Combinatorial Optimization - Laboratories

Artificial Intelligence 2021 sem. 3

Library Book Assignment

Kacper Trębacz 145453

Jan Gruszczyński 145464

**Theoretical description**

Our task was to implement an algorithm that from a given set of libraries choses such libraries (and assignments of books to them), that it maximizes the sum of values of sent books before the deadline.

Main Part of our algorithm is an genetic algorithm. An individual in this algorithm is a ordered tuple of library ids. The order corresponds to the order in which libraries will be signed up during the signup process.

Initial population consists of a few individuals generated by our heuristic algorithms and some entirely randomly generated ones. Every heuristic algorithms sort libraries according to a given function (details in implementation) and then, selects them till we reach the deadline. There is also one special algorithm that we decided to call “greedy interval solver” because it works the same as the algorithms above, except the fact that it sorts the libraries that are left again every n iterations.

In genetic algorithm the score of an individual is calculated by selecting as many most valuable books as possible excluding books chosen for previous libraries.

Our algorithm has the following steps:

1. Initial population generation:

Here we use various heuristic algorithms to generate initial population. Every heuristic algorithms sort libraries according to a given function (details in implementation) and then, selects them till we reach the deadline. Then we add random individuals till we reach a given population size.

1. Genetic algorithm:

Genetic algorithm starts with our initial population and works in the following way. Every iteration it creates the next population by:

* 1. Preserving a given population percentage by selecting individual by means of a tournament.
  2. While population size is not reached it selects 2 individuals by means of tournament and then with 50% probability it mutates them or performs a simulation of cross-over.

The algorithm until it reaches the end of time dedicated to it.

1. Randomized hill climbing:

In this step we select best individual from previous algorithm (genetic) and perform “randomized hill climbing”. - Essentially we perform a few mutations on this individual, than select best of them, and if it is better than current best we make it the best one. Algorithm runs till it runs out of the time.

1. Max flow max cost

In this step we take individual obtained in previous step and then make library to book assignment be transforming it into flow graph. Then using ortools we solve max-flow and max-cos problems.

Explanation:

At first we generate our population using heuristics, thanks to that the genetic algorithm does not need to waste time generating potentially good solutions from just random ordering. We also added some random solutions, because we want our genetic algorithm to explore more at the beginning and converge to better solutions later thanks to a tournament selection. Than we use randomized hill climbing to tweak a little score of just the potentially best individual .

Till this point we were making book assignments in a ‘greedy’ way, which isn’t always the best. That is why we use max flow min cost to generate best possible assignment of books to libraries for our final individual.

Implementation, conclusion and sources that we have used are enclosed in a separate file called report, which was included in Jupyter noteboook, html and pdf formats.