## Combinatorial Optimization Problems

A combinatorial optimization problem is one in which the number of possible solutions (combinatorics) is too large to perform brute force search for the general problem. Try to solve by hand the following.

### 1. Assignment Problem

- Assign taxis to trip requests, (or workers to jobs, students to projects, …, etc) so that total cost/time is **minimized**.
- Each trip request / job / project must be assigned.
- Simple case -> 1:1 matching. Each taxi/worked/student can be assigned at most one item.
- **Extensions**:
  - Add capacity constraints → becomes Generalized Assignment Problem.
  - Add multiple subsets per assignee, multiple assignees per subset, different costs per subset (i.e. price of subset is not simply sum of price of individual items) → becomes Combinatorial Auction Problem

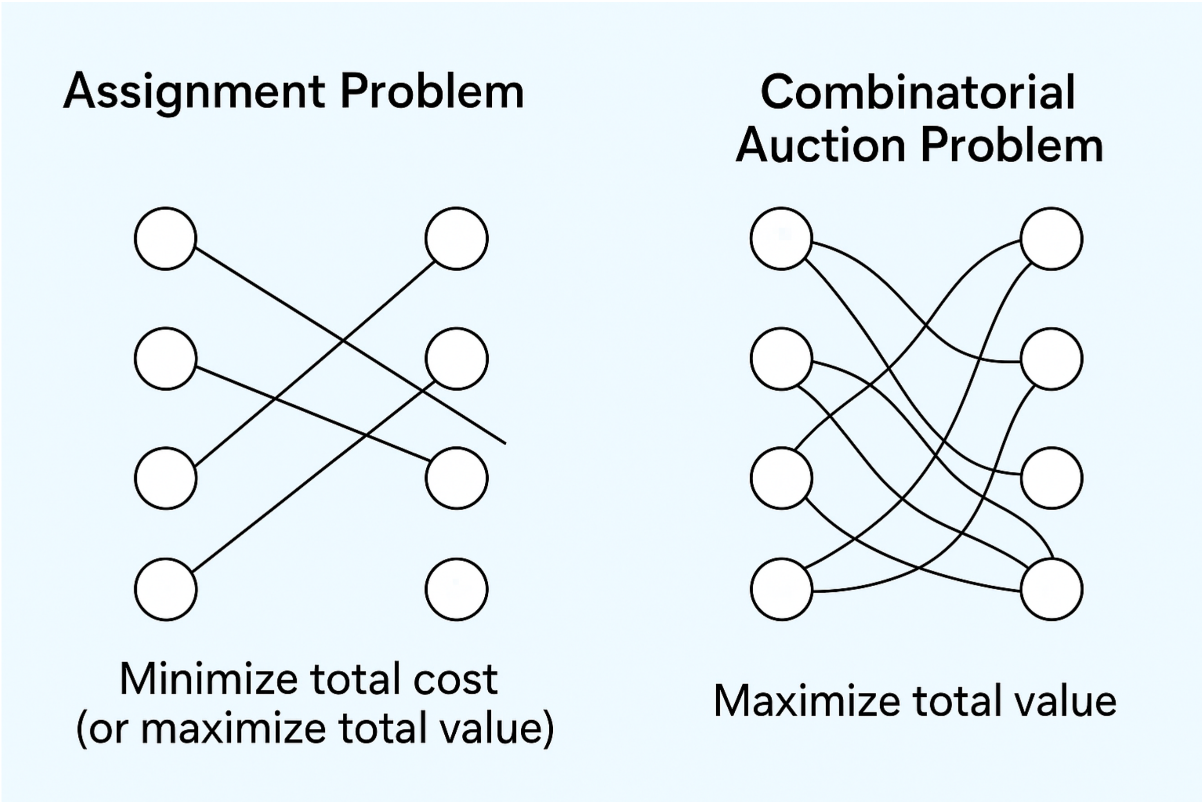
| |  |  |  |
|---|---|---|---|
|  | 11 | 7 | 10 |
|  | 12 | 7 | 11 |
|  | 8 | 5 | 6 |
|  | 12 | 9 | 10 |

*Figure 1: Taxis costs for different taxis for different trip requests*

Find assignment of taxis to customers (remember a taxi can only handle one customer) such that overall cost is minimized!

Combinatorial Auctions: Multiple bidders, multiple items, bidders can place bids of different value on different combinations.

https://www.keelvar.com/



## 2. Knapsack Problem

- Choose items with weights and values to **maximize** value under a weight constraint.
- **Extensions**: Multi-knapsack, Multi-dimensional knapsack, fractional knapsack.

---

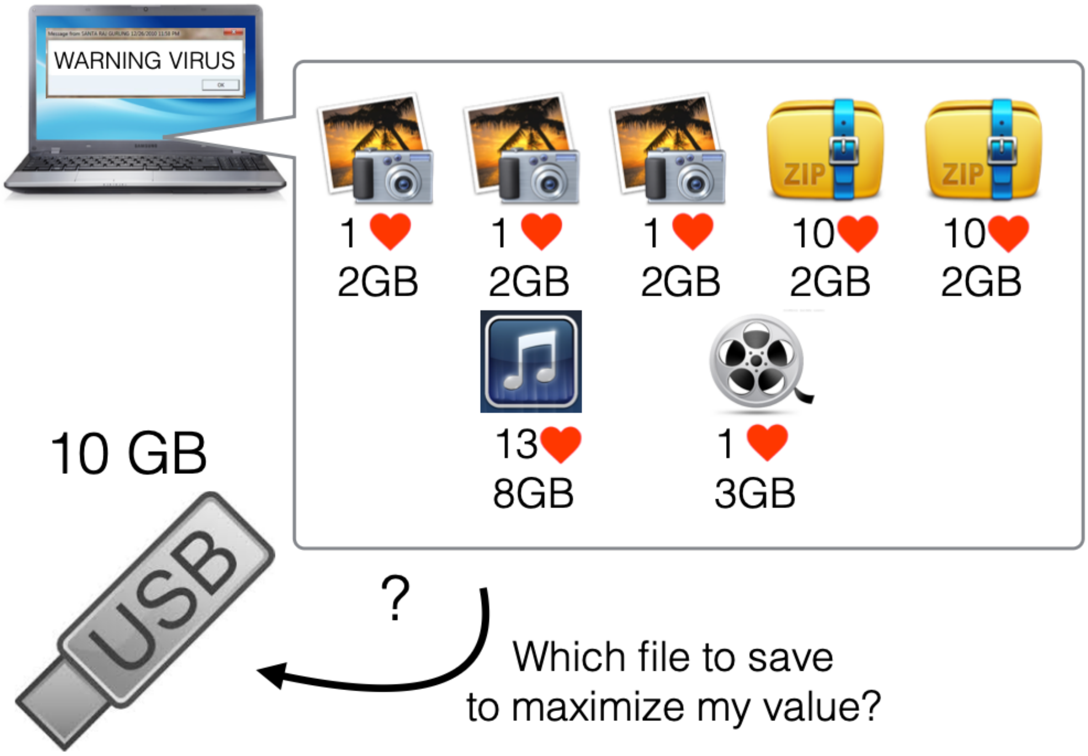



*Figure 2: 0-1 Knapsack github.com/pschaus /linfo2266*

Solve where **at most** one of each item can be added.

| File | Size GB | Value |
|---|---|---|
| **1** | 12 | 24 |
| **2** | 7 | 13 |
| **3** | 6 | 12 |
| **4** | 5 | 10 |
| **5** | 4 | 8 |
| **6** | 3 | 7 |

*Figure 3: Harder 0-1 Knapsack with capacity 16GB*

1. Solve same again to maximize value according to Figure 3 with capacity 16.
2. Solve where size is 35GB and can take multiple of each item (again we don't have to take every item so we can take multilpe of some items and none of others).
3. Solve where at most one of each item can be added but we have two knapsacks, each with size 10 GB.

---

**3. Traveling Salesperson Problem (TSP)**

- Find the route that visit each city once and returns home where the route is minimised.
- **Extensions**: Vehicle routing problem, time windows.

| | Cork | Fermoy | Cobh | Youghal | Mallow |
|---|---|---|---|---|---|
| Cork | - | 42 | 38 | 51 | 32 |
| Fermoy | 42 | - | 33 | 27 | 25 |
| Cobh | 38 | 33 | - | 39 | 53 |
| Youghal | 51 | 27 | 39 | - | 67 |
| Mallow | 32 | 25 | 53 | 67 | - |

---

## 4. Bin Packing

- Pack items of varying sizes into bins of fixed capacity such that we minimise the number of bins used.
- Examples: Delivery service wants to minimise the number of trucks needed to deliver all their parcels. Data centre has servers, pack VMs according to max resource usage. Shipping company has containers, want to minimise number of containers used. Backup of files/media onto storage devices.

  Glass cutting problem of $40 Billion company
  https://roadef.org/challenge/2018/en/sujet.php

- **Extensions**: Online vs offline versions.

Example 1: 1-D
Bin capacity = 10
Items = [2, 2, 2, 3, 5, 6]
(Total size = 20 ⇒ **Lower Bound** = 2 bins)

Example 2: 2-D
**Bin size**: 8 × 8
**Items**:
- 4 × 4
- 4 × 4
- 4 × 4
- 2 × 4
- 2 × 4
- 4 × 2
- 4 × 2

Some additional problems

---

## Set Cover (or Minimum Hitting Set)

- Cover all elements (say, students with class times) using as few sets (classrooms) as possible.
- **Extensions**: Weighted set cover, applications to facility location.

---

**Maximum Matching in Bipartite Graphs**

- Pair up students and projects, or doctors and patients, without overlap.
- **Extensions**: Stable matching, general graph matching.

---

**Minimum Spanning Tree (MST)**

- Connect cities with roads at minimum cost.
- **Extensions**: Steiner tree, network design.

---