**Question 1. The N-queens problem requires you to place N queens on an N × 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 all 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.

For the N-queens problem with N=4, we need to place 4 queens on a 4x4 chessboard. The total number of states is obtained by considering all possible configurations where each queen is in a different column. For the first queen, we have 4 choices (4 rows), for the second queen, we have 4 choices (excluding the row of the first queen), and so on. Therefore, the total number of states is 4 \* 4 \* 4 \* 4 = 256.

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

To generate a successor, we move one of the Queens to a new square (within its column). There are 4 Queens to choose from and 3 new positions available in its respective column. This makes the number of successors 3\*4 = 12 successors.

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



The heuristic function h(state) equals to the number of pairs of queens that are attacking each other,

either directly or indirectly.  
Queen at column 1 attacks queen of column 2 and 3 directly, attacks queen of column 4 indirectly. h =3

Queen at column 1 attacks queen of column 3 and 4 directly. h = 2

Therefore, the total h value of this state is 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).

Taking the state of above figure for an example

We have S (h=5), we decide to use first choice hill climbing algorithm to solve the problem

In this problem it will randomly generate the successor states where it could lead to solution

Moving queen of column 1: it will randomly generate state to <4214> h(3), <2214> h(3)

Moving queen on column 2 from successors of queen 1:

From <2214> h(3) I can generate to <2314> h(1) and <2414> h(1)

Moving queen on column 3 from successors of queen 2:

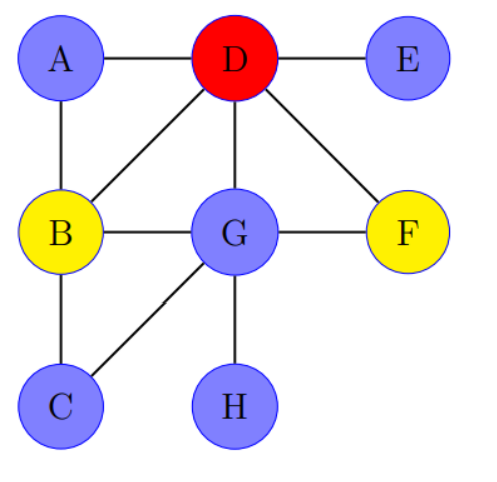
From <2414> h(1) is the most optimal

Moving queen 4 from successors of queen 3:

From <2414> h(1) generate to <2413> h(0)

The search tree is <1214> h(5) => <2214> h(3) => <2414> h(1) => (2413) h(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 xa, xb, . . . , xg, where, e.g., xa denotes the color assigned to vertex a**

****

**• Define the state space associated with this model.**

The state space associated with this model consists of all possible combinations of colors assigned to each vertex. Each variable x represents the color of a vertex V, and there are three possible colors: red, blue, and yellow. Therefore, the state space can be defined as follows:

State Space = {(xa,xb,xc,xd,xe,xf,xg,xh) ∣ xi∈{’red’,’blue’,’yellow’} for i∈{a,b,c,d,e,f,g,h}}

**• How big is this space?**

Considering there are 3 colors (red, blue, yellow) for each vertex, and there are 8 vertices in total, the size of the state space is 38 since each vertex can be assigned one of the 3 colors.

**• Give an example of a solution state.**

An example of a solution state is where each vertex is assigned a color such that no two adjacent vertices have the same color. For instance using the graph provided:   
Vertices (V): {A, B, C, D, E, F, G, H}

Edges (E): {(A, B), (A, D), (B, C), (B, D), (B,G), (C,G), (D, E), (D,F), (D,G), (F,G), (G,H )}

An example solution:   
<xa = blue, xb = yellow, xc = red, xd = red, xe = blue, xf = yellow, xg = blue, xh = red >

A = blue adjacent to B = yellow and D = red

B = yellow adjacent to C = red, D = red and G = blue

C = red adjacent to G = blue

D = red adjacent to E = blue, F =yellow and G = blue

F = yellow adjacent to G = blue

G = blue adjacent to H =red

(We do not check the backward because it is already true)

**• 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.**The neighborhood function ν(s) typically refers to the set of states that are adjacent or neighboring to a given state s in the state space. Each state in the graph coloring problem represents a possible assignment of colors to vertices in a graph, and the goal is to find a valid coloring where no adjacent vertices share the same color.

Let's denote a state as s, where s=(xa​,xb​,..,xi,...,xg​), and xi​ represents the color assigned to vertex i.

ν(s) = {s′ ∣ s′=(x1​,x2​,...,xi′​,...,xg​) where ci′​!=ci​}

From the coloring shown in graph:

v(s) = <xa = blue, xb = yellow, xc = blue, xd = red, xe = blue, xf = yellow, xg = blue, xh = blue >

The conflict adjacent vertices are xg = blue to xc = blue and xh = blue

Change the color of vertex xc to red:

v(s) = <xa = blue, xb = yellow, xc = red, xd = red, xe = blue, xf = yellow, xg = blue, xh = blue >

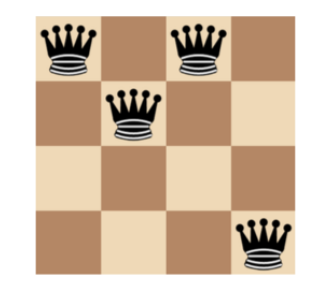
Because xb is yellow we can’t chage to xc to yellow. (B is adjacent to C)

Change the color of vertex xh to red:

v(s) = <xa = blue, xb = yellow, xc = red, xd = red, xe = blue, xf = yellow, xg = blue, xh = red >

The local path is xc from blue to red and xh from blue to red.

**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).**

****

**• 𝑭𝒊𝒕(𝒏) = the number of non-attacking pairs of queens**

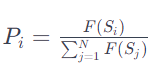
**• Let the current generation includes 4 states: S1 = 2341; S2 = 2132; S3 = 1232; S4 = 4321.**

**• Calculate the value of 𝑭𝒊𝒕(𝒏) for the given states and the probability that each of them will be chosen in the “selection” step.**

A non attacking pair is when two queens don't attack each other.For max condition no queen attacks any other queen, so number of non attacking pairs:

Every queen can have 3 non-attacking queens. 4 queens will have overall 4x3=12 non-attacking queens. Thus, total number of non attacking pairs for 4 queens would be = 4 x 3 /2 = 6

1. S1=2341
   * Attacking pairs: (2,3), (2,4), (3,4), (3,1)
   * Fitness function: F(S1) = 6 - 4 = 2
2. S2=2132
   * Attacking pairs: (2,1), (2,2), (3,2)
   * Fitness function: F(S2) = 6 – 3 = 3
3. S3=1232
   * Non-attacking pairs: (1,2), (1,3), (2,2), (2,3)
   * Fitness function: F(S3) = 6 - 4 = 2
4. S4=4321
   * Non-attacking pairs: (4,3), (4,2), (4,1), (3,2), (3,1), (2,1)
   * Fitness function: F(S4) = 6 – 6 = 0

Now, to calculate the selection probability for each state, you can use a formula like proportional selection. The probability Pi​ for state Si​ is given by:

P1 = = = 29%

P2 = = = 42%

P3 = = = 29%

P4 = = = 0%