Evolutionary Computation - lab assignment 4

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

Problem description

The goal of the task is to improve the time efficiency of the steepest local search with the use of candidate moves using the neighborhood, which turned out to be the best in the previous assignment,

As candidate moves, we use moves that introduce at least one candidate edge to the solution. We define the candidate edges by determining for each vertex 10 other “nearest” vertices (“nearest” taking into account sum of the edge length and vertex cost). This parameter (10) can also be selected experimentally to obtain the best results.

Note that both in the case of intra-route moves (e.g. two-edges exchange) and inter-route moves (nodes-exchange with one selected one not selected) we should ensure that at least one candidate edge is introduced to the solution. In particular, it is not correct to exchange a selected node with one of its nearest neighbors without adding to the solution at least one candidate edge.

Note that there are in both cases two moves that introduce a given candidate edge. As starting solutions use random solutions.

As baseline report also results of the steepest local search with random starting solutions without these mechanisms.

Computational experiment: Run each of the methods 200 times.

Reporting results: Use tables as in the previous assignment.

The outline of the report as previously

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
- “**Total move cost**” is the total change in objective function after adding a node, that is, its edge distance and additional cost
- Pseudocode is simplified to the required configuration of Local Search - that is: Steepest LS, Random starting solution, two edges exchange intra moves

Local Search with Candidate Moves

```
1. Generate random initial solution
2. Initialize list of all possible improving moves from each node in
cycle to its 10 closest vertices (distance + cost wise)
     2.Ad.1. Improving moves - that is, those resulting in decrease of
objective function
     2.Ad.2. In case a given “closest node” is within cycle - intra
edges exchange
     2.Ad.3. In case a given “closest node” is not within cycle -
inter nodes exchange
3. WHILE there are improving moves
3.1. Choose a move yielding the best improvement
3.2. Apply the improving move on the cycle
3.3. Update list of all possible improving moves (
4. Return the cycle
```

Results

Note: All best solutions were checked with the solution checker
Note: Local Search config is provided in order:
(search_type, initial_solution, intra_moves)

Statistics

Method	**TSPA** avg (min - max)	**TSPB** avg (min - max)
Random	265351 (236147 - 290929)	213808 (193230 - 236483)
NN at any	73173 (71179 - 75450)	45870 (44417 - 53438)
Local Search (steepest, random, two_edges)	74222 (71725 - 78624)	48662 (46269 - 52049)
Local Search candidate moves (steepest, random, two_edges)	80467 (75373 - 85896)	49613 (46306 - 53763)

Running times (s)

Method	**TSPA** avg (min - max)	**TSPB** avg (min - max)
Random	0.00008 (0 - 0.00133)	0.00008 (0 - 0.00150)
NN at any	0.15767 (0.14758 - 0.1954)	0.16658 (0.15078 - 0.53)
Local Search candidate moves (steepest, random, two_edges)	1.68 (1.35 - 2.19)	1.77 (1.34 - 2.14)
Local Search candidate moves (steepest, random, two_edges)	0.63 (0.55 - 0.69)	0.60 (0.54 - 0.69)

Local Search statistics

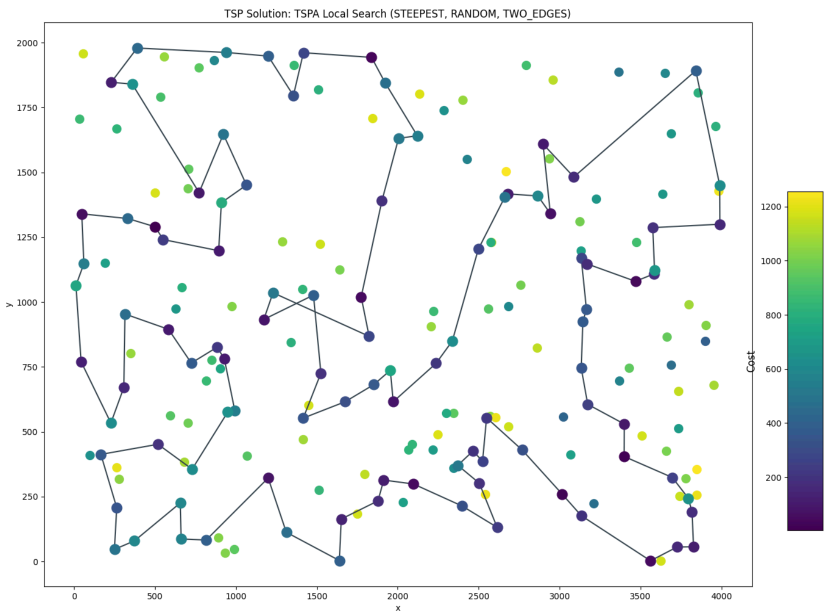
Note: additional statistics I thought may be interesting to investigate

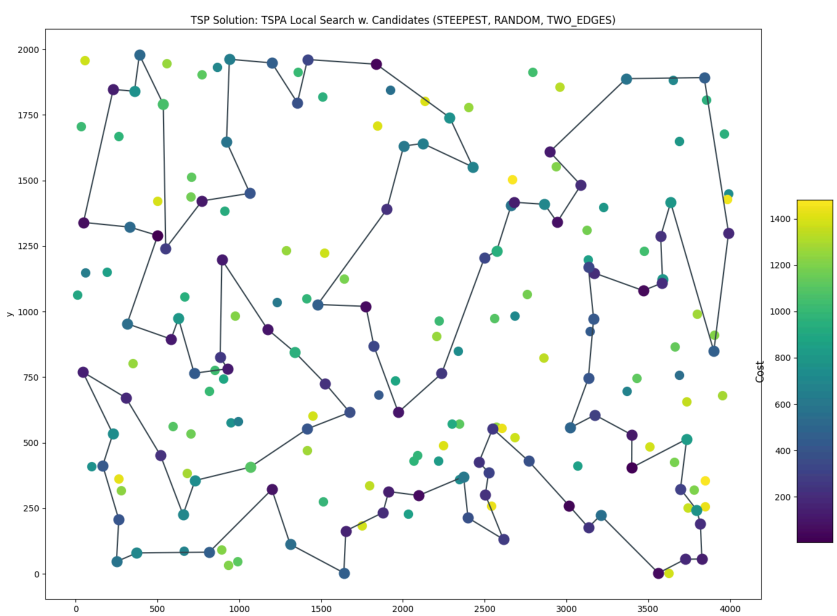
Method	**Moves evaluated** TSPA avg (min - max) TSPB avg (min - max)	**Total moves** TSPA avg (min - max) TSPB avg (min - max)	**Inter moves %** TSPA avg (min - max) TSPB avg (min - max)	**Intra moves %** TSPA avg (min - max) TSPB avg (min - max)
Local Search (steepest, random, two_edges)	1196022 (974681 - 1465351) 1205847 (959433 - 1439920)	132 (118 - 148) 134 (119 - 160)	41 (35 - 51) 41 (34 - 49)	59 (49 - 65) 59 (51 - 66)
Local Search candidate moves (steepest, random, two_edges)	417753 (370800 - 469989) 385893 (344612 - 445006)	134 (121 - 149) 140 (126 - 162)	28 (22 - 34) 32 (26 - 39)	72 (66 - 78) 68 (61 - 74)

Best solutions

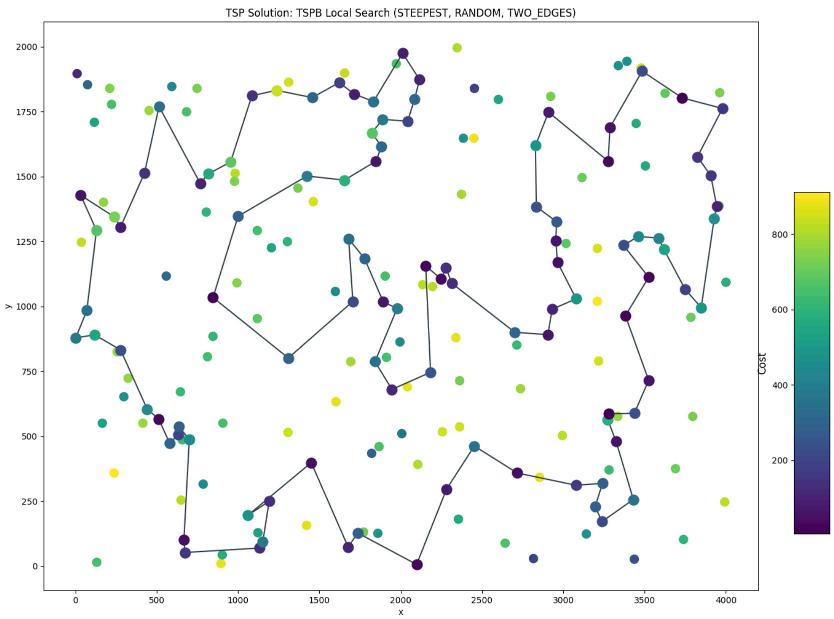
Note: additional cost is depicted using a color scale

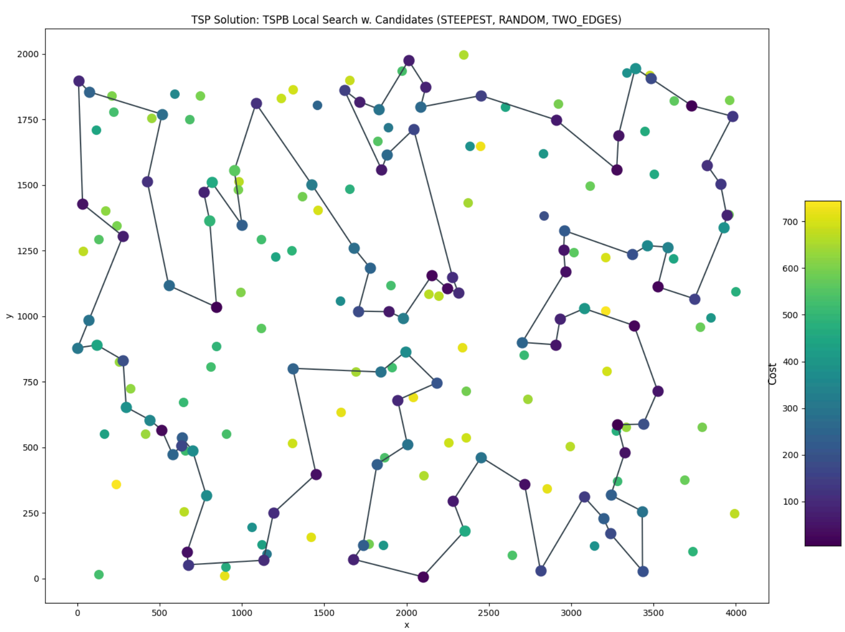
TSPA





TSPB


TSP Solution: TSPB Local Search (STEEPEST, RANDOM, TWO_EDGES)
x
y
Cost


TSP Solution: TSPB Local Search w. Candidates (STEEPEST, RANDOM, TWO_EDGES)
x
y
Cost

Conclusions

- Use of candidate moves leads to significant time complexity improvement;
  On the considered problem instances
    - Around 3x smaller avg running times
    - Around 3x less moves evaluated
- Use of candidate moves leads to deterioration of solutions' quality, relative to "full" LS
    - Relatively insignificant deterioration in case of TSPB
        - It's possible that with not as big differences in costs as in TSPA, choosing the right edges matters more than choosing nodes with smaller costs
        - Consequently, "closest nodes" are more likely to be relatively optimal
    - Considerably higher deterioration in case of TSPA
        - It's possible that with relatively big differences in costs compared to TSPB, choosing the right nodes is of higher importance
        - Consequently, "closest nodes" may not capture the fact that there are better moves available in wider neighborhood
- It's worth mentioning that there exist heuristic approaches (e.g. NN at any) that may offer both better results and smaller time complexity