Problem Solving and Search in AI

Practical Work by Robin Boeltzig & Lukas Zimmermann

How to use

1. Open input.py
2. Change the value for "input_table" to whatever input matrix you want
   Note: The matrix needs to be entered in one line, as seen in the example below.
3. Change the value for "maxiter" to the desired number of iterations for the Iterated Hill Climber
4. Run input.py

Example:
Input Matrix: 3 3
0 4 1 3 2 2
1 3 2 3 0 2
2 4 1 3 0 2

→ input_table = "3 3 0 4 1 3 2 2 1 3 2 3 0 2 2 4 1 3 0 2"

General workflow of the solver

1. Input matrix gets processed
2. Generate input graph (Disjunctive Graph Model)
3. Generate initial solution through bidirectional algorithm
4. Use iterative hill climbing method to optimize solution
5. Output: Critical Path and makespan of the schedule.

Neighbourhood function of G: Set of all graphs which differ from G in the direction of exactly one disjunctive arc.

Benchmark results compared to optimal solution

name	optimum	maxiter = 20	difference		maxiter = 200	
la01	666	848	27%		770	16%
la02	655	804	23%			
la03	597	805	35%			
la04	590	772	31%			
la05	593	697	18%			

la06	926	1163	26%		1078	16%
la07	890	1108	24%			
la08	863	1134	31%			
la09	951	1140	20%			
la10	958	1092	14%			
la11	1222	1496	22%		1395	14%
la12	1039	1285	24%			
la13	1150	1496	30%			
la14	1292	1470	14%			
la15	1207	1731	43%			
la16	945	1089	15%			
la17	784	848	8%			
la18	848	943	11%			
la19	842	951	13%			
la20	902	1027	14%			

Sources

For this project, we have used various articles and lecture slides to find approaches to the problem. They are listed below:

Disjunctive Graph Model:

- http://www.or.uni-bonn.de/lectures/ss10/scheduling_data/sched10_5.pdf
- http://www.kecl.ntt.co.jp/as/members/yamada/galbk.pdf
- https://acrogenesis.com/or-tools/documentation/user_manual/manual/ls/jobshop_def_data.html

Bidirectional Algorithm to determine feasible solutions:

- Dell'Amico, Mauro & Trubian, Marco. (1993). Applying Tabu Search to the Job-Shop Scheduling Problem. Annals of Operations Research. 41. 231-252. 10.1007/BF02023076.

Calculation of longest path from source to sink:

- https://github.com/csirmaz/dag_longest_path/blob/master/dag_longest_path/dag_longest_path.py