag{51}$$

$$\sum_{i \in M} d_i x_{ij} \leq b_j y_j, \qquad \forall j \in N (\text{capacity}) \tag{52}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall j \in N, \forall i \in M, \tag{53}$$

$$y_j \in \{0, 1\}, \qquad \forall j \in N. \tag{54}$$

Shift Assignment

You want to allocate your employees to the daily shifts based on the daily requirements. Each employee has to work for five consecutive days (regardless of the starting day within the week) and take the two subsequent days off. Below you are given the daily requirements for employees in the shifts.

Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Number	17	13	15	19	14	16	11

Construct a model so that each shift is covered by at least the required capacity and the total number of employees is minimum.

Shift Assignment

You want to allocate your employees to the daily shifts based on the daily requirements. Each employee has to work for five consecutive days (regardless of the starting day within the week) and take the two subsequent days off. Below you are given the daily requirements for employees in the shifts.

Day	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Number	17	13	15	19	14	16	11

Construct a model so that each shift is covered by at least the required capacity and the total number of employees is minimum.
Mon = the number of employees that start their weekly shift on Monday (and work until Friday). Tue, Wed, Thu, Fri, Sat, Sun are defined similarly.

Shift Assignment

$$\text{minimize } Mon + Tue + Wed + Thu + Fri + Sat + Sun, \tag{55}$$

$$\text{subject to } Mon + Thu + Fri + Sat + Sun \geq 17, \quad \text{(Mon)} \tag{56}$$

$$Mon + Tue + Fri + Sat + Sun \geq 13, \quad \text{(Tue)} \tag{57}$$

$$Mon + Tue + Wed + Sat + Sun \geq 15, \quad \text{(Wed)} \tag{58}$$

$$Mon + Tue + Wed + Thu + Sun \geq 19, \quad \text{(Thu)} \tag{59}$$

$$Mon + Tue + Wed + Thu + Sun \geq 14, \quad \text{(Fir)} \tag{60}$$

$$Tue + Wed + Thu + Fri + Sat \geq 16, \quad \text{(Sat)} \tag{61}$$

$$Wed + Thu + Fri + Sat + Sun \geq 11, \quad \text{(Sun)} \tag{62}$$

$$Mon, \ldots, Sun \geq 0 \text{ and integer.} \tag{63}$$

References

M. Fischetti, Introduction to Mathematical Optimization, Kindle Direct Publishing, 2019.