**Prof. Dr. Goran Glavaš,**
**M.Sc. Fabian David Schmidt**
**M.Sc. Benedikt Ebing**
Lecture Chair XII for Natural Language Processing, Universität Würzburg

# 9. Exercise for "Algorithmen, KI & Data Science 1"

## 1 Metaheuristic Search

1. How do state space search problems differ from discrete constraint optimization problems?
2. Define the 0-1 knapsack problem as discrete constrained optimization problem. The 0-1 knapsack problem is defined as follows: given a set of $N$ items, each with a weight $w$ and a value $v$, determine the subset of items to choose such that the total weight is less than or equal to a given limit $W$ and the total value is as large as possible.

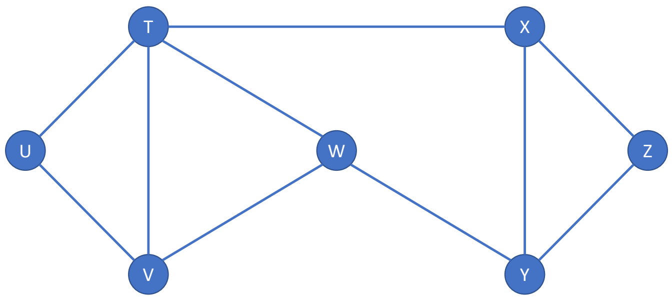
   Example 0-1 knapsack problem:
   You are packing your bag pack for a trip. You have the following items at home:

   - knife, weight 2, value 6
   - water, weight 7, value 9
   - sleeping bag, weight 5, value 7
   - mobile phone, weight 1, value 2
   - jacket, weight 2, value 3

   With $W = 7$, you could pack {knife, sleeping bag} with weight 7 and value 13 or {mobile phone, jacket} with weight 3 and value 10 or ..., but you would not be allowed to pack {knife, water} with weight 9.
3. How does simulated annealing escape local optima?
4. How does genetic algorithm escape local optima?

## 2 Constrained Satisfaction Problems

1. How do state space search problems differ from constrained satisfaction problems?
2. How many solutions does the map coloring problem from lecture 18 have? How many if two colors are allowed?
3. Explain why choosing the variable that is most constrained, but the value that is least constrained is considered a good heuristic?
4. Consider the problem of placing k knights on an $n \times n$ chessboard such that no two knights are attacking each other, where $k$ is given and $k \leq n^2$. Choose a CSP formulation giving your (a) variables, (b) the domain of each variable, and (c) the constraints.
5. Consider the following graph that represents a coloring problem. Each vertex of the graph should be colored with either blue (B), green (G) or red (R), such that adjacent nodes do not have the same color (similar to the coloring problem shown in lecture 18).



Fill out the table below by applying the backtracking algorithm with inference, degree heuristic and least constraining value heuristic to the coloring problem. In case the heuristics do not produce a single next best move, process the variables and values in lexicographical order. The assignment of a value to a variable is depicted with "(a)".

| | T | U | V | W | X | Y | Z |
|---|---|---|---|---|---|---|---|
| Init | B,G,R | B,G,R | B,G,R | B,G,R | B,G,R | B,G,R | B,G,R |
| 1. Iter | B (a) | G,R | G,R | G,R | G,R | B,G,R | B,G,R |
| 2. Iter | | | | | | | |
| 3. Iter | | | | | | | |
| 4. Iter | | | | | | | |
| 5. Iter | | | | | | | |
| 6. Iter | | | | | | | |
| 7. Iter | | | | | | | |
| 8. Iter | | | | | | | |

6. Implement the class $ColoringCSP$ and the additional methods in the provided $.ipynb$ to solve coloring problems similar to the one presented in lecture 18. You should select variables and colors in lexicographical order. In the provided $.ipynb$, you can find a cell that provides examples on how to use the class' attributes. Implement the following methods:

   a) $complete(s, csp)$: The method takes the current assignment $s$ and the $csp$ as input. It returns $True$ if the assignment is complete and $False$ otherwise.

   b) $select\_unassigned\_var(csp)$ : The method takes the $csp$ as input. It returns an unassigned variable in lexicographical order.

   c) $order\_values(csp)$ : The method takes the $csp$ as input. It returns the values in lexicographical order.

   d) $violates(s, v, val)$ : The method is a class method of $ColoringCSP$. It takes the current assignment $s$, and the new variable $v$ together with its new value $val$ as input. It returns $True$ if the new assignment violates the constraints and $False$ otherwise.

   e) $backtrack(s, csp)$ : Implement the naive backtracking from in lecture 18.