Kajetan Kubik
Bartosz Maślanka

generate_inter_moves:

- ☐ for each node in path:
  - ☐ for each new node not yet in path:
    - ☐ replace node with new node in path
    - ☐ calculate delta
    - ☐ add to output

generate_intra_moves:

- ☐ if we change nodes:
  - ☐ for each possible combination of indexes of nodes in path:
    - ☐ create new path switching those indexes
    - ☐ calculate delta
    - ☐ add to output
- ☐ if we change edges:
  - ☐ for each possible combination of indexes of nodes in path:
    - ☐ create new path reversing nodes between those indexes
    - ☐ calculate delta
    - ☐ add to output

local search:

- ☐ if no path was declared as input, generate random path
- ☐ do until their is change in path
  - ☐ generate all neighbors(generate_inter_moves + generate_intra_moves)
  - ☐ if greedy version:
    - ☐ shuffle neighborhood
    - ☐ for each neighbor in neighborhood:
      - ☐ if neighbor is better then path, neighbor become new path
  - ☐ in steepest version:
    - ☐ sort neighborhood
    - ☐ if change in the best in neighbor is lesser than 0, this neighbor become new path

Results:

	TSPC	TSPD
Greedy Local Search from random path with **edge** swap	51459 (49504 - 54304)	48504 (45802 - 52552)
Greedy Local Search from random path with **node** swap	63668 (57261 - 71584)	61971 (53772 - 72536)
Steepest Local Search from random path with **edge** swap	**51380** (49324 - 54334)	**48019** (45227 - 51657)
Steepest Local Search from random path with **node** swap	66281 (60498 - 73567)	65029 (56013 - 76130)
Greedy Local Search from predefined path with **edge** swap	51074 (49105 - 53589)	47956 (45198 - 51939)
Greedy Local Search from predefined path with **node** swap	63997 (58418 - 71653)	62334 (54172 - 71258)
Steepest Local Search from predefined path with **edge** swap	**50933** (48743 - 53786)	**47914** (45437 - 50774)
Steepest Local Search from predefined path with **node** swap	66716 (57794 - 73072)	65472 (55189 - 75799)
Random	215158 (188514 - 237161)	219142 (190675 - 246691)
Nearest neighbor	61115 (54164 - 67502)	56739 (52821 - 63391)
Cycle	55726 (53140 - 58784)	54530 (50144 - 59907)
2-regret	69028 (65044 - 73039)	70341 (64636 - 74554)
Greedy heuristics with a weighted sum criterion	55984 (53962 - 58242)	53603 (49105 - 59035)

time (in hours):

	TSPC	TSPD
Greedy Local Search from random path with **edge** swap	46 (35 - 55)	46 (32 - 55)
Greedy Local Search from random path with **node** swap	52 (41 - 66)	45 (36 - 62)
Steepest Local Search from random path with **edge** swap	13 (10 - 18)	13 (10 - 16)
Steepest Local Search from random path with **node** swap	18 (14 - 23)	16 (12 - 21)
Greedy Local Search from predefined path with **edge** swap	35 (25 - 49)	42 (31 - 52)
Greedy Local Search from predefined path with **node** swap	34 (24 - 43)	38 (27 - 49)
Steepest Local Search from predefined path with **edge** swap	12 (9 - 16)	13 (8 - 17)
Steepest Local Search from predefined path with **node** swap	13 (10 - 19)	15 (11 - 19)
Old methods:	Almost instant, or around 1 second	

Conclusion:

- Predefined path is generated by 2-regret.
- In general, we observe that starting from a predefined path, which is known to be somehow good, yields better results than starting from a random path.
- Also, in local search, performing edge swap in generating neighbors (intra moves) for our current solution gives us a much better score then performing node swap.
- It is hard to say which version of local search (greedy vs steepest) is better, because for the edge version, both yield very similar results.
- Also, local search in the node version does not beat Greedy heuristics with a weighted sum criterion or greedy cycle.
- The best solution was achieved by: local search in steepest version, performing edge swap, starting from predefined solution.
- Steepest version in general is faster.
- The slowest solution was Greedy Local Search from random path with node swap
- For greedy version, starting from predefined solution gives significant time upgrade
- For steepest version, starting from predefined solution gives small time upgrade
- Unfortunately, the random solution is still the worst one.