TU Wien
Institute for Logic and Computation
Algorithms and Complexity Group

Heuristic Optimization Techniques, WS 2023

November 15, 2023

Exercise Sheet 2

These exercises have to be solved alone. Hand in a pdf-file with your answers/solutions in a well-readable and understandable form **and** mark the questions you have solved in TUWEL by **Friday, 1st of December 2023, 23:55**. The exercise sheet contains five tasks yielding different numbers of points. Via TUWEL please register, coordinating with your team colleague so that you appear in the same slot, for an interview session by **Friday, 1st of December 2023, 23:55**. Questions related to your solutions and more generally to the topics of the respective lecture units will be asked. You may get up to eight additional points in this interview.

Q: (8P) Prepare yourself for questions about the following topics:

- Simulated Annealing
- Tabu Search
- Configuration Checking
- Evolutionary/Genetic Algorithms
- Ant Colony Optimization

1. (2P) This exercise concerns the 0-1 Knapsack Problem: Assume a volume capacity $c_{\text{vol}} \in \mathbb{R}^+$ and set of goods represented by $G = \{1, \ldots, n\}$ with associated prizes c_i, $i \in G$, as well as volumes v_i, $i \in G$, are given. The goal of the problem is to find the selection of goods whose (i) summed volumes do not exceed c_{vol} and (ii) summed prizes are maximum.

   Consider the 0-1 Knapsack Problem instance $\mathcal{I}$ given by

   $$c_{\text{vol}} = 24,\ |G| = 7,\ \text{and}\ [(c_i, v_i)]_{i=1}^{|G|} = [(22,10),(4,2),(8,4),(18,8),(16,6),(8,2),(18,12)].$$

   Start from the initial solution $x^0 := \{1, 2, 4\} \subseteq G$.

   Assume you have a random number generator yielding the values

   $$[0.3, 0.05, 0.4, 0.1, \ldots, \langle \text{your own random values from } [0,1) \text{ if needed} \rangle]$$

   in this order.

   Perform by hand the Simulated Annealing (SA) algorithm as follows: Consider the list $L = [z_1, \ldots, z_\ell]$ of all feasible neighbors of the current solution sorted here in increasing order of their objective values; see below for the specification of the neighborhood.

   Draw a random number ρ and assume $z_k = L[k]$ with $k = \lceil \rho \cdot \ell \rceil$ is the neighboring solution that is selected by the random neighbor step function within the SA. Decide if this solution will be accepted as new incumbent by the Metropolis criterion. Continue the execution of the SA algorithm until the second acceptance of a new solution.

   The neighborhood $\mathcal{N}(x)$ of a candidate solution $x \subseteq G$ is given by all subsets $y \subseteq G$ where the symmetric set difference $x \ominus y = (x \cup y) \setminus (x \cap y)$ has exactly two elements and y is a *feasible* solution.

   Assume the temperature within the SA to be $T = 2.86$ and ignore cooling in this example.

   What are the weaknesses of this specific SA approach? Explain what will be the problem if the instance size gets very large. How would you realize the selection of the next neighbor.

2. For the MaxSAT problem on an instance made up of variables v_i, $i = 1, \ldots, n$, and clauses C_i, $i = 1, \ldots, m$, consider the following selection strategy of a neighbor in the 1-flip neighborhood:

"Determine the number g_i of additionally satisfied clauses (with respect to the current count of satisfied clauses) when variable v_i, $i = 1, \ldots, n$, is flipped and select $i^* \in \operatorname{argmax}\{g_i : i = 1, \ldots, n\}$ (in case the choice of i^* is not unique, the lexicographically prior variable is selected)"

(a) (2P) Find a small MaxSAT instance $\mathcal{A}$ on variables $(v_1, \ldots, v_n)$ *simultaneously* fulfilling the following properties:

- Starting from the truth value assignment $(v_1, \ldots, v_n) \mapsto (0, \ldots, 0)$, selecting the 1-flip neighbor y_{next} according to the above strategy (i.e., y_{next} will have variable v_{i^*} flipped) and visiting y_{next} as successive solution, never will reach a global maximizer of $\mathcal{A}$, even if this process would be re-applied arbitrarily often to its last visited solution.
- A *Tabu Search* (add a brief description) relying on the 1-flip neighborhood structure and ranking[1] non-tabu 1-flip neighbors according to their above defined gain g_i visits the global optimum of the problem instance after a certain number of steps.

(b) (2P) Find a small MaxSAT instance $\mathcal{B}$ on variables $(v_1, \ldots, v_n)$ *simultaneously* fulfilling the following properties:

- Starting from the truth value assignment $(v_1, \ldots, v_n) \mapsto (0, \ldots, 0)$, selecting the 1-flip neighbor y_{next} according to the above strategy (i.e., y_{next} will have variable v_{i^*} flipped) and visiting y_{next} as successive solution, never will reach a global maximizer of $\mathcal{B}$, even if this process would be re-applied arbitrarily often to its last visited solution.
- A *Configuration Checking* (CC) algorithm (add a brief description) relying on the 1-flip neighborhood structure and ranking[2] 1-flip neighbors (whose configuration[3] indeed changed in the meantime) according to their above defined gain g_i visits the global optimum of the problem instance after a certain number of steps.

3. (3P) Design a Genetic Algorithm that solves Sudokus.

- Describe what a Chromosome looks like.
- How will the fitness function look like? Is it possible that you produce invalid solutions? If so how do you plan to handle those?
- Discuss how to apply, at least two different recombination operators to your solution. Design at least one of the recombination operators yourself specifically to be applied to Sudoku. This operator should utilize the rules of Sudoku. Discuss how you expect them to perform.
- Design two mutation operators and describe how to apply them to your solutions. What performance do you expect of these operators?
- Can your GA design be expanded to solve larger Sudokus containing the numbers 1 through n^2 where $n \in \mathrm{N}$ on a n^2 rows, columns and boxes. If so how can you expand your design?

1	2	3	4	5	6	7	8	9
4	5	6						
7	8	9						
2								
5								
8								
3								
6								
9								

Figure 1: Partially filled Sudoku.

[1] Higher gain means higher priority of selection.
[2] Again, higher gain means higher priority of selection.
[3] You are free to employ any of the two versions of CC discussed in the lecture.

4. (3P) Design a variant of ACS that solves Sudokus.
   - How does the construction graph look like? How many edges and nodes does the graph have?
   - How does an ant set a solution and what objective function can be used?
   - How does the pheromone update work? What initial values for the pheromone matrix seem to be reasonable?
   - What information can we use during construction to prune the construction graph?