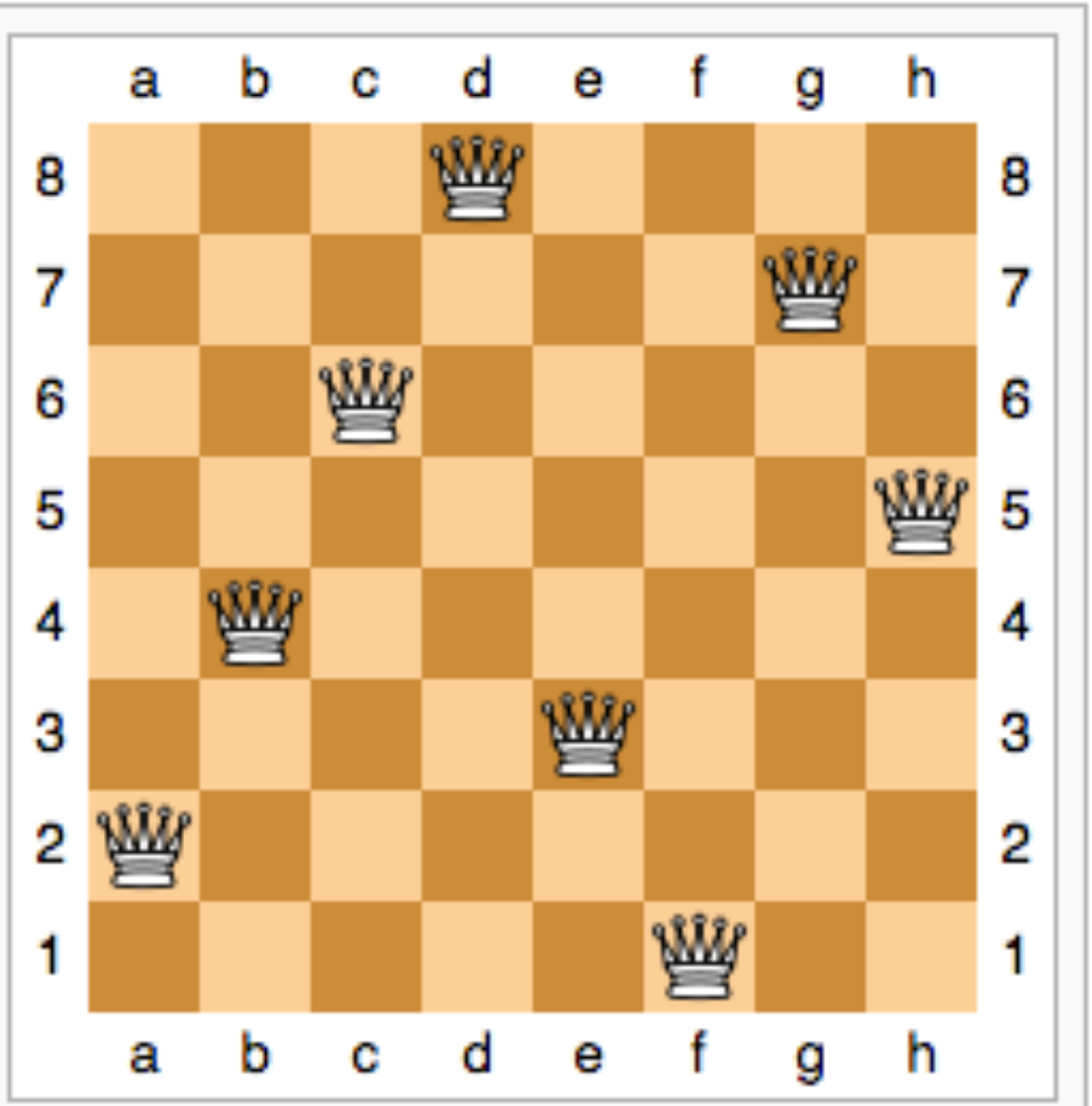
Problems recursion makes easier!

(or possible)

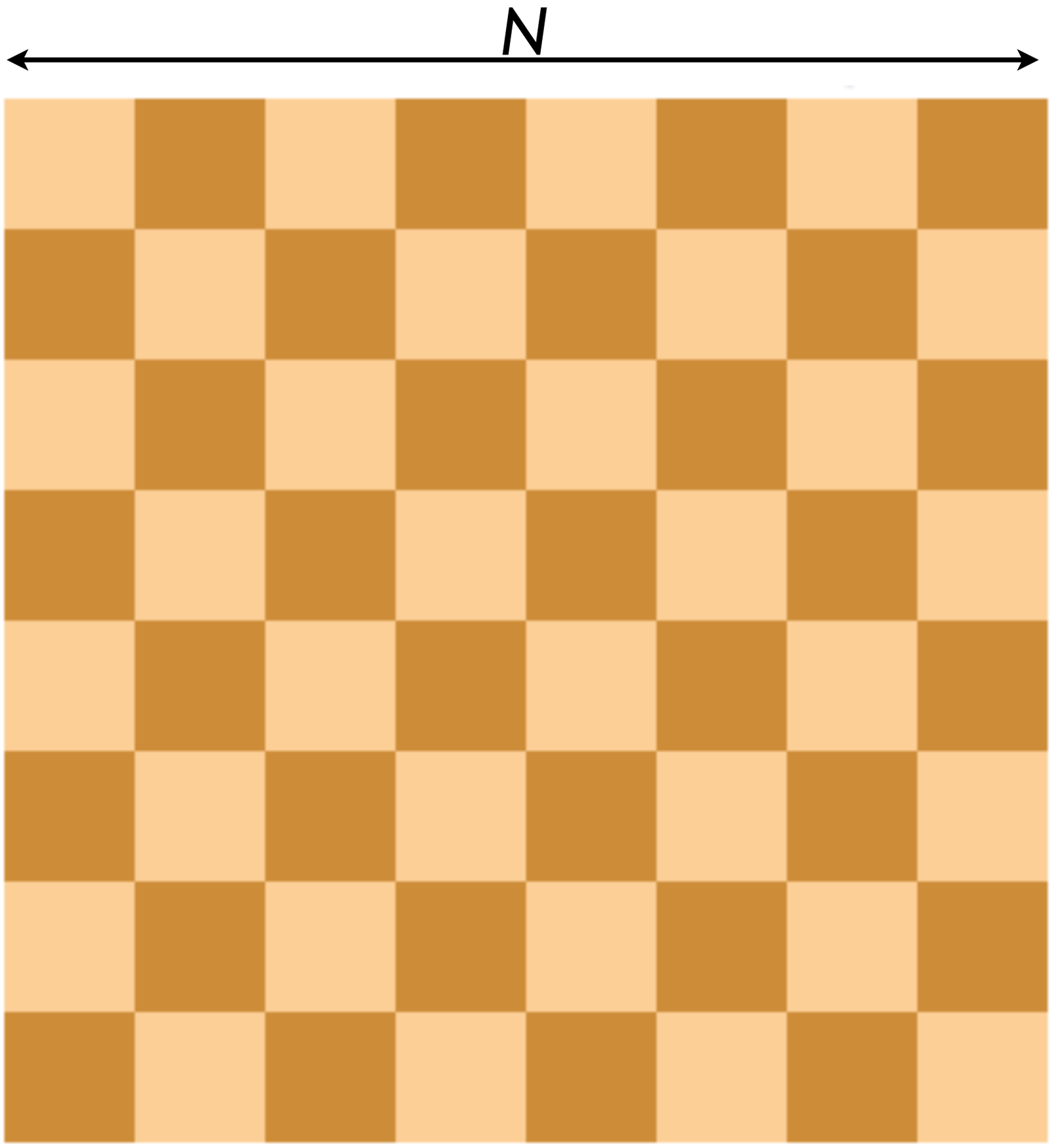


One solution to the eight queens puzzle

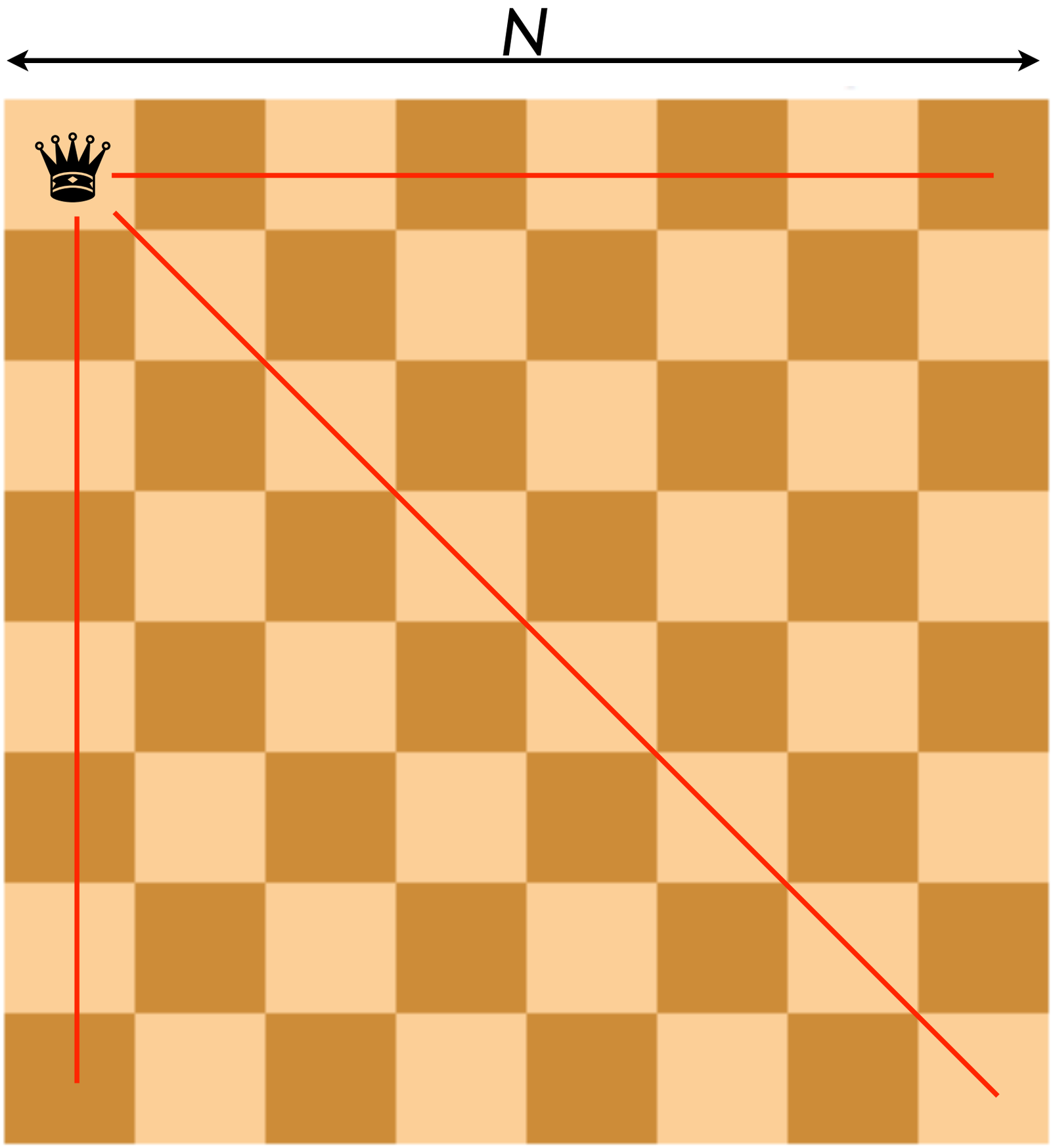
5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Thanks for the pictures, Wikipedia!

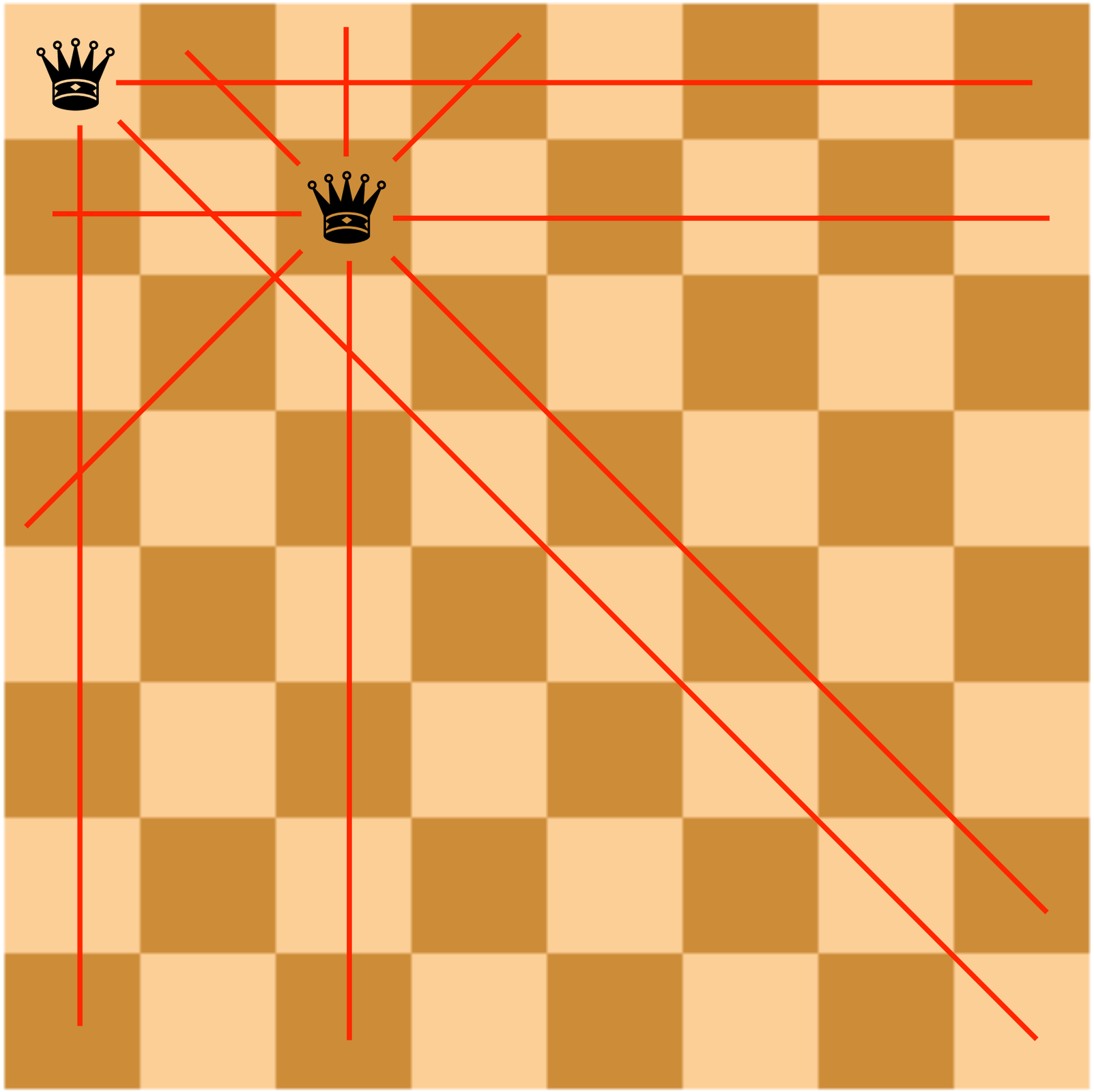
N Queens



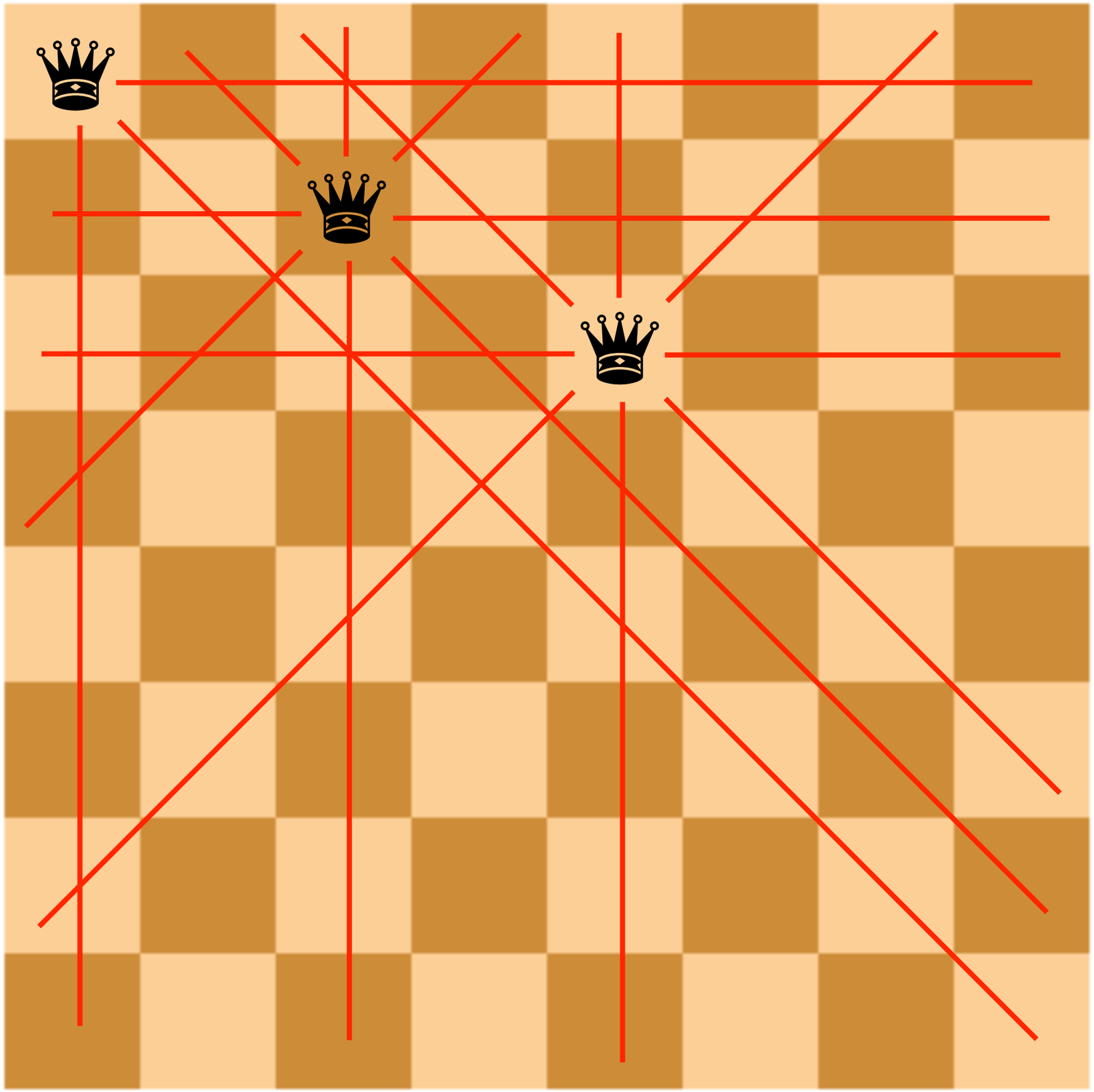
N Queens



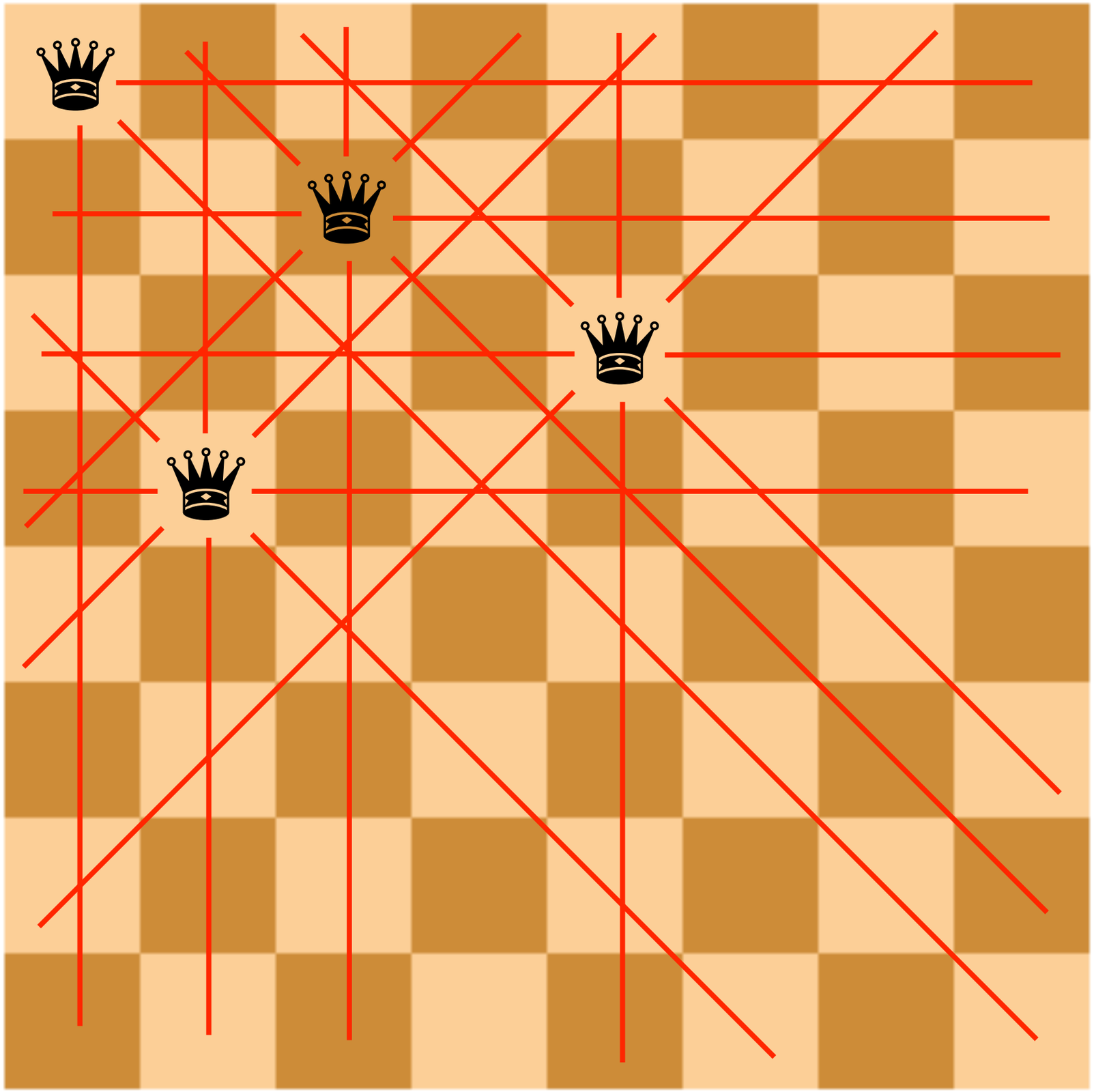
N Queens



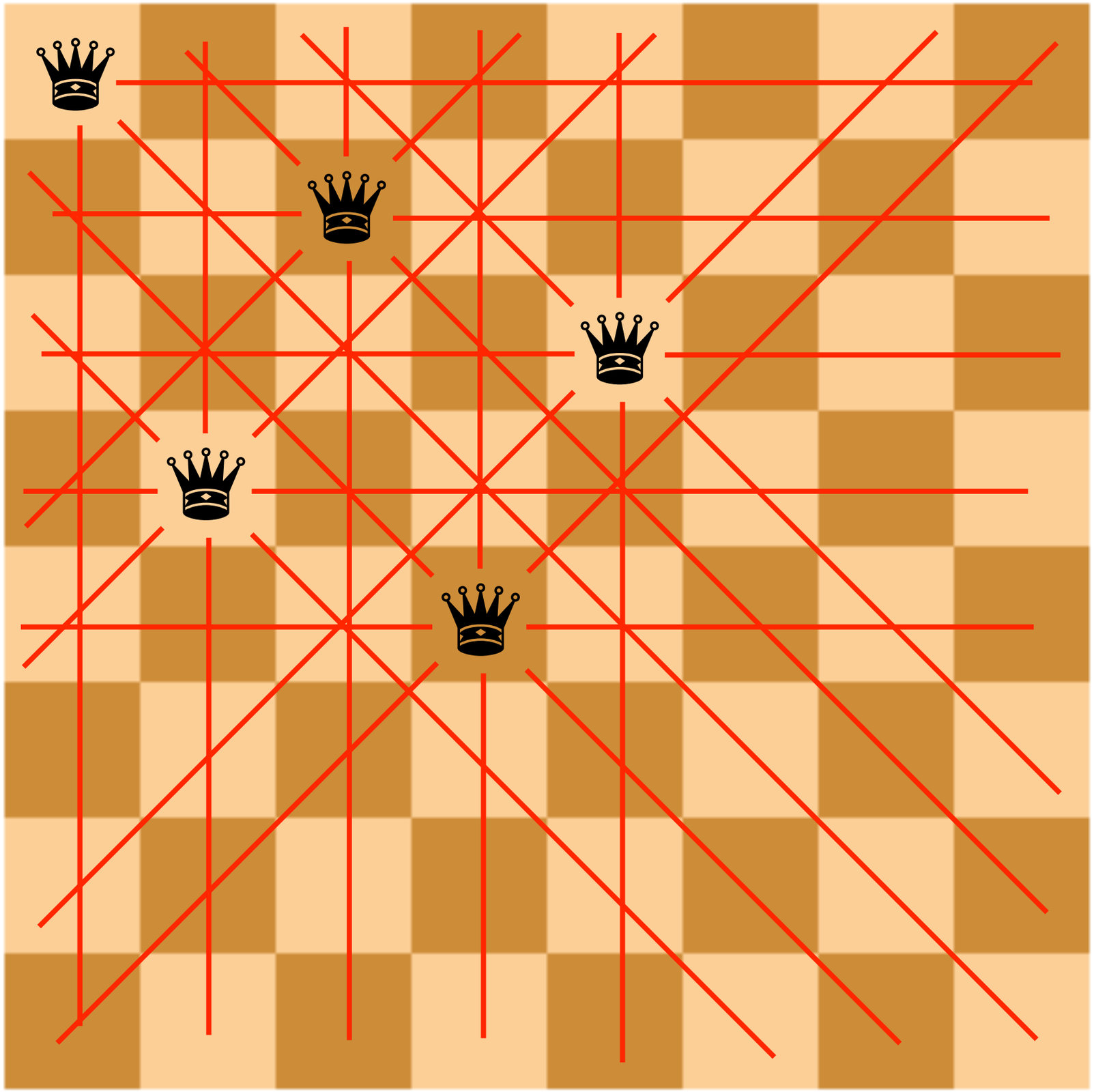
N Queens



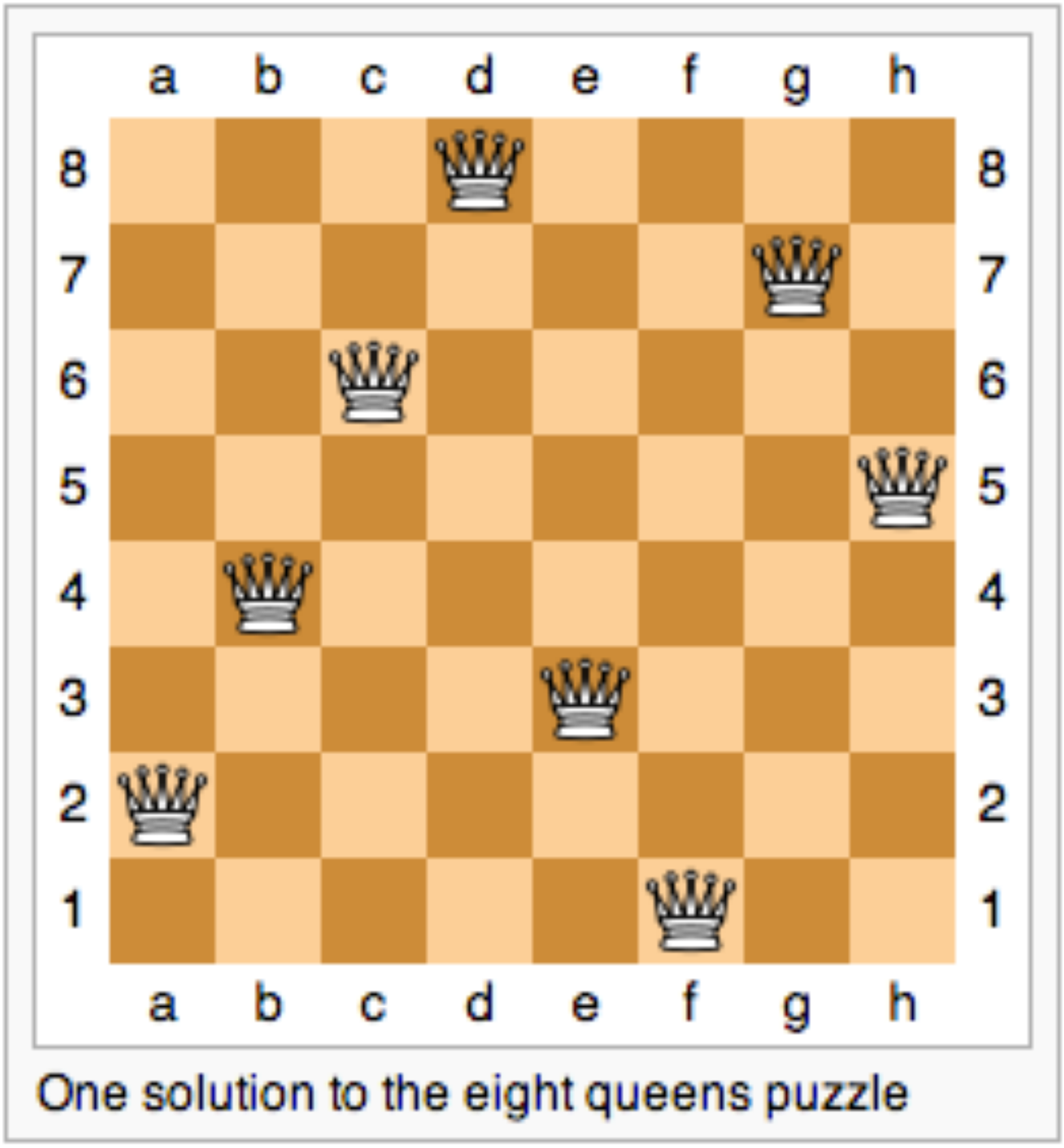
N Queens



N Queens

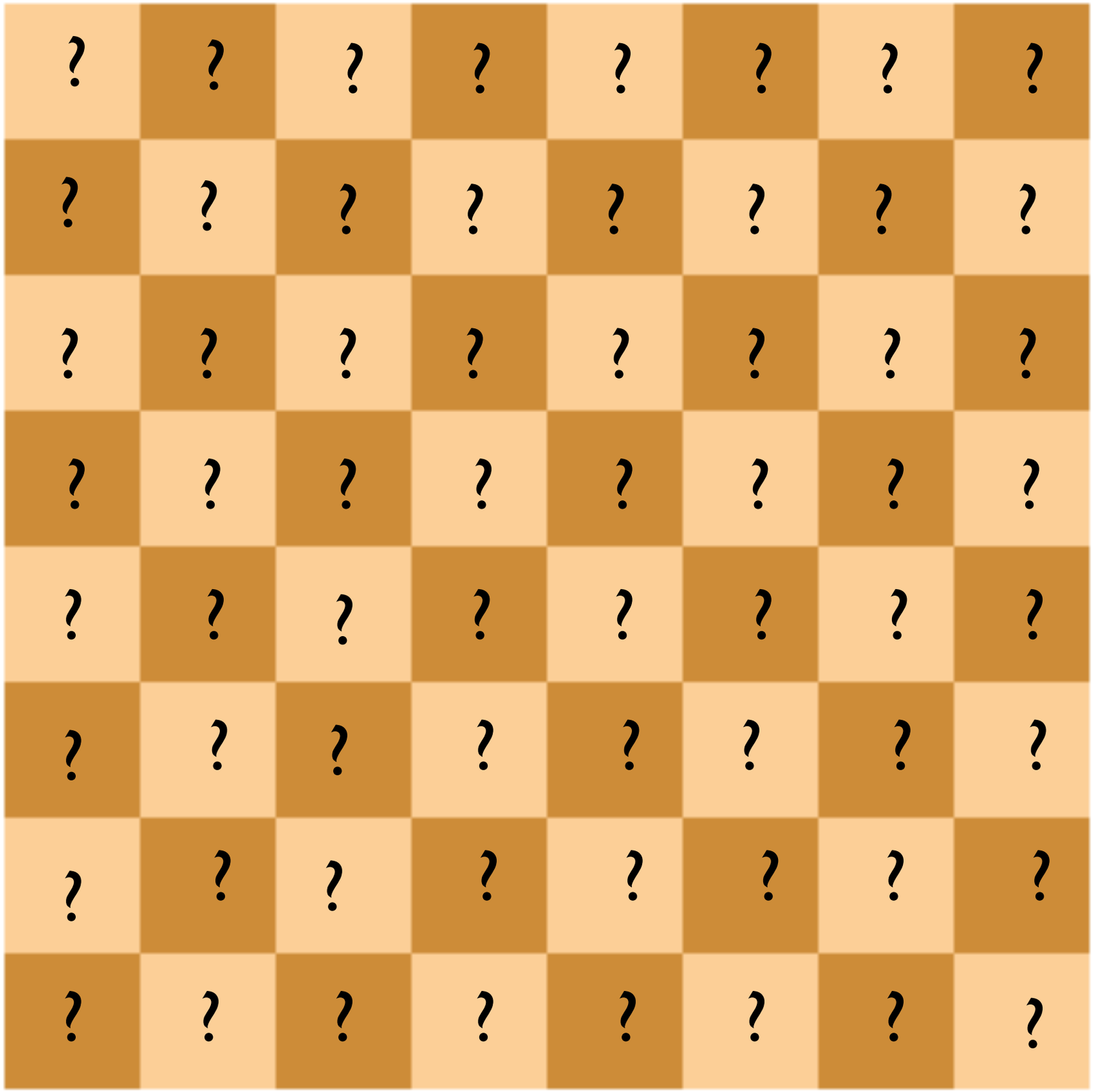


N Queens

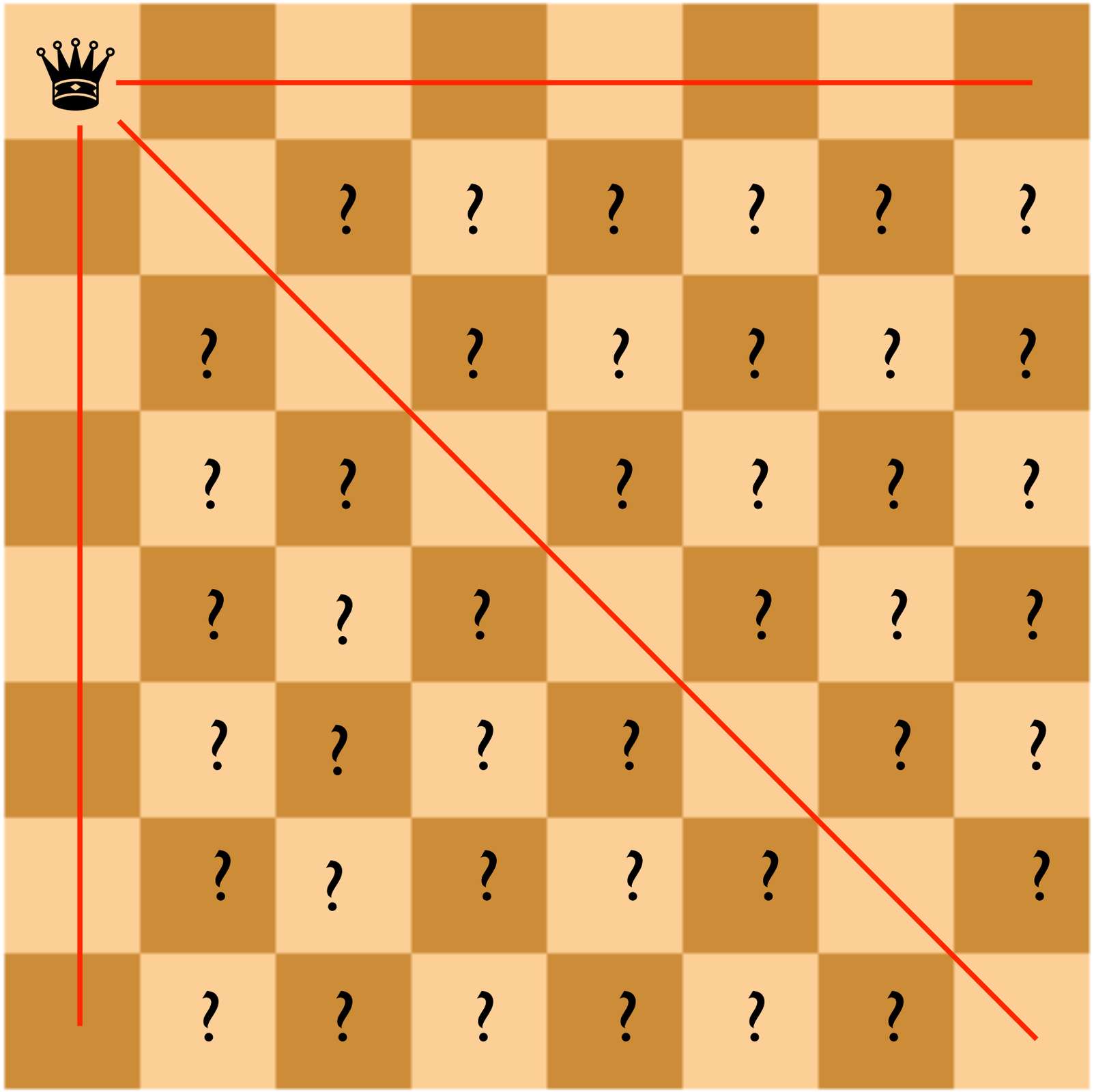


One solution to the eight queens puzzle

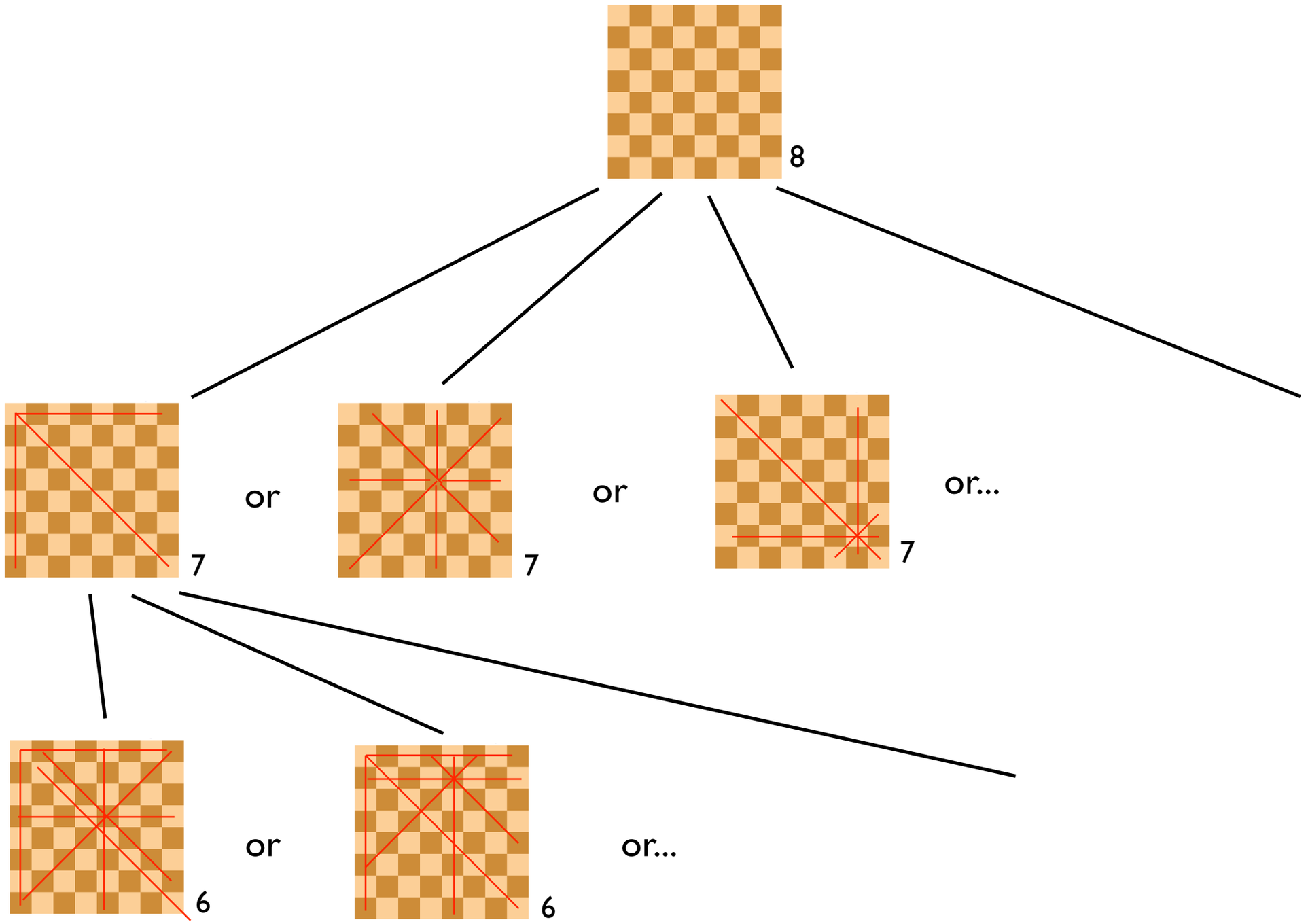
Recursive Backtracking



Recursive Backtracking



Recursive Backtracking



Recursive Backtracking

Base Case?

Remember the human algorithm for writing recursive algorithms!

Recursive Backtracking

Base Case?

Zero Queens

Recursive step?

Remember the human algorithm for writing recursive algorithms!

Recursive Backtracking

Base Case?

Zero Queens

Recursive step?

For every free space:

- Copy the board & add a queen there

- Recurse on n-1 queens

- If that worked:

* - return true*

- else:

* - return false*

Remember the human algorithm for writing recursive algorithms!

Helper Functions

```
public static void main(String[] args) {
   NQueensBoard b = nQueens(8);
   if (b != null) {
      System.out.println(b);
   } else {
      System.out.println("No solution!");
   }
}
```

Helper Functions

```
    public static void main(String[] args) {
        NQueensBoard b = nQueens(8);
        if (b != null) {
            System.out.println(b);
        } else {
            System.out.println("No solution!");
        }
    }


public static NQueensBoard nQueensHelper(NQueensBoard b,
                                         int n) {
    // code!
}
```

Helper functions should take everything you could possibly need *as arguments.*

An addendum to the human algorithm!

Helper Functions

```
    public static void main(String[] args) {
       NQueensBoard b = nQueens(8);
       if (b != null) {
          System.out.println(b);
       } else {
          System.out.println("No solution!");
       }
    }


public static NQueensBoard nQueensHelper(NQueensBoard b,
                                         int n) {
   // code!
}


public static NQueensBoard nQueens(int n) {
   NQueensBoard b = new NQueensBoard(n);
   return nQueensHelper(b, n);
}
```

Helper functions should take everything you could possibly need *as arguments.*

An addendum to the human algorithm!

SpreadingNews

Is hard!

Here's a hint.

Suppose you know how long all of your subordinates take to do their work. You will always call the slowest person first, the second-slowest person second, and so on.

See also: Collections.sort & Collections.reverse.

RatRoute

Base Case?

```
. R . . .
. . X . .
. . . . X
X . X . X
. . . C .
```

RatRoute

Base Case?

Recursive step?

```
. R . . .
. . X . .
. . . . X
X . X . X
. . . C .
```