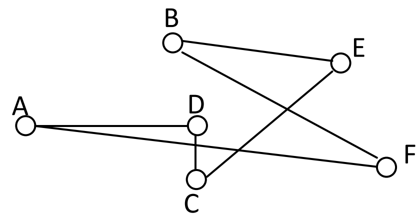
Exercises Lecture 10
Intelligent Systems Programming (ISP)

Exercise 1

Consider the following ***initial state*** of a Travelling Salesman Problem (TSP) (the problem of finding the shortest cyclic path from the start state to all the states and back to start) with 6 cities.



All cities can be reached from any other city and the following routing table describes the distance between each city:

	A	*B*	*C*	*D*	*E*	*F*
A	-	3	3	4	7	8
B	3	-	3	2	4	7
C	3	3	-	1	3	4
D	4	2	1	-	2	4
E	7	4	3	2	-	2
F	8	7	4	4	2	-

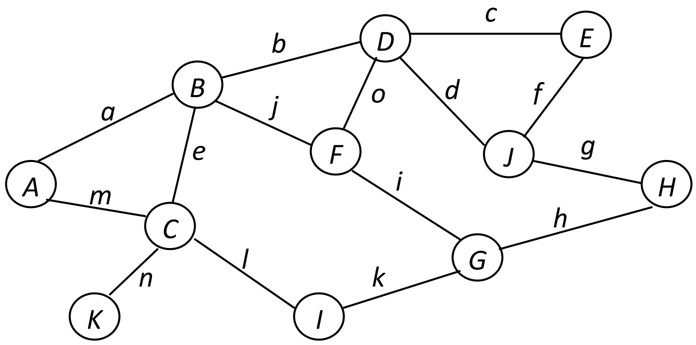
Given that *A* is the starting point, the state can be modeled as a sequence of cities $s=<s_1,s_2,s_3,s_4,s_5>$ thus the initial state shown in the illustration is $s_0=<D,C,E,B,F>$ (it could also be $<F,B,E,C,D>$) Notice that the city A is not represented as it is a constraint that it must be the initial and final city.

We define the neighborhood over the swaps between two cities in the sequence, and we force the neighbor generation to follow the sequence position ordering, e.g. at the first iteration we select the first city in the sequence for swapping, at the second the second and so on (if you reach the end start over). Use the following objective function: VALUE(s)=$C_{A,s1}+C_{s1,s2}+C_{s2,s3}+C_{s3,s4}+C_{s4,s5}+C_{s5,A}$ where $C_{i,j}$ return the path distance in routing table above.

a) Simulate a hill-climbing search (used for minimization). Show the resulting sequence and all the steps including the objective values.
b) Simulate a First-choice hill climbing (RN10 p. 124) using the same neighbor generation sequence instead of random selection.

Exercise 2

Consider the following state space, and the table summarizing the objective value of each state:



State	Value
A	1
B	5
C	4
D	7
E	9
F	5
G	4
H	6
I	3
J	6
K	10

Assume *A* is the initial state. If we were to perform a hill-climbing local search (maximizing the objective), we would return the state *E* with objective value 9 taking the actions <*a, b, c*>

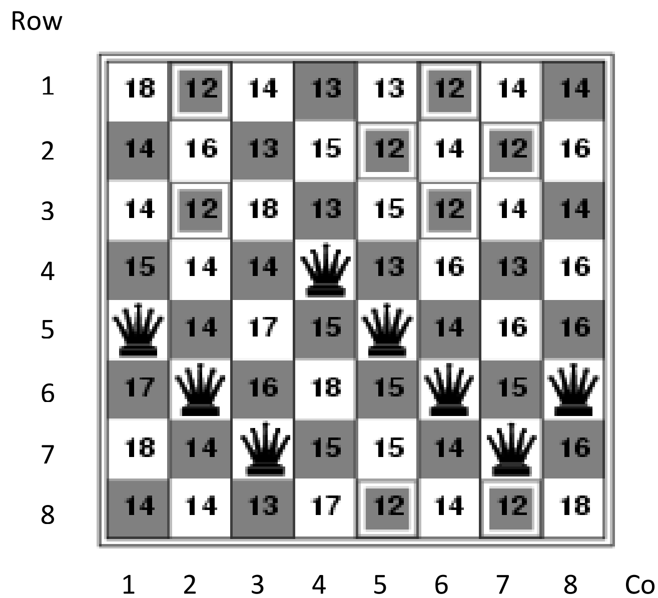
a) Assuming *A* as the initial state, perform a Tabu Search where we keep states in the tabu list. E.g. if we pass by state *G* the state is stored in the tabu list and thus we cannot go back to that state. Show the search process using the following template:

Current state	Best state	Action taken	Fringe	Tabu List
A	A	-	BC	<>
B	B	*a*	CFD	< *A* >

b) Now perform the tabu search where we keep the actions performed in the tabu list. Use the same template as for a) to show the search process.

c) Do the search performed in a) and b) return the same result? (why/why not)

Exercise 3



Consider the hill-climbing local search algorithm using the min conflict heuristic to find a valid state of the *n*-queens problem. Recall that the heuristic value of a state is the number of distinct queen pairs that can attack each other. Assume that the hill-climbing algorithm represents a state of the *n*-queens problem by a vector $a_1, a_2, \ldots, a_n$ where $a_i \in \{1..n\}$ is the row number of the queen in column *i*. The neighborhood of a state is all the states that can be reached by moving a single queen. In the figure above, the value of the min conflict heuristic has been computed for each queen. Thus, in the next state the hill-climbing algorithm will choose randomly among the states with 12 conflicts.

Consider the initial state of the 4-queens problem shown below

	Q	Q	
Q			Q

a) Draw the tree of states that hill-climbing using the min conflict heuristic explores from this initial state. In this tree, the children of a state are the set of successor states with least conflicts that the algorithm randomly chooses a next state from. Please draw the tree corresponding to all possible exploration paths of hill-climbing.

b) Are all the leaves in your tree from question a) solutions to the 4-queens problem? If not, is it possible to apply one of the extensions to hill-climbing to achieve this?

Mandatory Assignment

Some of the classic neighborhood functions used in Local Search are the 3-exchange and the 2-exchange neighborhoods. A 3-exchange neighborhood rotates the value of 3 variables between each other e.g. given the variables x_1=1, x_2=2 and x_3=3 if we apply 3-exchange(x_1,x_2,x_3) we would have as a result x_1=3, x_2=1 and x_3=2. The 2-exchange neighborhood is instead equivalent to swapping the values of two variables.

Assume you wrote a local search algorithm with a neighborhood function *F* that alternates between applying the 3-exchange and the 2-exchange neighborhood (thus a 3-exchange always must be followed by a 2-exchange and vice versa). Consider a problem with *n* decision variables x_1, x_2, …, x_n each with the domain {1,2,…,*n*}, where a valid assignment satisfies the allDiff constraint (i.e., $x_i \neq x_j$ for $i \neq j$ and $i,j \in$ {1,2,…,*n*}). Give a proof (a careful argument) that *F* is complete for this problem (i.e., that we can reach an arbitrary valid assignment from any initial valid assignment). You can assume that the number of variables is larger than 2.