Large neighborhood search

Reminder of the method:

Generate an initial solution **x**

x := Local search (**x**) *(optional)*

Repeat

y := Destroy (**x**)

y := Repair (**y**)

y := Local search (**y**) *(optional)*

If $f(\mathbf{y}) > f(\mathbf{x})$ **then**

x := **y**

Until stopping conditions are met

Destroy operator should remove a relatively large fraction of nodes/edges from the current solution, e.g. 20-40% (30% by default). The removed edges could be selected at random, as a single subpath, or several subpaths. You can try to propose some heuristic rules to remove "bad" nodes/edges, e.g. long edges or costly nodes. Such heuristics should, however, be randomized not completely deterministic. For example the probability of removal should depend on the length/cost.

As repair operator use the best greedy heuristic (including greedy-regret) from previous assignments.

The destroy-repair operators should be clearly described.

As the starting solution use random solution.

Implement two versions of LNS – using or not local search after destroy-repair operators. Use the best version of steepest local search. Always apply local search to the initial solution.

Computational experiment: Run each of the methods (with and without local search) 20 times for each instance. Use the average running time of MSLS from the previous assignment. Report also the number of iterations of the main loop.

Reporting results: Use tables as in the previous assignments. Add a table with the number of iterations of the main loop. Include results of MSLS, ILS, and the best greedy heuristic (including regret) an basic local search.

The outline of the report as previously.