

Knapsack Problem

PROJECT
Artificial Intelligence Laboratory

Carolina Crespi

January 28, 2026

1 Knapsack Problem – KP

The *Knapsack Problem* (KP) is one of the most well-known combinatorial optimization problems and is a classical example of an *NP-hard* problem. The problem consists of selecting a subset of items, each characterized by a weight and a value, in order to maximize the total value without exceeding a maximum available capacity. Formally, let a set of n items be given, where each item i is associated with:

- a weight $w_i > 0$,
- a value $v_i > 0$.

Let W denote the maximum capacity of the knapsack. The problem consists of determining a binary solution

$$x = (x_1, x_2, \dots, x_n), \quad x_i \in \{0, 1\},$$

that maximizes the objective function:

$$\max \sum_{i=1}^n v_i x_i$$

subject to the constraint:

$$\sum_{i=1}^n w_i x_i \leq W.$$

In the classical *0/1 Knapsack* problem, each item can be selected at most once.

2 Problem Instance

In this project, a single instance of the 0/1 Knapsack problem is considered and provided together with the assignment. The instance consists of five items, each with an assigned weight and value, and a maximum knapsack capacity W . For the considered instance, the optimal value of the objective function is known and can be used to verify the correctness of the obtained solutions.

3 Experimental Protocol

For the considered instance, the selected algorithm must be executed using the following stopping criteria:

- reaching the known optimal value of the objective function, **or**

- reaching a maximum number of 200 evaluations of the objective function.

For deterministic algorithms (Branch and Bound), a single execution of the algorithm is required.

For stochastic algorithms (Local Search, Genetic Algorithm, and Simulated Annealing), the algorithm must be executed for **5 independent runs**, using different initial or semi-random conditions.

The final report must include:

- the obtained value of the objective function;
- the identified solution (binary vector);
- verification of compliance with the capacity constraint;
- for stochastic algorithms, the average of the final values obtained over the 5 runs;
- a brief comment on the behavior of the adopted algorithm.

4 Algorithm to Implement

The candidate is required to choose and implement **one** of the following approaches for solving the Knapsack Problem:

1. **Branch and Bound**, including the definition of the decision tree and search space exploration strategies (depth-first, best-bound-first, or hybrid strategy);
2. **Local Search**, based on a neighborhood defined by a single bit flip and accepting only improving solutions;
3. **Genetic Algorithm (GA)** with binary encoding of the solution, selection (roulette wheel or tournament of size 2), one-point crossover, and mutation through bit flip;
4. **Simulated Annealing (SA)** with binary representation of the solution, neighborhood based on bit flip, high initial temperature, equilibrium condition with 5 moves per temperature value, and linear cooling.

The choice of the algorithm must be motivated in the final report, together with a description of the main design decisions adopted.

5 Final Notes

The final report must be written in L^AT_EX, with a minimum length of **4 pages** and a maximum length of **12 pages**. The inclusion of implemented code fragments is strongly discouraged, while the use of pseudocode and explanatory diagrams is strongly recommended.

The report must clearly and thoroughly illustrate:

- the developed algorithm and its main characteristics;
- the motivations behind the adopted design choices;
- any elements of novelty and originality introduced;
- a critical analysis of the results obtained on the assigned instance;
- personal considerations and conclusions on the developed project.

The final evaluation will be mainly based on:

- originality and complexity of the developed algorithm;
- quality of the obtained results;
- clarity and quality of presentation;
- completeness and clarity of the content.

Submission: the project (source code) and the final report (.pdf and .tex, including any figure and/or table files) must be sent by email to `carolina.crespi@unict.it` by **2:00 PM on February 16, 2026**.

The candidate is required to communicate the chosen algorithm by **4:00 PM on January 30, 2026**.