# Using Genetic Algorithm and Google OR-Tools to solve Knapsack Problem[*]

**An Khanh Vo**
19520007@gm.uit.edu.vn

May 2021

**Abstract**

In this report, we try to find the solution for Knapsack Problem using various algorithm such as Genetic Algorithm and OR-Tools. We have implemented these algorithms and compare their performance in terms of time and quality of the solution. This experiment we run entirely on Google Colab.

## 1 Genetic Algorithm

In GA algorithm, we set the maximum execution time to 300 seconds. We choose the population size 10 times if the number of items is less than 5000. Experimentally, we feel that the algorithm will be difficult to give good results if choosing a smaller population size. For larger tests like 5000 or 10000, we choose polulation sizes 10000 and 5000 respectively to avoid memory overflow. We do not use mutants in this experiment.

## 2 Google OR-Tools

In GA, we initially used Branch Bound solver. In the first 53 tests selected, Branch Bound had many solutions that were quite slow and still not converging over 300 seconds. To test 54, n = 2000, case 06, Branch Bound could not be solved and gave no result. However, when running on a personal computer, it still give results. We believe that Google Colab's limited memory allocation mechanism resulted in this error when our tree was too large and had too many branches.

So we decided to try another solver. That is KNAPSACK MULTIDIMENSION SCIP MIP SOLVER. This solver can deal with both large number of items and several dimensions. This solver is based on Integer Programming solver SCIP. Surprisingly, they solved the test cases very quickly and most of the remaining tests took less than 1 second to give results.

## 3 Result Analysis

In 13 folders of data, we select 8 testcases for each folder. The number of items are 50, 100, 200, 500, 1000, 2000, 5000 and 10000, respectively. The total is 104 testcases.

We attach the results in the same directory as **figure.xlsx**

We can see that GA had significantly lower performance than the 2 solvers of OR-Tools. The solutions given by GA are of equal or lower quality compared to 2 solvers of OR-Tools. Although there is no mutation, but GA has a lot of tests (number of items$> =$ 2000) reaching the time limit.

In the first solver of OR-Tools, use Branch and Bound. We were unable to run test 53 onwards due to the tree scaling too large and Google Colab was unable to allocate memory. Branch Bound's solution for run times less than 300 seconds is nearly optimal and is usually equal to or slightly better than SCIP

[*] This is my report for CS106 - Artificial Intelligence (Spring 2021) at University of Information Technology - Vietnam National University HCMC

solver. However, in cases with a run time more than 300 seconds, it proved to be slightly inferior to the SCIP solver.
On the second solver is SCIP. We got results very quickly, with most tests having run times less than 1 second. Despite having a very fast runtime, some solutions are slightly worse quality than Branch  Bound (where Branch  Bound has runtime less than 300 seconds). However, we believe that it is still a very good choice for cases with large number of items because although the solution is not optimal, the solution time is very short and the solution is acceptable.

## 4 Summary

We think Branch and Bound is an algorithm that can give good results, but it is very slow in case of large number of items. To counter this, we can use SCIP solver for cases with large number of items. While in this experiment GA has proved to be quite poor, but I believe there will be techniques for population size and mutation selection that could make GA even better.