Programming assignment 1

Heuristic Optimization Techniques, 2018WS

October 11, 2018

Single-solution-based Metaheuristics [30 points]

The first programming assignment is to develop your *own* single-solution-based metaheuristics for the Cost-Balanced Traveling Salesperson Problem (CBTSP). The subtasks for this exercise are:

1. Can you think of a real-world application for this problem?
   Hint: A more general problem is to find a Hamiltonian cycle with cost closest to some value x. This problem can be reduced to the CBTSP by transforming edge costs via $d_{i,j} = d'_{i,j} - x/|V|$.
2. Give a formula to calculate a big-M constant $M \in \mathbb{N}$ for a given graph G with edge costs $d_{i,j}$. Prove that when using M as cost for all non-existing edges, the objective of any "infeasible" solution is greater than the objective of every feasible solution.
3. Construct a Hamiltonian graph G with edge costs and an "infeasible" solution that is a 2-opt solution, i.e., a solution that cannot be improved by the exchange of two edges.
4. Develop a deterministic construction heuristic.
5. Develop a randomized construction heuristic.
6. Develop a framework for simple local search which is able to deal with
   - different neighborhood structures
   - different step functions (first-improvement, best-improvement, random)
7. Develop at least three different neighborhood structures.
8. Implement a GRASP using your randomized construction heuristic and one of your neighborhood structures with one step function.
9. Develop a VND framework which uses your neighborhood structures.
10. Implement one of the following metaheuristics:
    - General Variable Neighbourhood Search (GVNS) which uses your VND
    - Simulated Annealing (SA)

- Tabu Search (TS)

11. Run experiments and compare all your algorithms on the given instances:
    (a) deterministic and randomized construction heuristic
    (b) Use the solution of the deterministic construction heuristic to test the other implementations:
        i. Local search for each of your neighborhood structures using each of the three step functions (at least 9 different algorithms)
        ii. VND
        iii. GVNS, SA, or TS
12. **Use incremental evaluation whenever possible**. Test your approaches once with incremental evaluation and once without incremental evaluation and compare the performance and running times.
13. Write a report containing the description of your algorithms and the experiment results (see the general problem description)

For the development and the report consider the following points:

- How is your solution represented?
- Ad 2: How do you generate different solutions? Which parts of your algorithm can be reasonably randomized and how can you control the degree of randomization?
- Ad 1 & 2: Does randomization improve the generated solutions?
- What parameters do you use and which values do you chose for them?
- Can subsequent – possibly non-improving – moves in your neighborhood structures reach every solution in the search space?
- Local search: How many iterations does it take to reach local optima? What does this say about your neighborhood structures?
- How does incremental evaluation work for your neighborhood structures?
- VND: Does the order of your neighborhoods affect the solution quality?

Hand in your report via TUWEL until *2018-11-25, 23:55*. For further questions send an e-mail to: `heuopt@ac.tuwien.ac.at`