The use of candidate moves in local search

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.

Since our goal is to improve time efficiency of local search, it is not acceptable to iterate over all moves and check in the move is candidate or not. We should iterate only over candidate moves. Foe example, we could have the first loop over all selected nodes and then second loop over all nodes nearest nodes to the node from the first loop. Then, depending on whether the node is selected or not, we evaluate appropriate candidate moves.

As starting solutions use random solutions.

As a 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.