**Report 5 – Dynamic Programming, Kuba Czech and Adam Wilczyński**

1. Short description of **Knapsack Problem**:

Knapsack problem is problem of packing a backpack with items, that this backpack will have maximum value, but sum of weights of packed items in this backpack does not exceed capacity of the knapsack.

1. Short description of **Partition Problem**:

Partition problem is a problem of dividing a set of numbers into two groups, such that sum of the numbers in each group is the same.

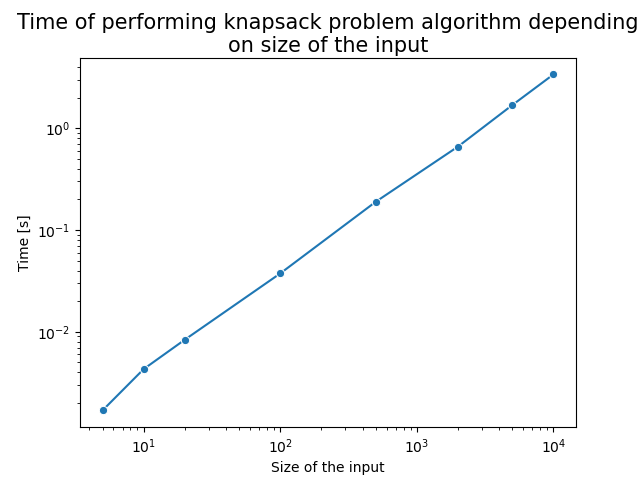
1. Why is Knapsack Problem **NP-Hard**:
2. **Complexity** of an algorithm solving Knapsack Problem:

To solve this problem using Dynamic Programming we construct a matrix of size (m+1) x (n+1), where m is capacity of knapsack and n is number of elements (size of an input), fill with data and at the end maximum value is located in the last cell of last line. Thus we need to perform (m+1)\*(n+1) operations, so complexity of this algorithm will be **O(m x n)** – pseudo polynomial.

1. **Time** of solving Knapsack Problem for different capacities and different number of elements:
2. Time of solving Knapsack Problem for different number of elements and fixed capacity (M=100):

|  |  |
| --- | --- |
| Nr of elements | Time [s] |
| 5 | 0.0016920 |
| 10 | 0.0043063 |
| 20 | 0.0083675 |
| 100 | 0.0374659 |
| 500 | 0.1892937 |
| 2 000 | 0.6593361 |
| 5 000 | 1.6867167 |
| 10 000 | 3.3741549 |

**Graph for this set of data:**

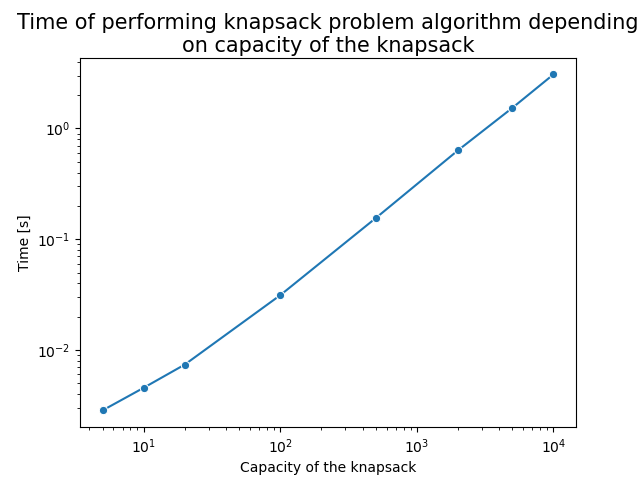


As we can see, for fixed capacity we have linear function depending on number of elements.

1. Time of solving Knapsack Problem for different capacities and fixed number of elements (N=100):

|  |  |
| --- | --- |
| Capacity | Time [s] |
| 5 | 0.0028433 |
| 10 | 0.0045503 |
| 20 | 0.0073751 |
| 100 | 0.0312110 |
| 500 | 0.1550366 |
| 2 000 | 0.6344636 |
| 5 000 | 1.5336568 |
| 10 000 | 3.0785759 |

**Graph for this set of data:**



As we can see, for fixed number of elements we have time of calculations depending on capacity of the knapsack

1. Time of solving Knapsack Problem for both different capacities and different number of elements (both are equal):

|  |  |
| --- | --- |
| Capacity/Nr of elements | Time [s] |
| 5 | 0.0002165 |
| 10 | 0.0007380 |
| 20 | 0.0020132 |
| 100 | 0.0325046 |
| 500 | 0.7784830 |
| 2 000 | 12.9957741 |
| 5 000 | 81.1961774 |
| 10 000 | 332.9520935 |

**Graph for this set of data:**

As we can see, when both capacity and number of elements change about the equal value, we have parabola.

1. Problem Transformation: