Algorithms and their Applications CS2004 (2020-2021)

Dr Mahir Arzoky

17.1 Further Evolutionary Computation

CodeRunner Class Tests and Laboratory Sessions…

- All four class tests must be passed to pass Task #1.
- Task #1 weighs 30% of the coursework
- If you do not pass Task #1 you will be capped at D- grade (coursework).
- Class tests needs to be completed by 16/02/2021

In This Lecture

- We are going to look at:
  - An application of Genetic Algorithms
  - Evolutionary Programming
  - Genetic Programming

Recap – Genetic Algorithms

- **Genetic Algorithms** (GA) are a powerful tool
- Based on the biological theory of evolution
- The correct use is down to experience
- The GA involves the iterative application of a number of **genetic operators**
- There are a number of parameters that need to be **carefully** selected

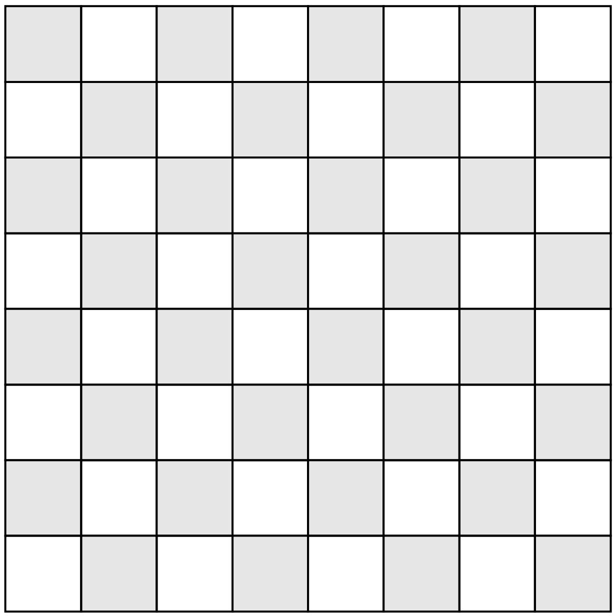
Recap - Holland's Algorithm

- Initial population
  - Create an initial population of random solutions to the problem
- Crossover
  - Pair up those allowed to breed and recombine genes from the parents to produce new children
- Mutation
  - Mutate some of the genes of some of the chromosomes
- Kill off invalid chromosomes
  - Removed invalid solutions from the population
- Survival of the fittest
  - Select the fittest to survive to the next generation

Recap - Holland's Algorithm

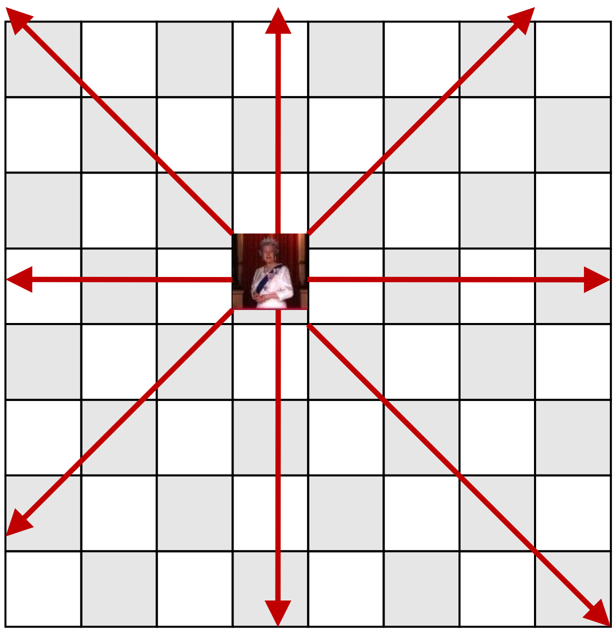
```
Input: The GA parameters: NG, PS, CP, MP and n
       The Fitness Function
1) Generate PS random Chromosomes of length n
2) For i = 1 to NG
3)    Crossover Population, with chance CP per Chromosome
4)    Mutate all the Population, with chance MP per gene
5)    Kill off all Invalid Chromosomes
6)    Survival of Fittest, e.g. Roulette Wheel
7) End For
Output: The best solution to the problem is the Chromosome
        in the last generation (the NGth population) which
        has the best fitness value
```

The Eight Queens (EQ) – Part 1



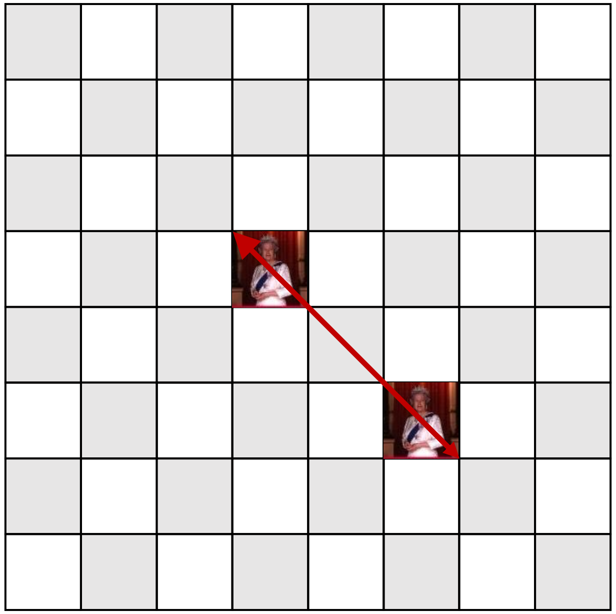
The problem is to place the eight Queens on a chess board so that none of the Queens are attacking each other

The Eight Queens (EQ) – Part 2



Queens can attack in straight lines and diagonal lines

In the example below the Queens are attacking each other



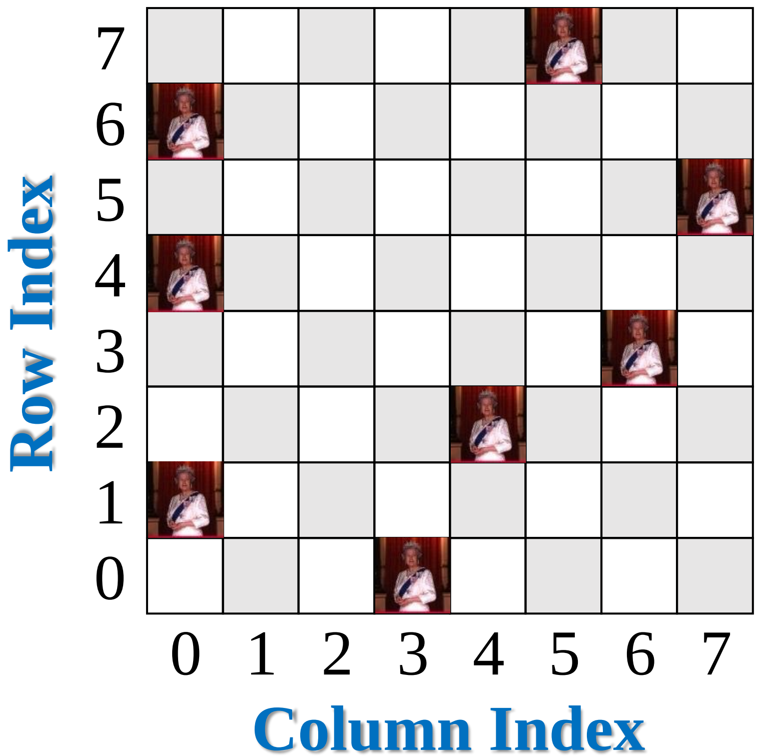
The Eight Queens (EQ) – Part 3

- In order to solve this problem using a **Genetic Algorithm** we need the following:
  - A representation
  - A fitness function
  - Parameter values

EQ Representation – Part 1

- Only one queen on each row
  - A reasonable assumption that reduces the **size** of the search space
- 8 sets of 3 bits
  - 0-7 for the column number the Queen occupies
- Total of 24 bits
- Each set of 3 bits represents the row and then the position on that row
- There are **no** invalid chromosomes

EQ Representation – Part 2



Representation Example

011000100110000111000101

Row 0 = 011 = 3
Row 1 = 000 = 0
Row 2 = 100 = 4
Row 3 = 110 = 6
Row 4 = 000 = 0
Row 5 = 111 = 7
Row 6 = 000 = 0
Row 7 = 101 = 5

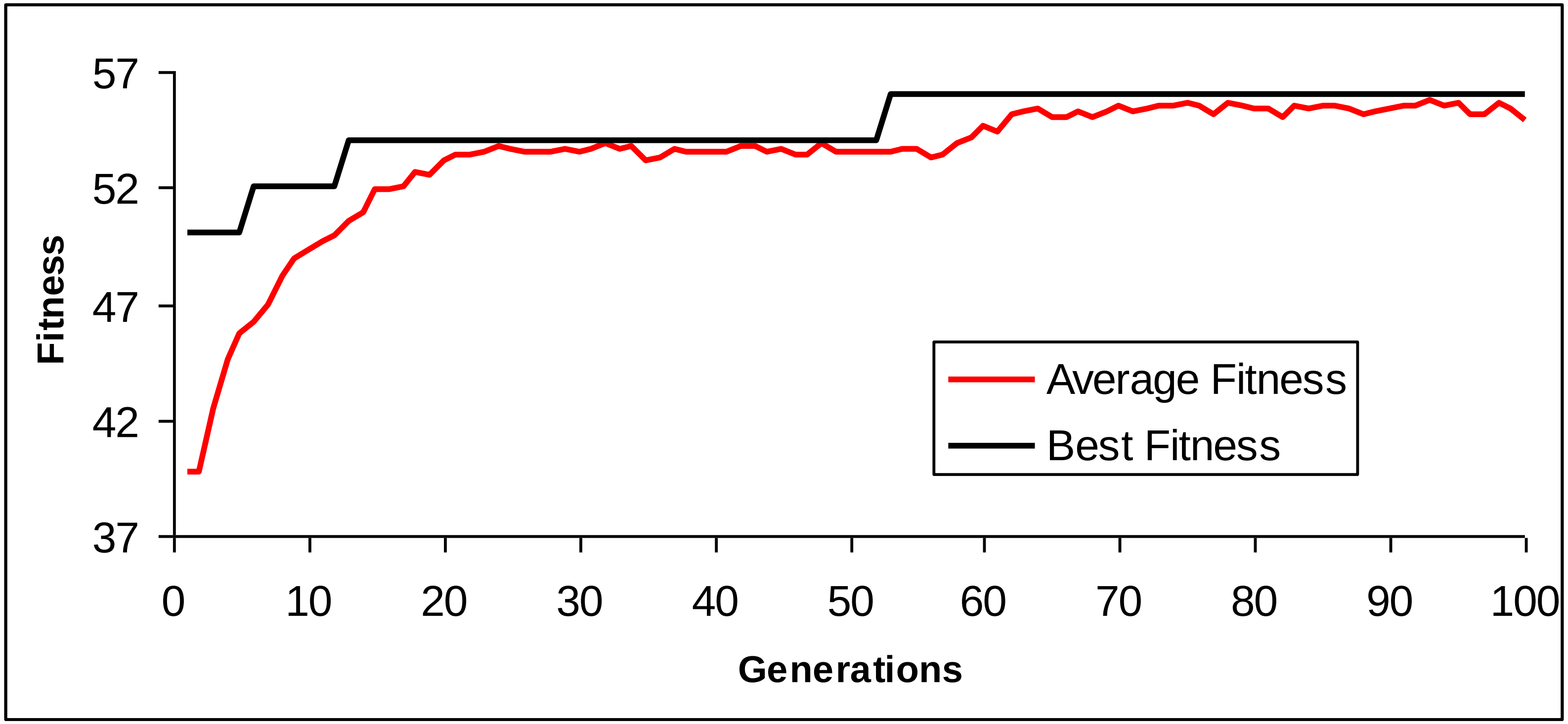
EQ Fitness Function

- The number of clashes
- This is the total number of attacks
- Maximum number of attacks is 56
  - $8 \times 7 = 56$
- Therefore the fitness function becomes 56 minus the number of attacks
  - For a maximisation problem

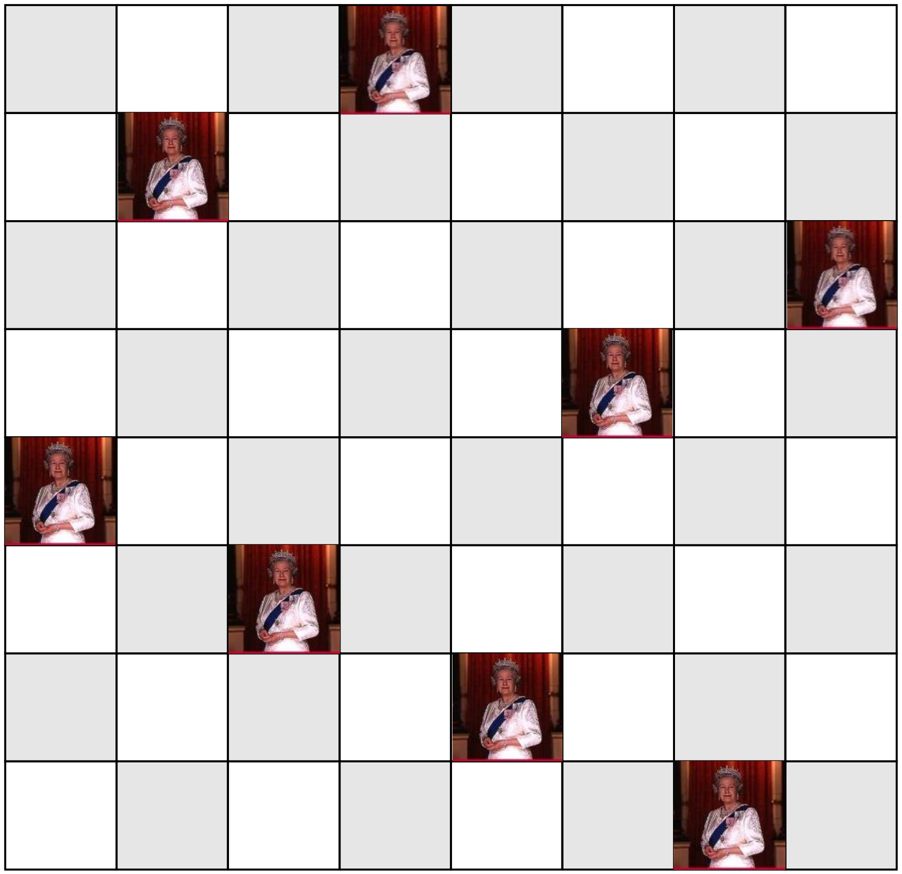
EQ Parameters

- NG Number of Generations 100 (Small)
- PS Population Size 100 (Small)
- CP Crossover Probability 75% (Guess)
- MP Mutation Probability 4% (~1/24)
- 5 Chromosome Elitism (5% of pop.)
  - Fittest solution selected

EQ Results – Part 1



EQ Results – Part 2



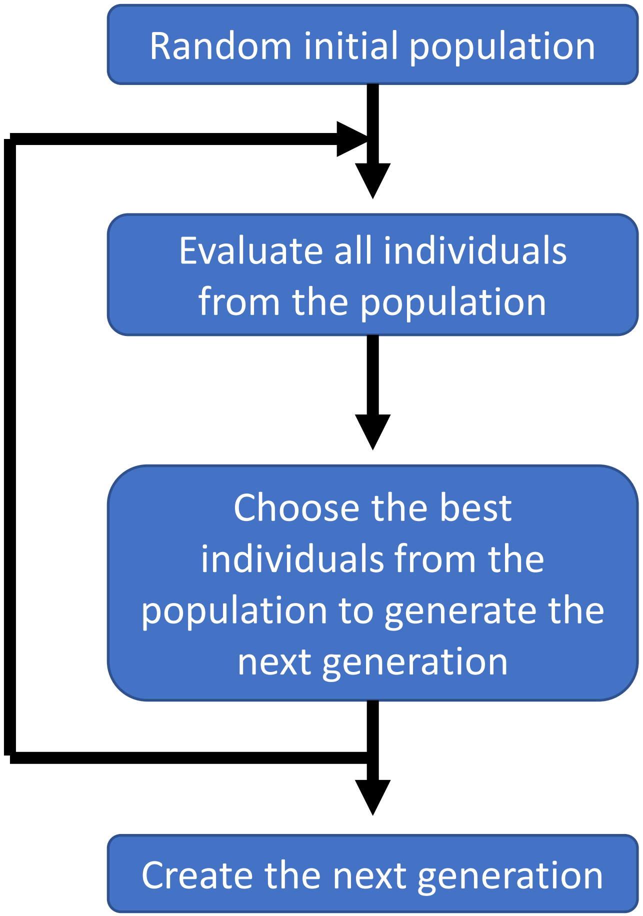
- 92 solutions
- 12 fundamental solutions

EQ Conclusion

- Not the best method but it works!
- Other methods such as using a permutation based representation would be better
  - $2^{24} = 16,777,216$
  - $8! = 40,320$
- N-Queens
- Pieces other than queens!
  - 16 kings, 32 knights, 14 bishops, 8 rooks

Evolutionary Computation – Introduction

- We have looked at the Genetic Algorithms (GA)
- GAs belong to family of techniques that are inspired from evolution theory
- They are typically aimed at solving optimisation problems
- They can be used to generate Artificial Intelligence e.g. in games and robots



Evolutionary Computation – Introduction

- These methods come under the field of Evolutionary Computation
  - Evolutionary Programming (covered)
  - Genetic Programming (covered)
  - Evolutionary Strategies (<u>not</u> covered)
  - Learning Classifier Systems (<u>not</u> covered)
  - Estimation of Distribution Algorithms (<u>not</u> covered)

Evolutionary Programming – Part 1

- Evolutionary Programming (EP) is a similar approach to that of Genetic Algorithms
- The emphasis is on **mutation** and there is no **crossover**
- Every individual mutates, doubling the population
- The mutation operators tend to be complex and/or adaptive
- The selection/survival operator tends to be **Tournament Selection**

Evolutionary Programming – Part 2

- With **Tournament selection**, each member of the population is compared with a fixed number of other individuals
- For each comparison, the individual is awarded a point if it’s fitness is better than the opponent
- The population is reduced back to its original size by retaining those with the highest score

Evolutionary Programming – Part 3

- The EP algorithm is as follows:

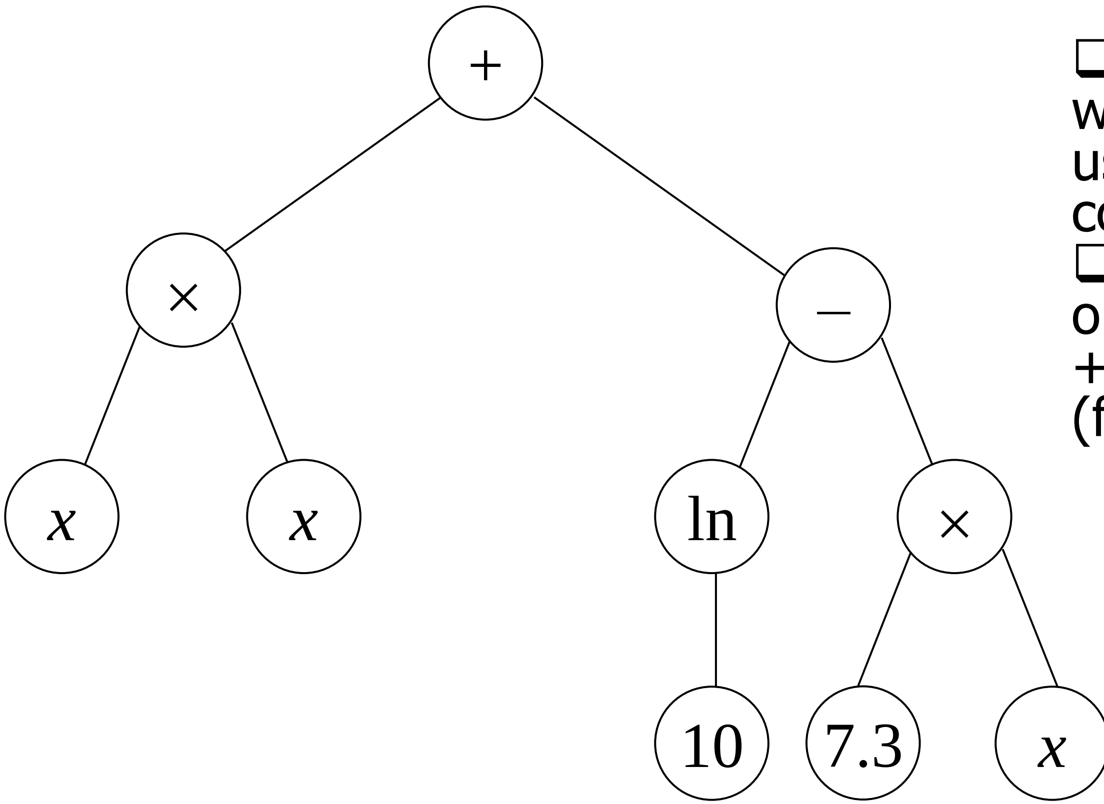
```
Input: Population size, number of generations
       and Fitness Function
1) Create the initial population
2) For i = 1 to number of generations
3)   Mutate the population
4)   Apply Tournament Selection
5) End For
Output: Return the best individual
```

Genetic Programming – Part 1

- **Genetic Programming (GP)** is an evolutionary approach that extends Genetic Algorithms
- We evolve computer programs by Natural Selection
- In this lecture series, the technique that is of interest is **Symbolic Regression**, which is a type of **Genetic Programming**
- With symbolic regression, a mathematical expression is represented as a tree structure

Genetic Programming – Part 3

- Consider the expression: $x^2 + \ln(10) - 7.3x$, then the tree representing this is:



- Terminal nodes within this tree are usually variables or constants
- Non terminals are operators (for example +,-,/,×) or functions (for example: logarithm)

Genetic Programming – Part 4

- Initially determine the set of terminals and functions
- The fitness of such a tree is a function of the observed data versus the calculated data resulting from evaluating the expression the tree represents
- Crossover and mutation are redesigned to handle tree structures
- The genetic programming algorithm is virtually the same as the genetic algorithm

Genetic Programming – Part 5

GP Operator	Description
Sub-tree Exchange Crossover	Two sub-trees are swapped between parents
Self Crossover	Sub-trees are exchanged within an individual parent
Point Mutation	A node in the tree is changed to a different symbol
Permutation Mutation	Two terminal symbols from the same sub-tree are swapped
Hoist Mutation	A sub-tree creates a new individual
Expansion Mutation	A sub-tree is added to the base of a tree
Prune Mutation	A sub-tree is removed
Sub-tree Mutation	A sub-tree is replaced for a random sub-tree

Some Genetic Programming Operators

Next Lecture

- We will look into Ant Colony Optimisation and Particle Swarm Optimisation

Next Laboratory

- The laboratory will involve..
  - Modifying Laboratory 11's simple Genetic Algorithm (GA) to solve a new problem
  - Running a number of experiments and plotting convergence graphs