Quiz 04

Question 1. The *N*-queens problem requires you to place *N* queens on an N x N chessboard such that no queen attacks another queen. (A queen attacks any piece in the same row or column or diagonal). Here are some important facts:

- The states are any configurations where **<u>all</u>** *N* queens are on the board, one per column.
- The ***moveset*** includes all possible states generated by moving a single queen to another square in the same column. The function to obtain these states is called the ***successor*** function.
- The heuristic function ***h(state)*** is the number of **attacking** pairs of queens.

a) Consider *N*=4. How many states are there in total? Explain your answer.

With N=4, the problem becomes the 4-queens problem, i.e. we need to place 4 queens on a 4x4 chessboard such that no queen attacks another queen. Because the states are any configurations where all the queens are on the board, one per column so each queen has 4 ways to put into the board. Therefore, there are 4 x 4 x 4 x 4 = 256 states in total.

b) For each state, how many successor states are there in the moveset? Explain your answer

The moveset includes all possible states generated by moving a single queen to another square in the same column. In this 4-queens problem, each queen in the column has 3 ways to move in the same column, and the board has 4 queens. Therefore, there are 4 x 3 = 12 successor states in the moveset.

c) What value will the heuristic function ***h*(*state*)** return for state *S* shown aside? Explain your answer.

state = (c1 c2 c3 c4) = (1 2 1 4)

h(state) = 3 + 2 + 0 + 0 = 5

d) Use some hill-climbing variant that can lead to a solution. Draw the search tree from S (Only draw the branches that lead to a solution; for each node on the tree, write down its ***h*( )** value).

state = (1 2 1 4), h(state) = 5

|

state = (1 3 1 4), h(state) = 2

|

state = (2 3 1 4), h(state) = 1

/ \

state = (2 4 1 4), h(state) = 1 state = (2 3 1 3), h(state) = 1

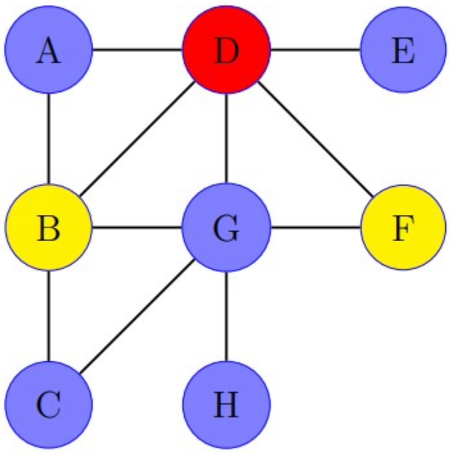
| |

state = (2 4 1 3), h(state) = 0 state = (2 4 1 3), h(state) = 0

Question 2.

Let ***G*** be the simple graph shown below. The problem is to find a coloring of each vertex V using colors **red**, **blue**, and **yellow**, so that no two adjacent vertices are assigned the same color.

We model the problem with the set of variables $x_a, x_b, \ldots, x_g$, where, e.g., x_a denotes the color assigned to vertex ***a***



- Define the state space associated with this model.

The state space associated with this model is defined by all possible assignments of colour to the vertices of the graph. Each variables x_A, x_B, ..., x_G, x_H represents the colour assigned to the corresponding vertex A, B, ..., G, H. The possible colours are red, blue and yellow.

- How big is this space?

There are 8 vertices in total, and each vertex has 3 options of colours. Therefore, there are 3^8=6561 possible states in this space.

- Give an example of a solution state.

An example of a solution state is:

{x_A = blue, x_B = yellow, x_C = red, x_D = red, x_E = yellow, x_F = yellow, x_G = blue, x_H = yellow}

- For an arbitrary state ***s***, define a "reasonable" neighborhood function ν(s) for s. Using these neighborhoods, provide a local path from the coloring shown below to your aforementioned solution state.

- Neighborhood function v(s) for s:

For an arbitrary state s, all states can be obtains by changing colour of a single vertex, i.e. choosing one variable xi and assigning it a different colour while keeping the colour of all other vertices unchanged. Then, we apply this neighborhood function to find a neighboring state with fewer conflicts and repeat the process until the solution state is reached.

- A local path:

Assume that all unassigned nodes is blue, then we have the initial state as below:

{**x_A = blue**, x_B = yellow, **x_C = blue**, x_D = red, **x_E = blue**, x_F = yellow, **x_G = blue**, x_H = **blue**}

{x_A = blue, x_B = yellow, **x_C = red**, x_D = red, x_E = blue, x_F = yellow, x_G = blue, x_H = blue}

{x_A = blue, x_B = yellow, x_C = red, x_D = red, x_E = blue, x_F = yellow, x_G = blue, **x_H = yellow**}

Question 3.

Consider the 4-queens problem, in which each state has 4 queens, one per column, on the board. The state can be represented in genetic algorithm as a sequence of 4 digits, each of which denotes the position of a queen in its own column (from 1 to 4).

- $\boldsymbol{Fit}(\boldsymbol{n})$ = *the number of non-attacking pairs of queens* • Let the current generation includes 4 states:

  S1 = 2341;   S2 = 2132;   S3 = 1232;   S4 = 4321.
- Calculate the value of $\boldsymbol{Fit}(\boldsymbol{n})$ for the given states and the probability that each of them will be chosen in the "selection" step.

The value of Fit(n) for the given states are:

$Fit(S_1) = 2$

$Fit(S_2) = 3$

$Fit(S_3) = 1$

$Fit(S_4) = 0$

⇨ $Fit(S_1) + Fit(S_2) + Fit(S_3) + Fit(S_4) = 2 + 3 + 1 + 0 = 6$

The probability for each given state will be chosen in the selection step are:

$P(S_i) = Fit(S_i)/ [Fit(S_1) + Fit(S_2) + Fit(S_3) + Fit(S_4)]$

$P(S_1) = 2/6 = 1/3 = 0.33$

$P(S_2) = 3/6 = ½ = 0.50$

$P(S_3) = 1/6 = 0.16$

$P(S_4) = 0/6 = 0.00$