COSC343: Artificial Intelligence

Lecture 14 : Genetic algorithms

Lech Szymanski

Dept. of Computer Science, University of Otago

In today's lecture

- Biological evolution
- Genetic algorithms – optimisation algorithms inspired by biology

Biological fitness

The natural world is full of optimisation processes

In each species, there's some definition of **fitness** for each organism *o*.

- How likely is it that *o* will survive long enough to reproduce?

This depends on different things for different species...but in general

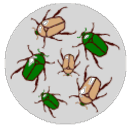
- *o* needs to get food/energy...
- *o* needs to avoid being killed...
- *o* needs to find a mate...

Genetic diversity

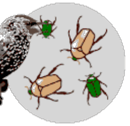
Within any population *P* of a species, there is a **variation**.

- Each individual in *P* is built from its **genes**.
- When individuals reproduce, their genes are not perfectly copied
  - There can be mutations...
  - In sexual reproduction, offspring are created by rcombining genes from two individuals
- So there's **genetic diversity** in *P*...and hence variation.
- The individuals in *P* are at different positions in the fitness landscape.

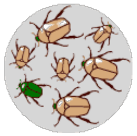
Evolution by natural selection



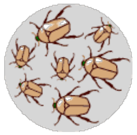
Within any population *P*, there's an algorithm that *optimises* fitness: **natural selection**. This is a variety of stochastic search (random walk optimisation).



- The state space is **the space of genetic variation within *P***.
- The fitness landscape is defined by **the fitness function for *P***.
- If *P* contains *k* individuals, the search involves ***k* current states**.



- The process of selecting the best current states is implemented by **the world**: the fitter individuals are more likely to reproduce.



- The process of finding *neighbouring states* is implemented by **biological reproduction**:
  - offspring may contain *mutations*...
  - offspring may *recombine* their parent's genes.

Evolution by natural selection

Biological evolution is an algorithm, implemented by the world!

We can implement similar algorithms to solve optimisation problems in computers...these algorithms are called **genetic algorithms**.

Genetic algorithms (GA): choose a representation

In a genetic algorithm we define a population of **individuals**, where each individual's position in state space is encoded by a **chromosome**.

8	6	4	2	7	5	3	1

A chromosome is a sequence of characters from some **alphabet**.

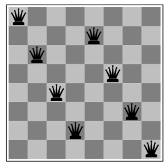
- The alphabet could be a set of numbers, characters...
- Each chromosome encodes a possible solution to the optimisation problem (i.e. a point in the state space).

GA: choose a fitness function

Consider a well-known optimisation problem: the **8 queens problem**.

- Position 8 queens on a chessboard so that no queen can take any other queen
- Here's a possible chromosome and its corresponding board position

8	6	4	2	7	5	3	1



- As a fitness function, we can take the number of pairs of queens that can't take one another.

GA: the algorithm

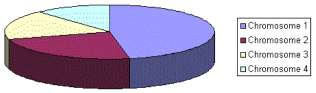
1. Create a population of individuals with random chromosomes.
2. Evaluate the fitness of each member of the population.
3. Pick two **parents** randomly, as a function of their fitness (fitter individuals more likely to be picked)
4. Create a new individual by mixing the parents' chromosomes

Two issues:

- How to **pick** the parents...
- How to **mix** their chromosomes...

GA: roulette wheel selection

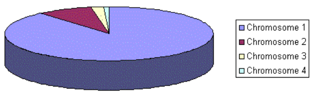
A common method is **roulette wheel selection**.



- Sum the fitness of all individuals in the population
- Normalise fitness to sum to 1
- Express fitness of each individual as a range in the interval [0,1].
- Choose a random number from uniform distribution in the interval [0,1]
- Pick the corresponding individual

GA: roulette wheel selection

A common method is **roulette wheel selection**.



- Sum the fitness of all individuals in the population
- Normalise fitness to sum to 1
- Express fitness of each individual as a range in the interval [0,1].
- Choose a random number from uniform distribution in the interval [0,1]
- Pick the corresponding individual
- Very unfit individuals are hardly ever picked

GA: tournament selection

Another method is **tournament selection**.

- Pick a subset of n individuals from the population at random
- Two individuals with the highest fitness are selected as the two parents.

The size of n controls the chance that unfit individuals are selected.

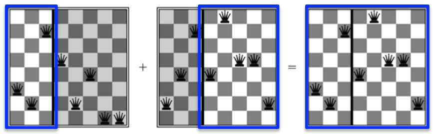
GA: mixing chromosomes

Now we create a new individual based on the chromosomes of the selected parents.

The key mixing operation is **crossover**.

- First choose a **crossover point** in the chromosome
- Then select half of one parent and half of the other

parent 1	3	2	7	5	2	4	1	1
parent 2	2	4	7	4	8	5	5	2
offspring	3	2	7	4	8	5	5	2



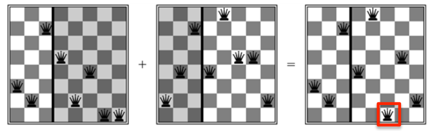
GA: mixing chromosomes

Now we create a new individual based on the chromosomes of the selected parents.

The key mixing operation is **crossover**.

- First choose a **crossover point** in the chromosome
- Then select half of one parent and half of the other

parent 1	3	2	7	5	2	4	1	1
parent 2	2	4	7	4	8	5	5	2
offspring	3	2	7	4	8	1	5	2



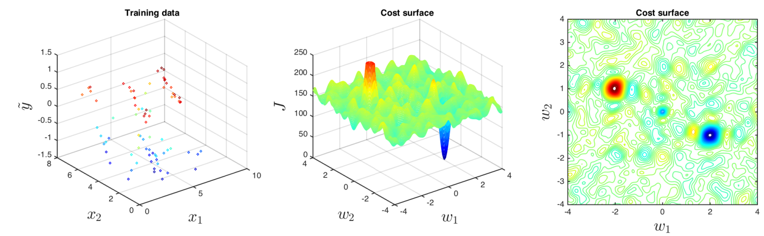
Then, with some low probability, introduce a **mutation** in the child.

The parameters of a genetic algorithm

There are several things you can change:

- How chromosome encode possible solutions - **this is the most important thing! Crossover operations must make sense...**
- Population size – larger populations mean more diversity
- Selection – the way you select will or will not promote diversity
- Mutation probability – more mutations mean more randomness
- You can play with crossover probabilities too
- Elitism: you can choose to retain n fittest individuals in the next generation (often a good idea).

An example: optimisation using genetic algorithm



$y_i = \sin(w_1 x_{1i} + w_2 x_{2i})$

$J = \frac{1}{2}\sum_i (y_i - \tilde{y}_i)^2$

fitness is related to mean squared error of the individual function over sample data

individual is a function of 2-D input

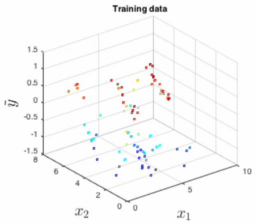
Chromosome:

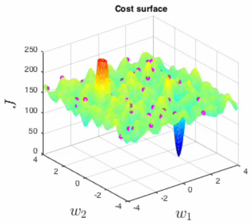
s_1	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	s_2	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}

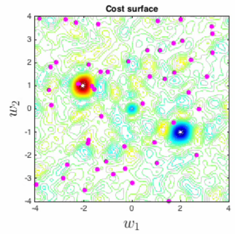
w_1 w_2

$s_i, a_{ij} \in \{0,1\} \quad w_i = (1-2s_i)\sum_{j=1}^{7} 2^{2-j} a_{ij}$

An example: optimisation using genetic algorithm



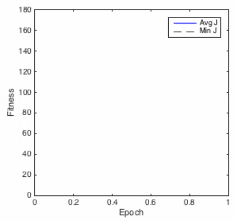




$y_i = \sin(w_1 x_{1i} + w_2 x_{2i})$

$J = \frac{1}{2}\sum_i (y_i - \tilde{y}_i)^2$

GA:
Polulation $k = 50$
Tournament selection $n = 30$
Mutation prob. $= 0.001$



Summary

- Genetic algorithms – a random walk optimisation through the state (or parameter) space
  - Population of individuals is equivalent to a set of possible solutions
  - Fitness function is the cost evaluation
  - Chromosome representation determines the dynamics of the evolution
  - Selection and mixing of fittest individuals
  - Mutation to encourage random exploration

Reading for the lecture: AIMA Section 4.1.4

Reading for next lecture: Review everything