Programming Assignment 3

Heuristic Optimization Techniques

David Molnar, 1326891
Daniel Füvesi, 1326576
Gruppe 9

January 15, 2017

1 Advanced Local Search

1.1 General Variable Neighbourhood Search

In the third assignment we chose and developed the General Variable Neighbourhood Search (GVNS) for our probleme instance. GVNS combines the Variable Neighborhood Descent (VND) with the Basic Variable Neighborhood Search (BVNS). To understand how our algorithm works we may discuss the two parts of it. The VND part aims to find the local minimum not only in one neigbourhood but it returns the local optimum with respect of all neigbourhood structures in the set. The other part (BVNS) is "combination of deterministic and stochastic moves in the search space". It iterates through the neighbourhood structure set, generetes a random solution in the neighbourhood and checks if it was a better solution or not. So, the combination of this to approaches uses two sets of local searches, that should not be identical. Therefore we developed in additon few more local searches.

1.2 Additional local searches

In order to easily have more local searches, we did not have to invent new neigbourhood structures, but only scaling them. So, in addition to the existing neigbourhoods (Single Edge Move, Node Swap, Node Edge Move) we developed the Double Edge Move and the Double Node Swap.

2 Experimental Setup

The GVNS with randomized construction heurictic was tested on a Windows 10 PC with Intel Core i7 870 (4 cores, 2.93GHz) and 8GB RAM, implemented in Java without external libraries. For the testing of the GVNS with deterministic conctruction heuristic we used a Windows 10 PC with Intel Core i7 3630QM (4 cores, 2.4GHz) and 8GB ram. All instances received a CPU time limit of 14 minutes. If the running time exceeds the limit (checked on each iteration), the current best solution is returned. Since we have five different local searches, there is lot of combinations

in the two sets of GVNS. We tested many of it to get the proper model to test with. During the evaulation of these tests we found out that the order of the local searches within the sets does have an effect on the quality of the solution. At the end we came up with {Single Edge Move, Node Edge Move, Double Node Swap} and {Single Edge Move, Node Swap, Double Edge Move} for the two sets in the algorithm. We used this model for all the test cases later. For each construction and randomized construction heuristic we tested the algorithm with all step functions with 15 runs. We used incremental evaluation in all of the local searches. For instance when moving an edge from a page to another, we only calculate the deference for the first page, and for the second we only calculate the new crossings and add it to the existing number of crossings.

3 Results

Due to the high amount of possible combinations of local search sets within the algorithm we invested a lot of time in testing. As already mentioned we used the same model for our test cases. It showed that for the test instances 1, 2, 4, 5 both constuction types with any step functions gave the best solutions. In the second half of the test cases we noticed, that the algorithm did not have enough time to terminate. Since the algorithm starts with a gerenation of a random solution, the quality of this initial solution has a great effect on the solution. Tests showed that we can reach the best solutions with the random step function. For the higher instances (6-10) the first and best step functions resulted relatively constant (bad) results. This causes the quitly big differences between the average and the best solutions.

Abbreviations: (step_function)

- **b** ... best step function
- **f** ... first step function
- **r** ... random step function

3.1 Results with Deterministic Construction Heuristic

Instance	Crossings B4 GVNS	Best Crossings	Avg. Crossings	Time	Runs
automatic-1	18	9 (**b**)	9	1414 ms	3x15
automatic-2	12	0 (**f**)	0	45 ms	3x15
automatic-3	127	22 (**b**)	27,4	650'977 ms	3x15
automatic-4	20	0 (**b**)	0	149 ms	3x15
automatic-5	23	0 (**b**)	0,4	5724 ms	3x15
automatic-6	5'751'426	1'998'401 (**r**)	4'494'828	842'376 ms (!)	3x15
automatic-7	9872	7276 (**r**)	8345	840'076 ms (!)	3x15
automatic-8	751'523	200'994 (**r**)	546'144	840'498 ms (!)	3x15
automatic-9	149'327	28'679 (**r**)	106'653	840'445 ms (!)	3x15
automatic-10	75255	16'251 (**r**)	48'560	840'213 ms (!)	3x15

3.2 Results with Randomized Construction Heuristic

Instance	Avg. Crossings B4 GVNS	Best Crossings	Avg. Crossings	Avg. Time	Runs
automatic-1	20	9 (**b**)	9	36010 ms	3x14
automatic-2	12	0 (**f**)	0	1149 ms	3x14
automatic-3	151	21 (**r**)	28	478173ms	3x14
automatic-4	18	0 (**f**)	0	6760 ms	3x14
automatic-5	23	0 (**r**)	0.90	206990 ms	3x14
automatic-6	9'813'265	2'280'656 (**r**)	7'393'025	844817 ms (!)	3x14
automatic-7	24'935	7803 (**r**)	8505	840158 ms (!)	3x14
automatic-8	1'543'365	194'286 (**r**)	1'102'975	841170 ms (!)	3x14
automatic-9	933'825	34'563 (**r**)	622'879	841333 ms (!)	3x14
automatic-10	186'191	15'831 (**r**)	100'160	840415 ms (!)	3x14