Evolutionary Computation - lab assignment 9

Szymon Bujowski, 148050, source: https://github.com/bujowskis/put-evolutionary-computation

Problem description

Hybrid evolutionary algorithm

The goal is to implement a hybrid evolutionary algorithm and compare it with the MSLS, ILS, and LNS methods implemented in the previous assignments.

Proposed algorithm parameters:

- Elite population of 20.
- Steady state algorithm.
- Parents selected from the population with the uniform probability.
- There must be no copies of the same solution in the population (you can compare the entire solution or the value of the objective function).

Proposed recombination operators:

- Operator 1. We locate in the offspring all common nodes and edges and fill the rest of the solution at random.
- Operator 2. We choose one of the parents as the starting solution. We remove from this solution all edges and nodes that are not present in the other parent. The solution is repaired using the heuristic method in the same way as in the LNS method. We also test the version of the algorithm without local search after recombination (we still use local search for the initial population).

If the algorithm described above would cause premature convergence, it can be modified, e.g. additional diversification preservation mechanisms.

Additionally, another custom recombination operator can be proposed.

Experiment parameters same as ILS/LNS.

Report – analogous to before.

Implemented algorithms and pseudocodes

Notes

- 50% was the required number of nodes in a solution, and “required number of nodes” will be referred to this way in the pseudocode
- Pseudocode is simplified to the required configuration of Local Search - that is: Steepest LS, Random starting solution, two edges exchange intra moves
- LS with candidate moves was used
- Solutions are considered “the same” based on **objective function** value, not their whole
- Greedy heuristic used is Nearest Neighbor at any place

Hybrid Evolutionary Algorithm

1. Generate population of unique initial solutions using LS
2. While running time is lower than specified timeout:
2.1. Randomly choose 2 solutions from the population to be used as parents (parent1, parent2)
2.2. Generate child by applying recombination operator
2.2.1. Locate common nodes of the parents (nodes that are in both parents)
2.2.2. Locate common edges (edges that are in both parents, in either direction)
2.2.3. Create stripped solution - copy of parent1, without nodes that are not common nodes
2.2.3.Ad.1. Note that common edges will also be preserved
2.2.4. Create recombined solution - fill the stripped solution with nodes until it has required number of nodes
2.2.4.Ad.1. If random recombination operator is used - insert random node from outside cycle at random place in cycle, except between common edges
2.2.4.Ad.2. If heuristic recombination operator is used - insert nodes using nn at any heuristic, except between common edges
2.2.5. (Depending on configuration) apply LS on recombined solution
2.2.6. Return recombined solution as child
2.3. If child is unique and worst solution in the population has higher objective function value is higher than that of child, remove worst from population and replace it with the child
3. Return the best solution from the population

Results

Note: All best solutions were checked with the solution checker

Statistics

Method	**TSPA** avg (min - max)	**TSPB** avg (min - max)
Multiple Start Local Search (MSLS)	72324 (71458 - 72825)	46801 (45825 - 47480)
Iterated Local Search (ILS)	69498 (69259 - 70321)	43858 (43568 - 44484)
Large-Scale Neighborhood Search (LSNS) - no LS	69935 (69230 - 71274)	44437 (44984 - 46112)
Large-Scale Neighborhood Search (LSNS) - with LS	69774 (69230 - 70258)	44373 (43550 - 45506)
Evolutionary Algorithm heuristic, no LS	70631 (70082 - 70996)	44453 (43759 - 45383)
Evolutionary Algorithm heuristic, with LS	70169 (69424 - 70688)	44050 (43555 - 44822)
Evolutionary Algorithm random, no LS	73164 (72330 - 74375)	47593 (46332 - 48515)
Evolutionary Algorithm random, with LS	70913 (70041 - 71706)	45098 (44274 - 45669)

Running times (s)

Method	**TSPA** avg (min - max)	**TSPB** avg (min - max)
Multiple Start Local Search (MSLS)	47.48 (42.77 - 53.01)	45.31 (44.61 - 45.89)
Iterated Local Search (ILS)	47.37 (47.26 - 47.51)	47.39 (47.26 - 47.54)
Large-Scale Neighborhood Search (LSNS) - no LS	47.29 (47.22 - 47.36)	47.28 (47.20 - 47.37)
Large-Scale Neighborhood Search (LSNS) - with LS	47.26 (47.20 - 47.32)	47.39 (47.18 - 47.54)
Evolutionary Algorithm heuristic, no LS	51.53 (51.36 - 51.68)	53.90 (51.47 - 55.08)
Evolutionary Algorithm heuristic, with LS	51.88 (51.25 - 53.65)	53.74 (53.53 - 54.18)
Evolutionary Algorithm random, no LS	50.75 (50.61 - 50.83)	53.92 (53.59 - 54.30)
Evolutionary Algorithm random, with LS	51.95 (50.71 - 54.07)	54.05 (53.71 - 54.63)

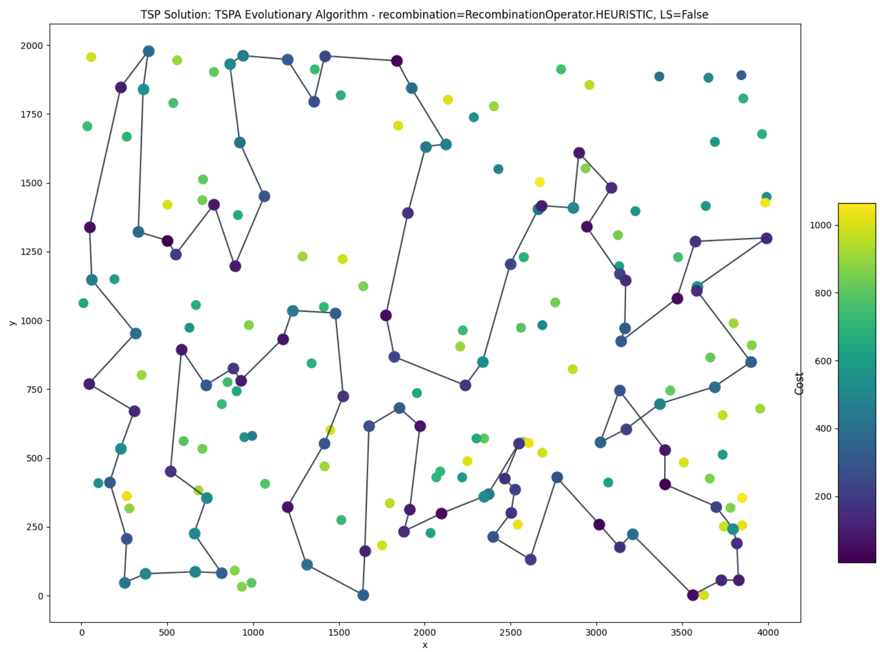
Number of Local Search / main loop runs

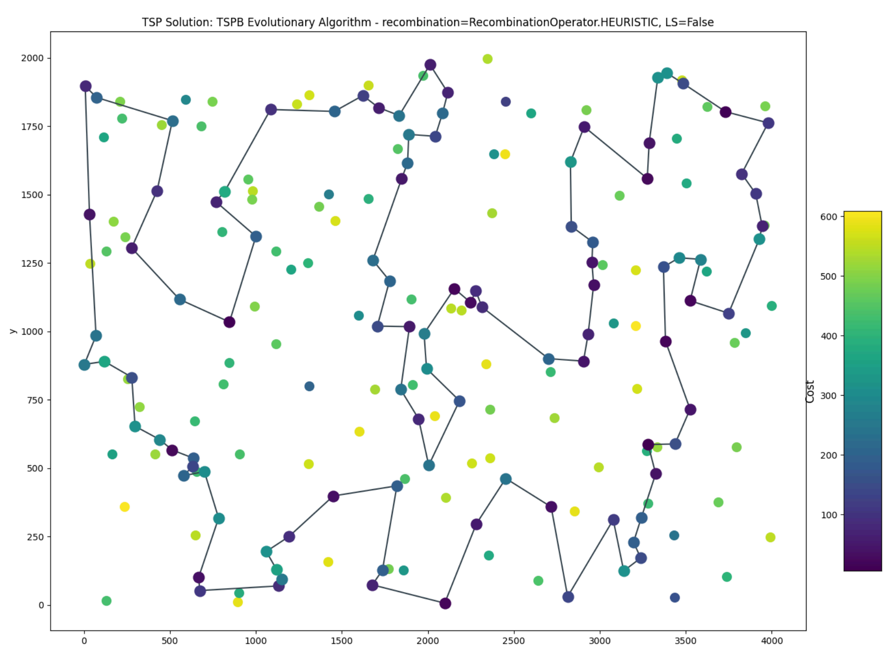
Method	**TSPA** avg (min - max)	**TSPB** avg (min - max)
Multiple Start Local Search (MSLS) *Number of LS runs*	200 (200 - 200)	200 (200 - 200)
Iterated Local Search (ILS) *Number of LS runs*	901 (815 - 951)	901 (875 - 940)
Large-Scale Neighborhood Search (LSNS) - no LS *Number of main loop runs*	678 (494 - 785)	720 (682 - 756)
Large-Scale Neighborhood Search (LSNS) - with LS *Number of main loop runs*	613 (592 - 622)	616 (593 - 630)
Evolutionary Algorithm heuristic, no LS *Number of main loop runs*	2348 (2033 - 2689)	1095 (908 - 1331)
Evolutionary Algorithm heuristic, with LS *Number of main loop runs*	1043 (779 - 1148)	637 (551 - 673)
Evolutionary Algorithm random, no LS *Number of main loop runs*	233518 (229341 - 236612)	125280 (122092 - 128467)
Evolutionary Algorithm random, with LS *Number of main loop runs*	891 (627 - 1073)	483 (459 - 523)

Best solutions

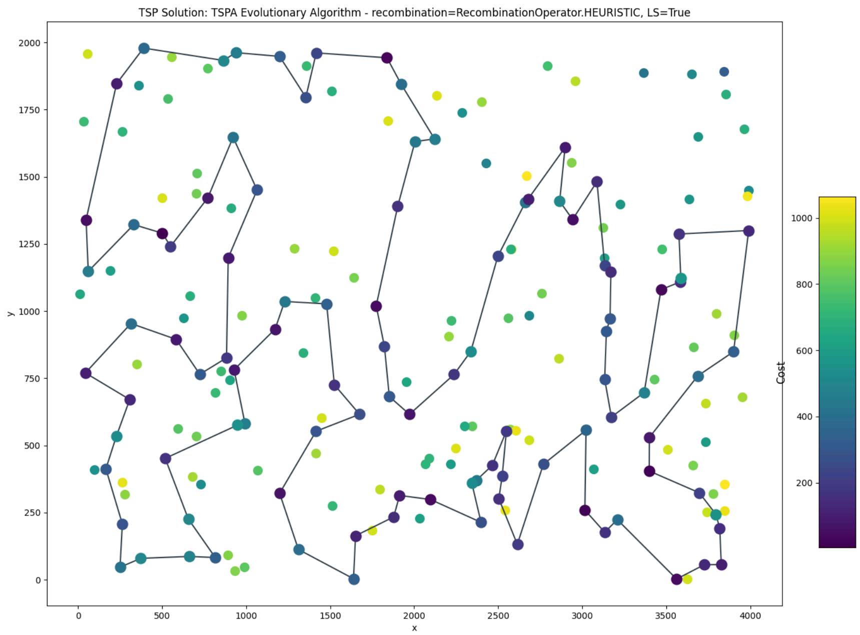
Note: additional cost is depicted using a color scale

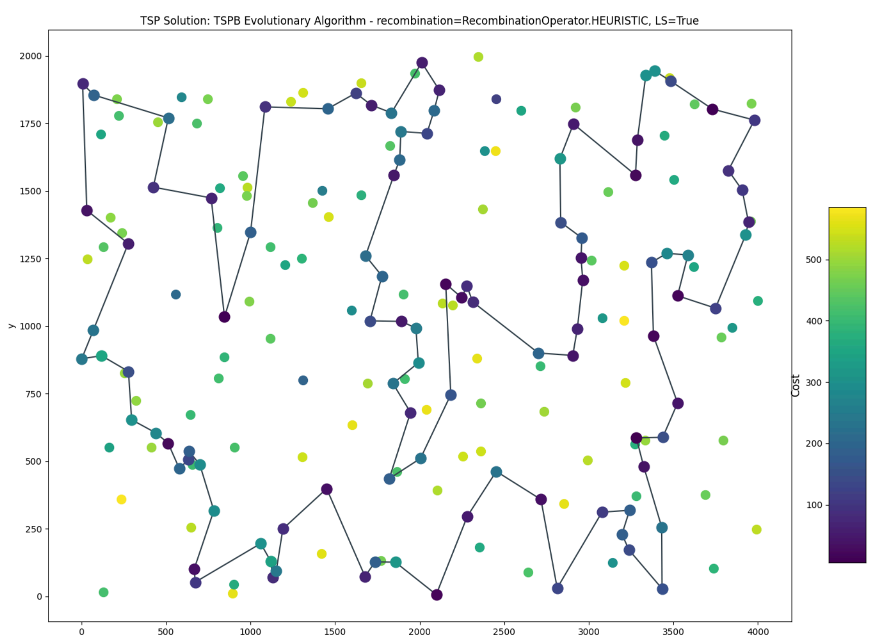
Evolutionary Algorithm - heuristic, no LS



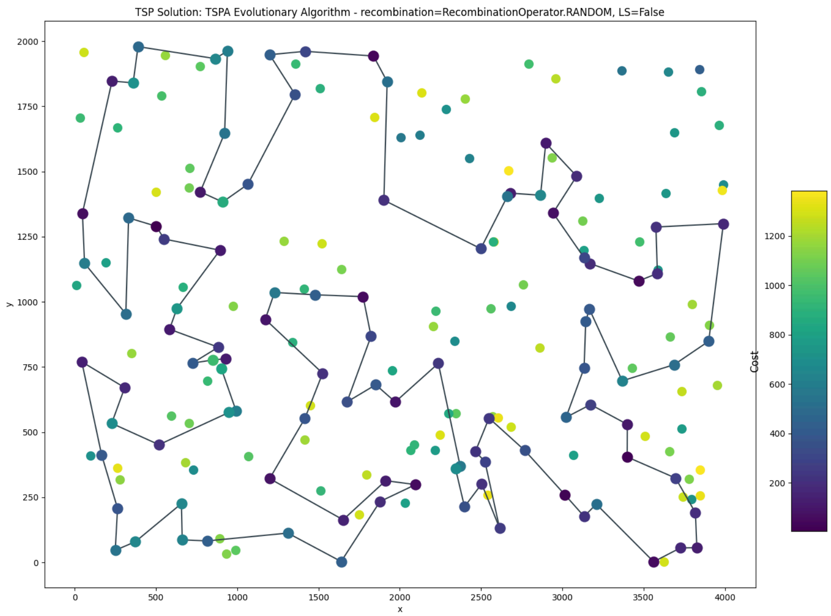


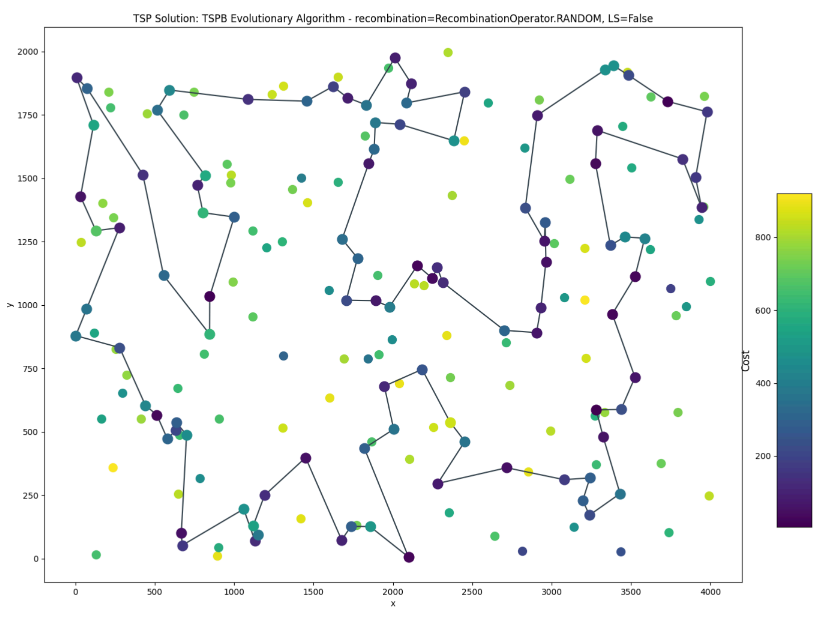
Evolutionary Algorithm heuristic, with LS



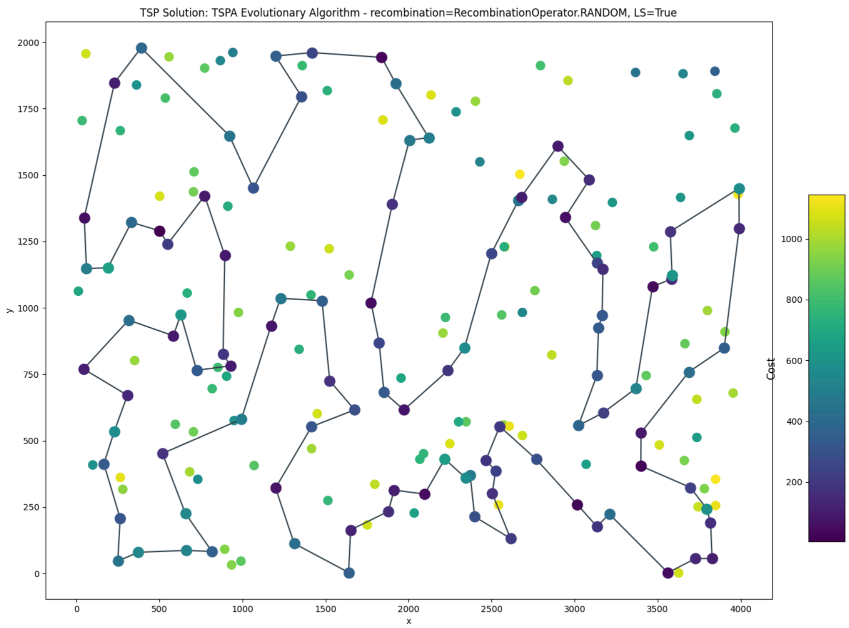


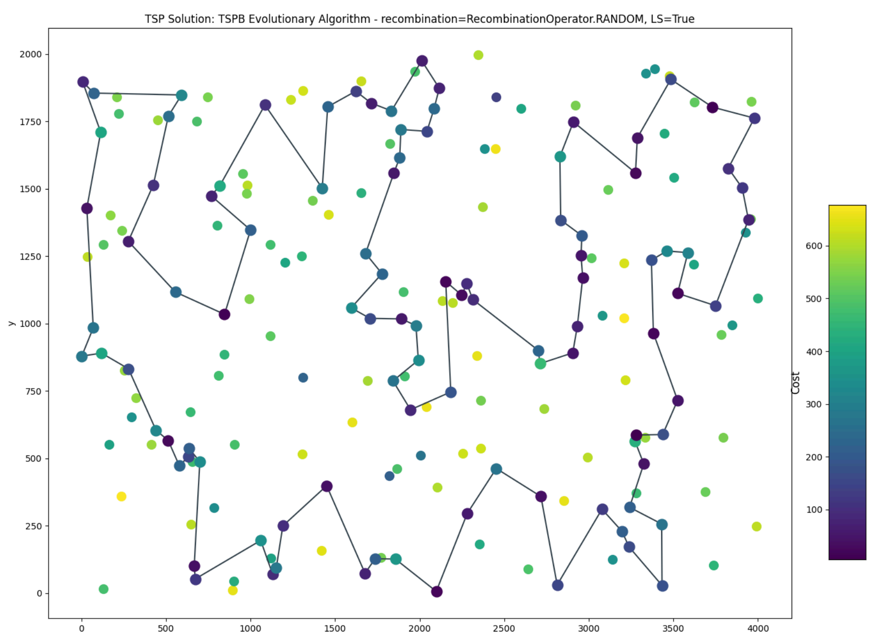
Evolutionary Algorithm random, no LS





Evolutionary Algorithm random, with LS





Conclusions

- EA (random, no LS) is overall the worst configuration of EA - despite a few factors higher number of main loop runs, it fails to generate meaningful results, compared to other methods
- EA (heuristic, with LS) is overall the best configuration of EA - despite its relatively low number of main loop runs (still better than “random, with LS” and relatively on pair with other compared methods) the solutions it explores are of high quality, leading to good results
- ILS still remains superior method of all compared ones
- Note that EA’s (in)effectiveness depends highly on its configuration - it’s possible that it may be superior even to ILS, given other config