An Introduction to Genetic Algorithms

Kristian Guillaumier 2006, 07

Search and Optimisation

- For example lecture timetabling.
- Finding an optimal set of parameters to some objective function.
- Some problems have a few possible candidate solutions – the search space is small.
    - Enumerating the search space (brute force) is tractable.

The Traveling Salesman Problem

- TSP is defined as:
  - A salesman must visit *n* cities once and ends up where he or she started. In what order must the cities be visited so that the distance traveled is minimised.
- For *n*-city symmetric TSP there are:

$$\frac{(n-2)!}{2}$$ combinations.

- Pick any city to start, there are (n-1) to choose from, then (n-2), (n-3), …
- Half the combinations because the TSP is symmetric.

Hardness

Tours/Sec	Size n				
	4	8	12	16	20
10	0.3 seconds	4.2 minutes	23.1 days	20.73 centuries	$1.92e^{06}$ centuries
50	0.06 seconds	50.4 seconds	4.62 days	4.15 centuries	$0.38e^{06}$ centuries
100	0.03 seconds	25.2 seconds	2.31 days	2.08 centuries	$0.19e^{06}$ centuries
500	0.006 seconds	5.04 seconds	11.09 hours	41.47 years	$0.04e^{06}$ centuries
1000	0.003 seconds	2.52 seconds	5.55 hours	20.74 years	$0.02e^{06}$ centuries
10000	0.0003 seconds	0.252 seconds	33.26 minutes	2.07 years	$1.93e^{03}$ centuries

Approximation

- No efficient algorithm is known to find the optimal solution in reasonable time.
- There are good reasons to believe that no algorithm exists.
- The requirement for a solution still exists.
- Quasi-optimal (approximation) algorithms are sometimes sufficient as long as they execute efficiently.

Approximation Techniques

- Genetic Algorithms.
- Ant Colony Optimisation.
- Neural Networks.
- Monte Carlo Algorithms.
- Simulated Annealing.

Monte Carlo Algorithms as An Example

- Stochastic-iterative.
- Create collections of random solutions (stochastic).
- Iterate till some termination condition (iterative) is met – possible some threshold.
- Best solution is returned.
- Simple to implement.
- Generic – no a-priori knowledge is required or expected from user.
- Performance is proportional to the ratio of ‘good’ solutions to the whole search space.

Genetic Algorithms (1)

- Uses principles of biological evolution to guide the search.
- Inspired by the Darwinian theory of evolution.
- Survival is construed as a search problem guided by some fitness function.
- Survival of the fittest, sexual reproduction and mutation direct the search.
- GAs belong to the class of randomised (not random) search algorithm.

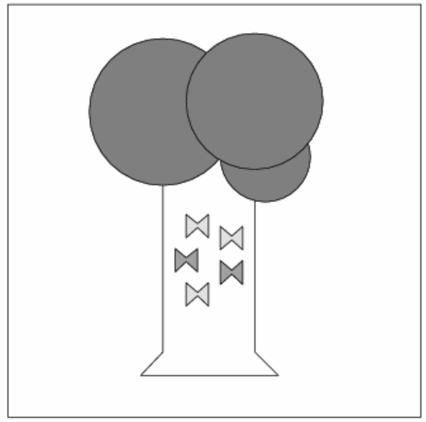
Genetic Algorithms (2)

- Suffers from premature convergence in very large search spaces.
- Combine heuristics to form hybrid-GAs.
- Relatively easy to ‘parallelise’.

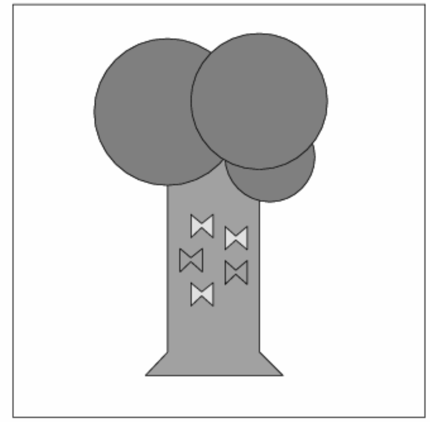
Motivation

- Based on natural selection.
- In nature populations are made up of individuals that vary from one another.
- These variations allow them to perform better in their environment.
- Hence have a change of reproducing and propagating their genetic ‘advantage’.
- Genetic material of weaker individuals will become increasingly diluted.

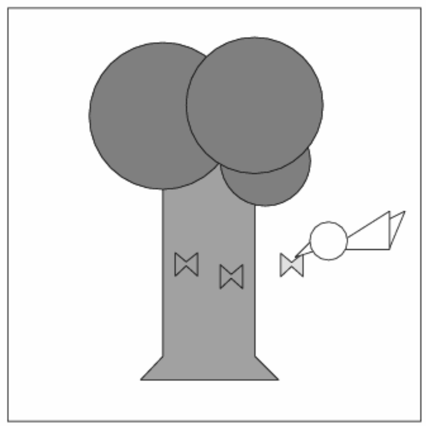
Motivation



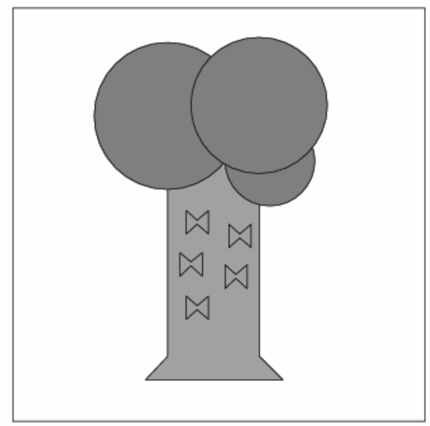
Dark and light coloured peppered moths.



Sooty smoke kills lichen on trees and the bark darkens.



Light coloured moths are easily seen by birds and are eaten.
In their environment the light coloured moths are 'unfit'.



The dark coloured moths have more opportunity to reproduce since they are 'fitter' in their environment. The species has now become resistant.

Components of a GA

- Represent candidate solutions encoding them as chromosomes.
- A method for creating an individual population.
- An evaluation that plays the role of the environment rating solutions in terms of their fitness.
- Genetic operators (reproduction) to derive new candidate solutions from previous ones (parents/children).
- Other parameters (e.g. population size).

Reality Check

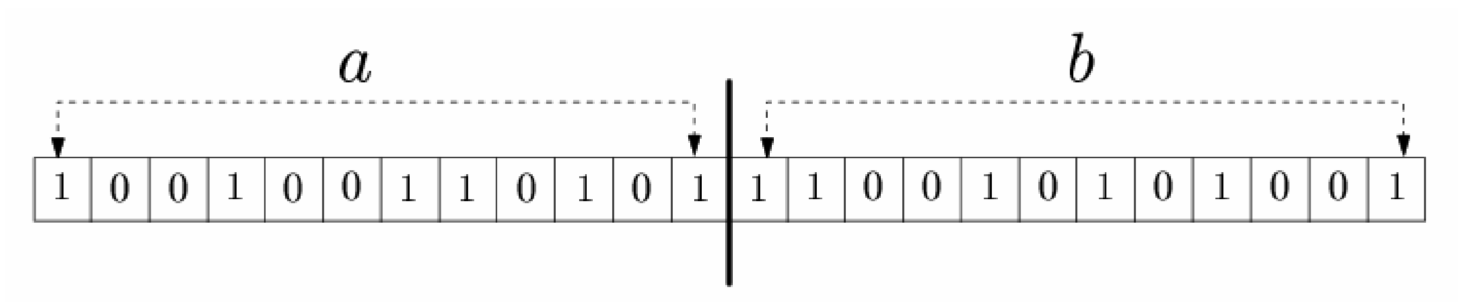
- Consider the requirement of maximising the function x^2 over the interval x: $0 \rightarrow 15$.

Chromosome Encoding

- The characteristics of a living organism are encoded in its chromosomes.
- In turn, chromosomes contain genes that are the basic determinants of these characteristics.
- A ‘gene value’ is an allele.
- For example we may have a gene for hair colour whose allele specifies black hair.
- A common chromosome encoding is binary encoding (as demonstrated in the previous reality check example).

Chromosome Encoding

- Multi-parameter binary encoding.



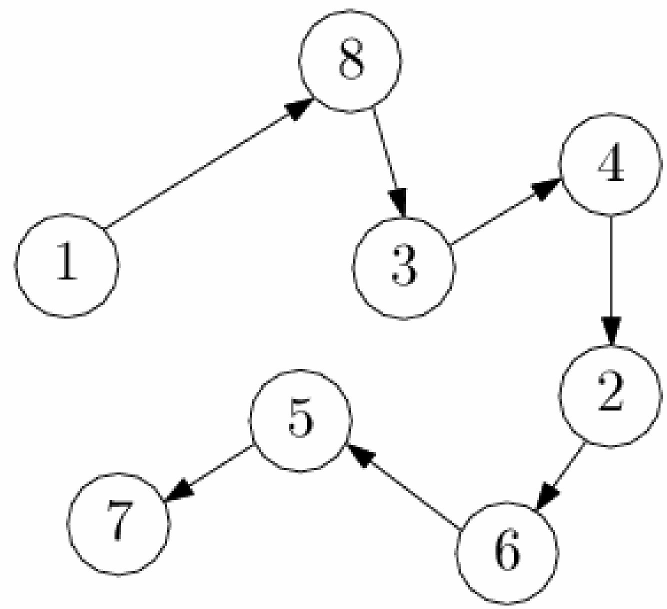
- Encoding may have higher cardinalities, for example using base-10 digits.
- Empirical studies show that sometimes using a domain-specific encoding scheme is convenient and efficient.

Permutation Encoding

- Useful for TSP:

Chromosome

1	8	3	4	2	6	5	7



Initialising the Population

- Sometimes the procedure to initialise the population is crucial to the preformance of the algorithm.
- In the best case a poorly initialised solution will affect the online time of the algorithm,
- In the worst case it will prevent it from converging to a satisfactory solution.
- Methods:
  - Random initialisation (simple but not guided, benchmarking quality if algorithm not an ‘unfair advantage’).
  - Seeded initialisation (seeded with the best solutions from a previous run, or seeded by the results of some other algorithm, too much seeding will destroy diversity).

The Fitness Function

- Sometimes the definition of fitness is guided by a simple function:

Chromosome 1:
a=36, b=53

Fitness $= a^2 + b^2 = 36^2 + 53^2 =$ **4105**

1	0	0	1	0	0	1	1	0	1	0	1

Chromosome 2:
a=24, b=39

Fitness $= a^2 + b^2 = 24^2 + 39^2 =$ **2097**

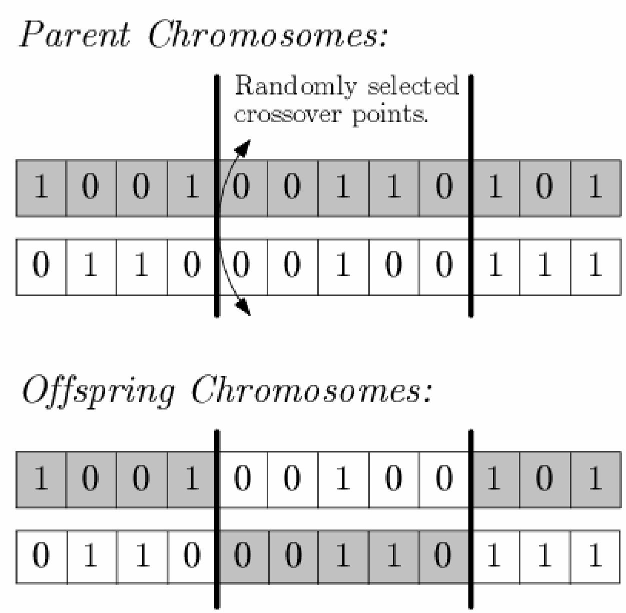
0	1	1	0	0	0	1	0	0	1	1	1

The Fitness Function

- In other cases the definition is more involving.
- E.g. Timetable scheduling:
  - Staff clashes.
  - Student clashes.
  - Exclusions from timeslots.
  - Preferences.
  - Hard constraints/Soft constraints.
- Penalty systems can be used in such cases.

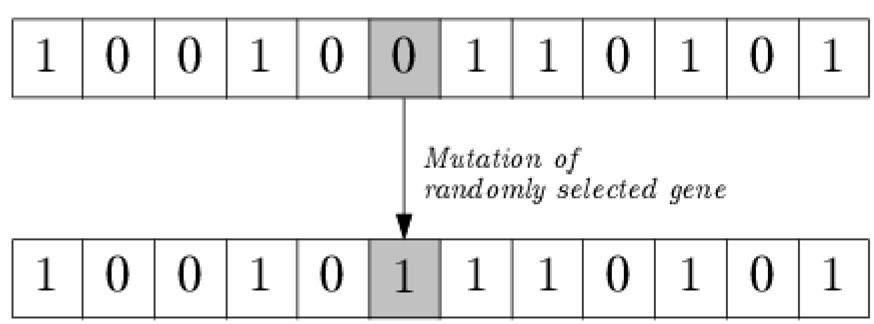
Genetic Operators

- Crossover: pairs of parent chromosomes combine to yield offspring.
- Many variants: single-point, two-point, n-point, uniform.
- Two-point crossover:



Genetic Operators

- Mutation: mutate genes in the chromosomes.
- Crossover does not create genetic material (active or deactivate new genes) in the population so getting caught in local maxima/minima can be a problem.
- On the other hand mutation can introduce new material.
- Too much mutation will degenerate the search into an MCA.



Genetic Operators

- Crossover and mutation can cause problems.
- Need for genetic repair or redesign of the algorithm.
- E.g.
  - Swap mutate for TSP.
  - Partially matched crossover for TSP.

Partially Matched Crossover

Parent Chromosomes

Genotype A	Chromosome 1
Gene 1	1
Gene 2	3
Gene 3	2
Gene 4	4
Gene 5	7
Gene 6	6
Gene 7	10

Matching Section

Genotype B	Chromosome 1
Gene 1	2
Gene 2	8
Gene 3	6
Gene 4	7
Gene 5	1
Gene 6	3
Gene 7	9

Offspring Chromosomes

Genotype A	Chromosome 1
Gene 1	1
Gene 2	8
Gene 3	6
Gene 4	7
Gene 5	4
Gene 6	2
Gene 7	10

Matching Section

Genotype B	Chromosome 1
Gene 1	6
Gene 2	3
Gene 3	2
Gene 4	4
Gene 5	1
Gene 6	8
Gene 7	9

Selection (Fitness based)

- Determine which chromosomes will act as parents to reproduce and yield offspring – propagate genetic material.

1. Roulette wheel.
2. Rank-based: If there is a ‘super’ individual in the population the roulette wheel will failed because that individual would be selected too often. Select by rank (relative position).

Selection (Fitness based)

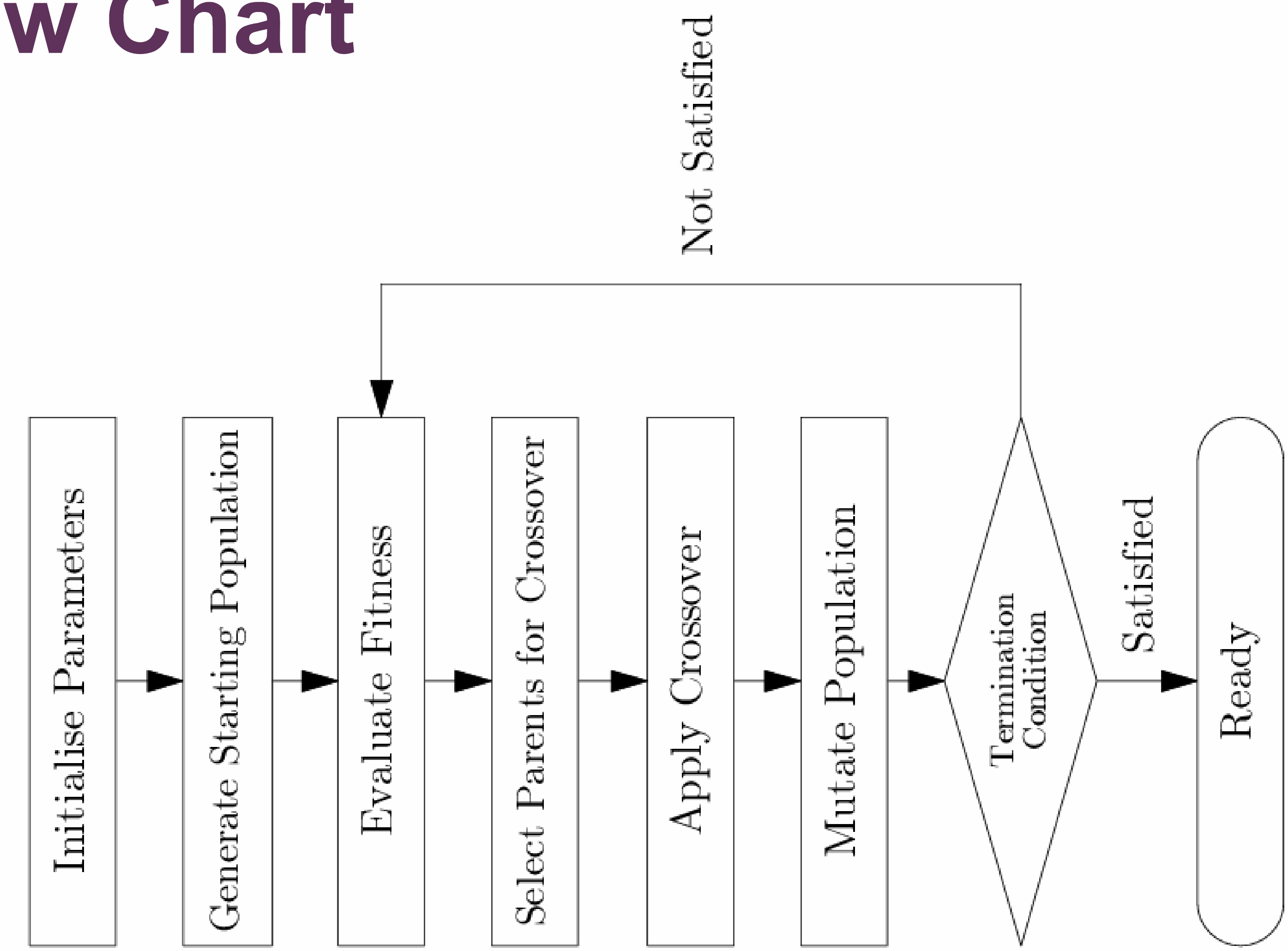
3. Tournament-based: K chromosomes are selected at random, the fittest out of the K is selected.

- Elitism:
  - Sometimes the parents can outperform the offspring (the children are not necessarily better).
  - If parents are substituted good solutions may be lost.
  - In elitism the fittest *n* individuals are selected from the current population to randomly seed the new population (or maybe replace the worst ones).

Other Parameters

- Population size.
- Crossover rate: the probability that 2 parents will reproduce before entering the new population.
- Mutation rate: the probability that a new chromosome entering the population is mutated.

Flow Chart


Not Satisfied
Initialise Parameters
Generate Starting Population
Evaluate Fitness
Select Parents for Crossover
Apply Crossover
Mutate Population
Termination Condition
Satisfied
Ready

TSP as a Case Study