Heuristic Optimization: Implementation exercise 1

Arnau Dillen - VUB - 0509766

April 2017

1 Implementation

Both algorithms for the experiments are implemented as methods in the Experiments class. These methods take the required parameters as objects implementing the required functionality.
The different neighbourhoods, pivot rules and solution initialisations are implemented following the strategy design pattern. There is an abstract class for each of these three parameters of the algorithms. The InitialSolution abstract class requires it's subclasses to implement a method that generates an initial solution given a problem instance. The same is true for the Neighbourhood class, its subclasses implement a method that returns an improving neighbour given a pivot object. The pivot class implements most of the required functionality. It keeps track of the current solution and decides when a neighbour is accepted as improving. Therefore it is initialised with the initial solution as current one. It also takes an instance object to be able to recompute the weighted completion time of a given neighbour solution based on the current solution and the index of the first element chosen for exchange, transpose or insert. That way it does not recompute the weighted tardiness for the whole solution that was given.
The program that allows us to perform experiments for iterative improvement and variable neighbourhood descent is responsible for creating the required objects. For both algorithms a function is provided that runs the algorithm on all instances and returns both the solution's weighted completion time and the computation time in seconds it took to find this solution. The requested neighbourhood, pivot rule and initialisation method are specified with string parameters. These parameters are parsed from the command line parameters that where provided when running the program.
The command line program for the experiments is called flowshopWCT and works as follows. It takes the name of the requested algorithm first, either ii or vnd. Next the different parameters are specified with – (double dash) before them. For example *flowshopWCT ii –first –transpose –random* runs iterative improvement on all instances with a first improve pivot rule, a transpose neighbourhood and random initial solution.
Due to some trouble with windows build tools for gcc, the make command did

not work for me. The makefile included is not tested and might not work. To compile the program, we need to run all g++ commands in the makefile, compiling all files one by one and finally linking them.

2 Results

We start by looking at the experimental results for each of the algorithm variants. The following list presents an overview of the sum of computation times over instances and the average deviation from the best solution.
Please note that results for best improvement are missing for both exchange and insert neighbourhoods. This was due to a last minute bug I was not able to solve.

Iterative improvement:

- Transpose, first, random
  - Average deviation: 35.78%
  - Total computation time: 4.542 seconds
- Transpose, first, RZ
  - Average deviation: 5.99%
  - Total computation time: 2.85 seconds
- Transpose, best, random
  - Average deviation: 37.38%
  - Total computation time: 3.567 seconds
- Transpose, best, RZ
  - Average deviation: 6.76%
  - Total computation time: 2.557 seconds
- Exchange, first, random
  - Average deviation: 1.83%
  - Total computation time: 1192.898 seconds
- Exchange, first, RZ
  - Average deviation: 2.82%
  - Total computation time: 165.237 seconds
- Insert, first, random
  - Average deviation: 6.32%
  - Total computation time: 1575.8 seconds

- Insert, first, RZ
  - Average deviation: 4.18%
  - Total computation time: 382.621 seconds

Variable neighbourhood descent:

- Transpose-¿Exchange-¿Insert, random
  - Average deviation: 3.61%
  - Total computation time: 1522.598 seconds
- Transpose-¿Exchange-¿Insert, RZ
  - Average deviation: 2.82%
  - Total computation time: 269.044 seconds
- Transpose-¿Insert-¿Exchange, random
  - Average deviation: 1.73%
  - Total computation time: 1186.682 seconds
- Transpose-¿Insert-¿Exchange, RZ
  - Average deviation: 2.49%
  - Total computation time: 171.138 seconds

We will now look at the results of statistical tests that where performed.
For our first test we want to test the impact of the initialisation function on the results. For this purpose we compare the different algorithms that use the same neighbourhood and pivot rule, and only differ in their initial solution. We do this by performing paired t-tests and wilcox tests.
The first comparison we make here is transpose neighbourhood and a first improve pivot rule. The t-test for this setup gives a p-value of $8.005825e-41$. This very small p-value points towards rejecting the null hypothesis that the mean deviation is equal for both setups, we perform a Wilcox test to see if our median values point to the same conclusion. The p-value for the Wilcox test is $1.671329e-11$, it's also very small and lets us conclude that there is a significant difference between the two algorithm variants.
To measure the impact of the initial solution further we perform some more tests. We now try a transpose neighbourhood with a best improvement pivot rule. The t-test and Wilcox-test give p-values of respectively $8.945028e-41$ and $1.671329e-11$. We can again reject the null hypothesis and conclude that there is a significant difference.
We now try the effect of the initial solution for a different neighbourhood. When we perform t-tests and Wilcox tests with the exchange and insert neighbourhoods and a first improve pivot rule we see the following results. For the Exchange neighbourhood the p-values are $2.573476e-14$ and $8.214752e-11$

for t-test and W-test respectively. For the insert neighbourhood those are $7.123403e-13$ and $2.060747e-10$. There does seem to be a significant difference between a random initial solution and one constructed with the RZ heuristic.
As a last test on this, we look at the impact of initial solution for a variable neighbourhood in the same order. The p-values for the t-test and Wilcox-test are respectively 0.0003429367 and 0.0006106798. We again reject the null hypothesis.

Due to last minute errors I was not able to perform the rest of the statistical tests.