Simulation and Optimization for Decision Support (G54SOD)

Semester 2 of Academic Session 2017-2018
Dr Dario Landa-Silva
dario.landasilva@nottingham.ac.uk
http://www.cs.nott.ac.uk/~pszjds

Lecture 7 – Introduction to Evolutionary Algorithms

- Basics of Evolutionary Algorithms
  To describe the motivation and fundamental concepts of EAs
- Design of Evolutionary Algorithms
  To explain some of the most important design issues to consider when implementing evolutionary algorithms

Additional Reading

Chapter 3 of (Talbi, 2009)

A Framework for the Description of Evolutionary Algorithms. A. Hertz, D. Klober. *European Journal of Operational Research*, Vol. 126, No. 1, pp. 1-12, 2000.

Genetic Algorithms for the Operations Researcher. C. Reeves. *INFORMS Journal on Computing*, Vol. 9, No. 3, pp. 231-250, 1997.

A Genetic Algorithm for the Generalised Assignment Problem. P. Chu, J. Beasley. *Computers and Operations Research*, Vol. 24, No. 1, pp. 17-23, 1997.

A Genetic Algorithm for a Workforce Scheduling and Routing Problem. Haneen Algethami, Rodrigo Lankaites Pinheiro, Dario Landa-Silva. *2016 IEEE Congress on Evolutionary Computation (CEC 2016)*, pp. 927-934 , IEEE Press, Vancouver, Canada, July 2016.

Basics of Evolutionary Algorithms

Natural Evolution as a Problem Solving Method

All the complexity in the natural world has evolved from a single common ancestor. How?

The theory is that given:

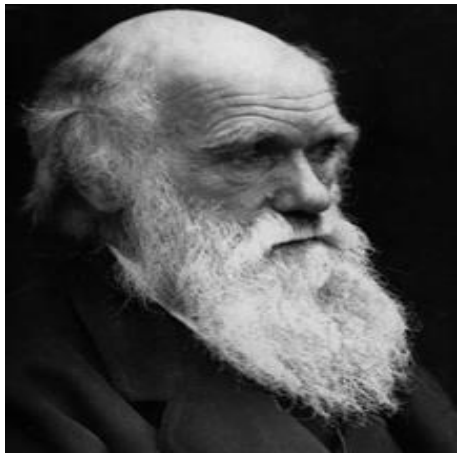
1. A population of organisms which have a lifetime and which can reproduce in a challenging/changing environment.
2. A way of continually generating diversity in new 'child' organisms.

A 'survival of the fittest' principle will naturally emerge: organisms which tend to have healthy, fertile children will dominate.

Natural Evolution

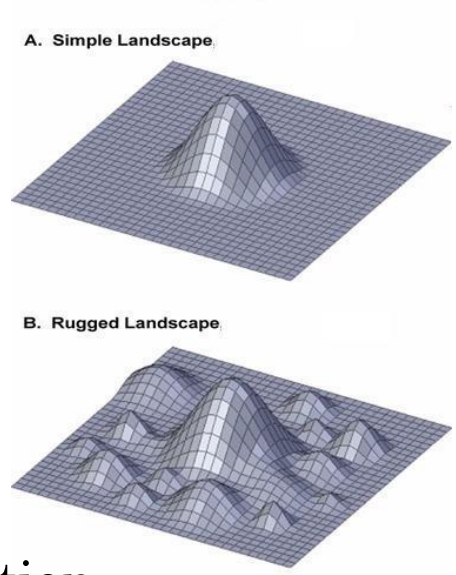
"As many more individuals of each species are born than can possibly survive; and as, consequently, there is a frequently recurring struggle for existence, it follows that any being, if it vary however slightly in any manner profitable to itself, under the complex and sometimes varying conditions of life, will have a better chance of surviving, and thus be naturally selected. From the strong principle of inheritance, any selected variety will tend to propagate its new and modified form."

Charles Darwin, The Origin of Species (1859)



Features of Evolutionary Algorithms

The rationale for evolutionary algorithms (EAs) is to maintain a population of solutions during the search. The solution (individuals) compete between them and are subject to selection, and reproduction operators during a number of generations.



Exploitation vs. Exploration

- Having many solutions instead of only one.
- Survival of the fittest principle.
- Pass on good solution components through recombination.
- Explore and discover new components through self-adaptation.
- Solutions are modified from generation to generation by means of reproduction operators (recombination and self-adaptation).

Darwinian Evolutionary Algorithm

- Competing population
- Dynamic population
- Fitness
- Dynamic genetic inheritance

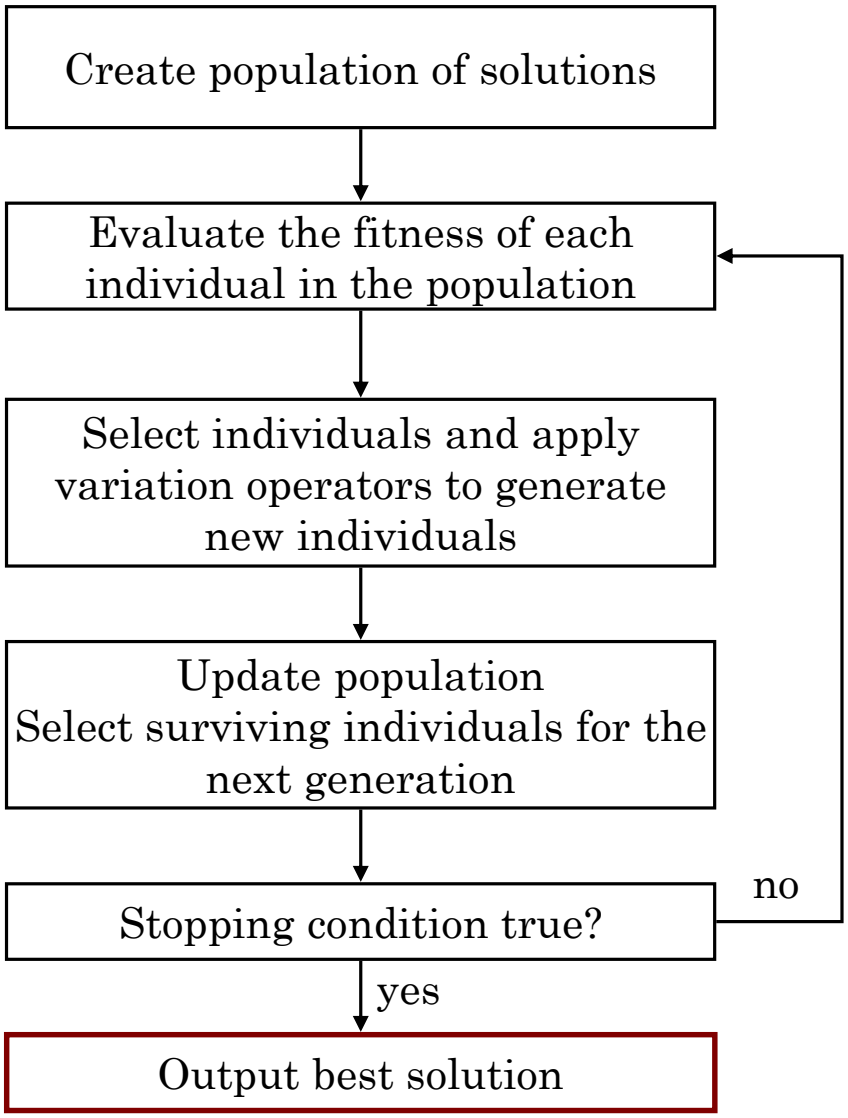
Lamarckian Evolutionary Algorithm

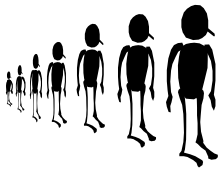
- Competing population
- Dynamic population
- Fitness
- Dynamic cultural inheritance

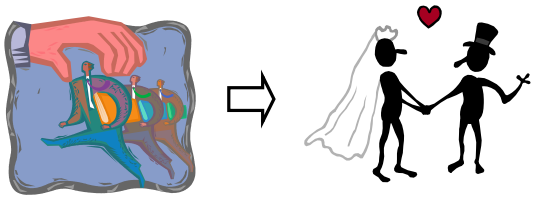
Baldwinian EAs use a kind of intermediate between genetics and culture suggesting the existence of a strong interaction between learning and evolution.

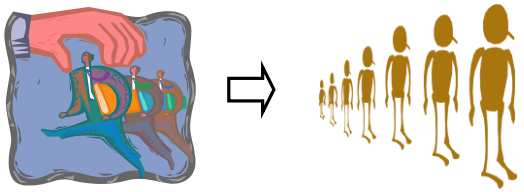
Dawkins proposed the concept of memetics to refer to cultural heritage.

Typical Evolutionary Algorithm Cycle









Example of EA for Function Optimization

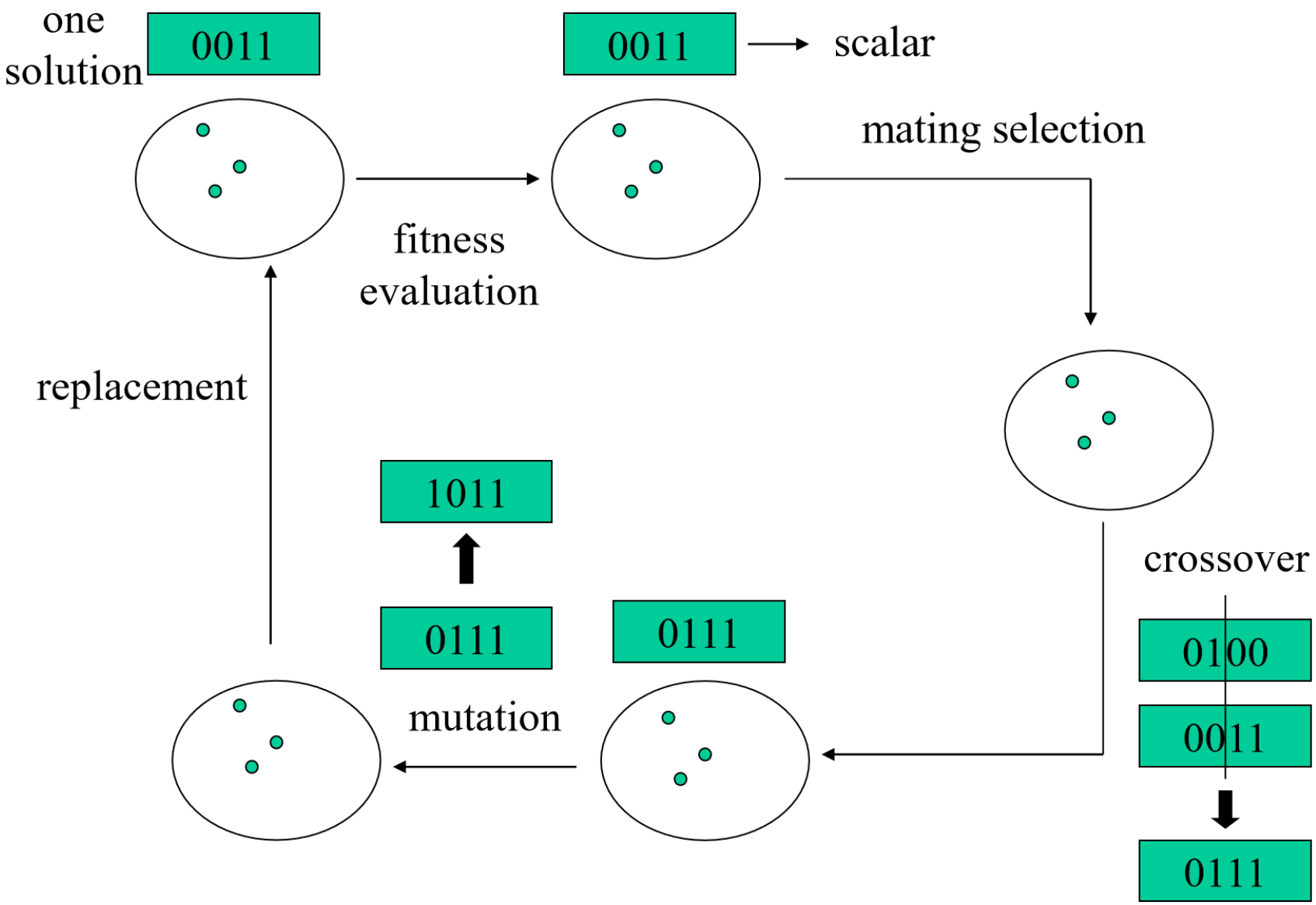
Example. Maximise a function

Maximise $x^3 - 60x^2 + 900x + 100$

where x is an integer in the range (0, 31).

Starting population:

Solution	String	x	F(x)
1	10011	19	2399
2	00101	5	3225
3	11010	26	516
4	10101	21	1801
5	01110	14	3684


one solution
0011
0011
scalar
mating selection
fitness evaluation
replacement
crossover
0100
0011
0111
1011
0111
0111
mutation

Generational Genetic Algorithm

Genetic operators applied repeatedly to generate a full new population in each generation.

Initial Population

Name	F(S)
S1	0.1
S2	0.5
S3	0.3
S4	0.2
S5	0.9
S6	0.7
S7	0.3
S8	0.4
S9	0.4
S10	0.1

Apply selection and genetic operators 10x to give new population

Generation 1

Name	F(S)
S11	0.5
S12	0.3
S13	0.3
S14	0.7
S15	0.7
S16	0.9
S17	0.4
S18	0.9
S19	0.9
S20	0.3

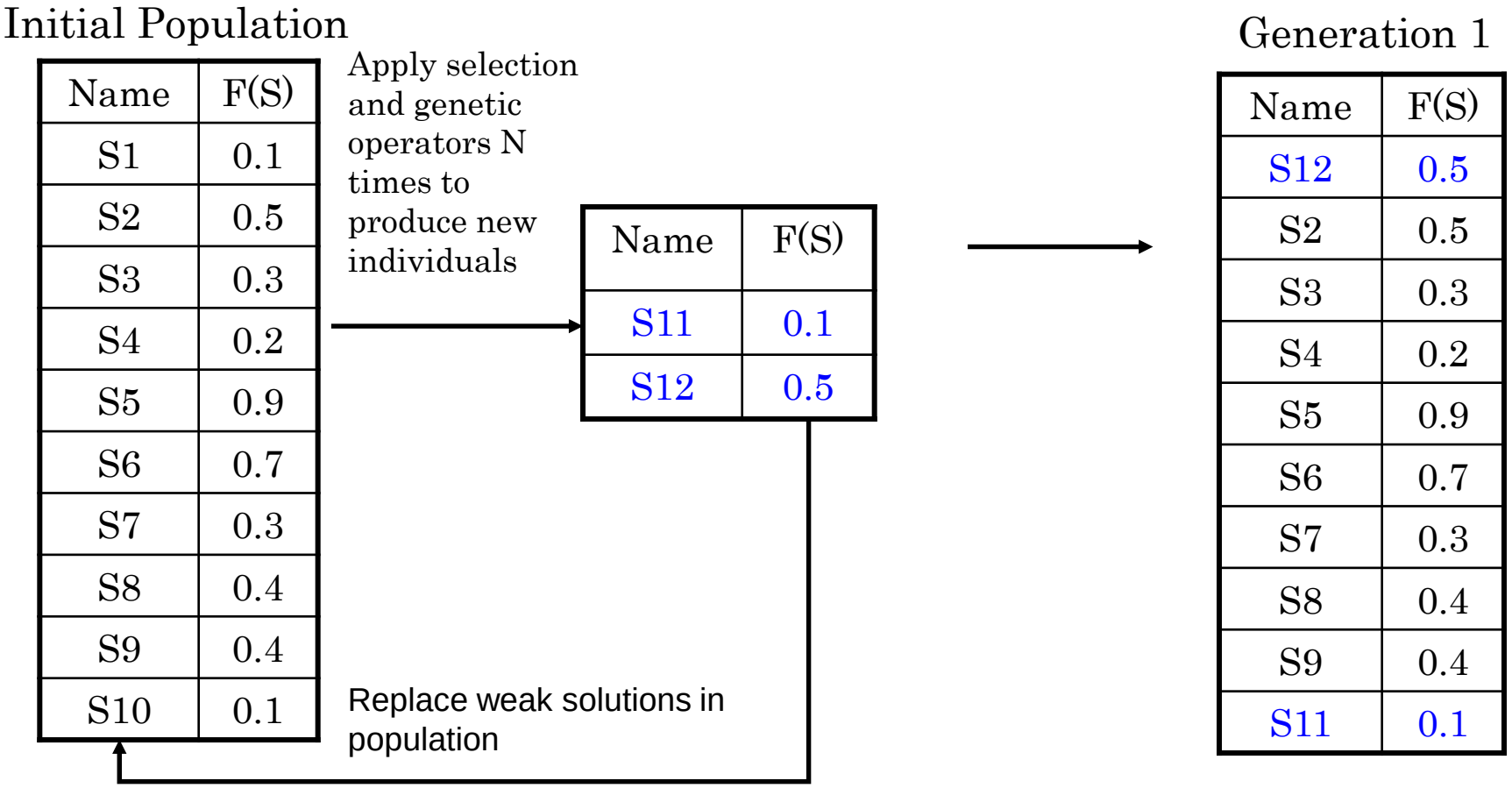
Apply selection and genetic operators 10x to give new population

Generation 2

Name	F(S)
S21	0.7
S22	0.8
S23	0.9
S24	0.9
S25	0.7
S26	0.8
S27	0.5
S28	0.7
S29	0.6
S30	0.7

Steady-state Genetic Algorithm

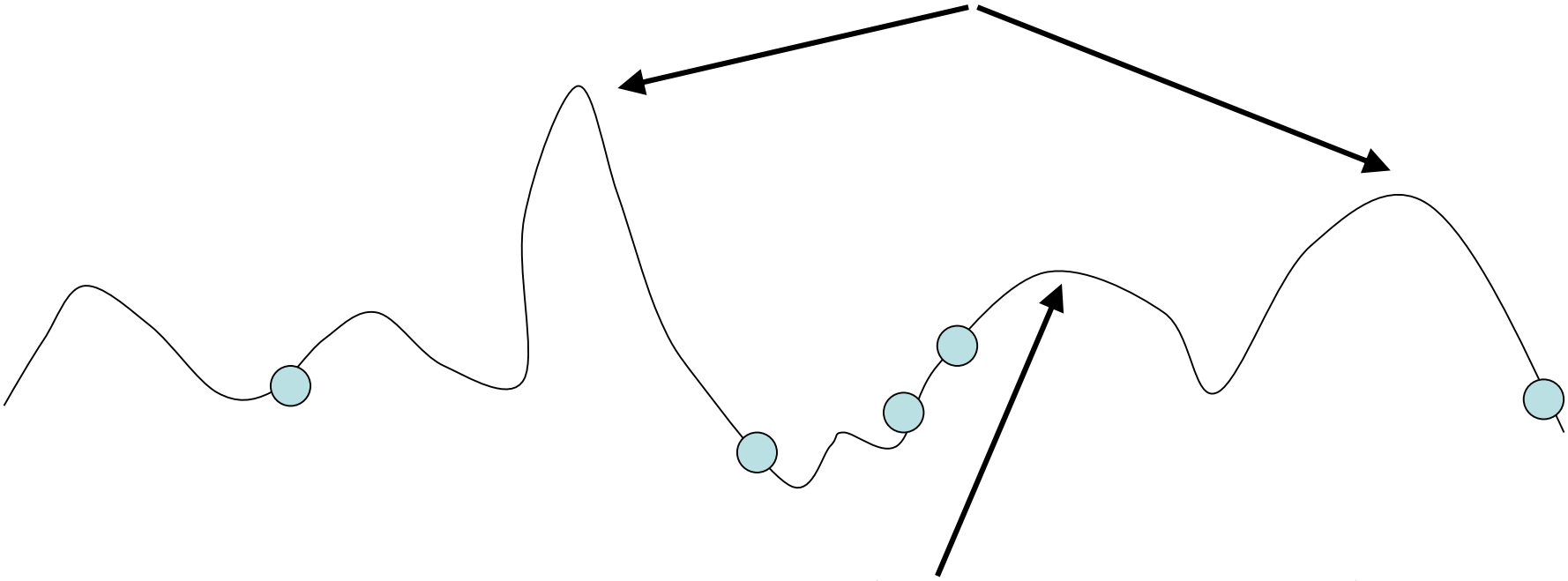
Genetic operators applied N times and only some bad solutions replaced in each generation.



Selection Pressure in EAs

Very low pressure selection (e.g. random selection)
No evolutionary `progress' at all

A modest level of pressure
You may well find yourself here or here



Very high pressure (e.g. always select best)
You will end up here

Design of Evolutionary Algorithms

Typical Components of an EA

- Solution representation (encoding)
- Population size
- Initialisation strategy
- Evaluation function
- Reproduction operators
- Selection mechanism
- Stopping condition to detect convergence

A diverse range of evolutionary procedures can be designed for the subject problem by tuning the above components.

It is crucial to design components and tune parameters while considering the choices made on the various parts of the algorithm.

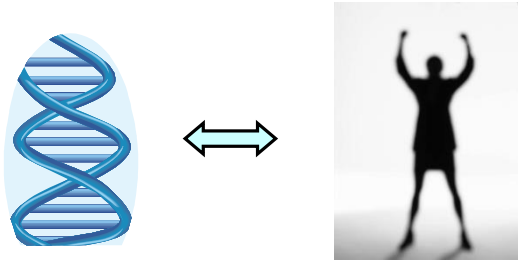
Solution Representation (encoding)

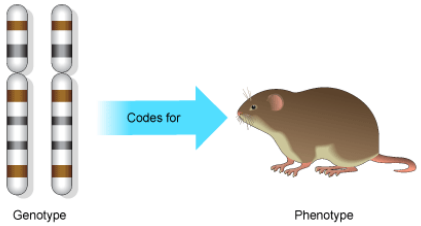
A good representation ensures that variation operators maintain a functional link between current and new solutions.

Some common representations:

- Binary vectors
- Permutations
- Symbolic (e.g. parse trees)

The representation can be genotypic of phenotypic and then the Hamming or Euclidean distance can be used to measure the distance between solutions.





Direct Encodings vs. Indirect Encodings

Direct

- Straightforward genotype (encoding) → phenotype (individual) mapping
- Easy to estimate effects of mutation
- Fast interpretation of chromosome (hence speedier fitness evaluation)

Indirect

- Easier to exploit domain knowledge
- Possible to 'encode away' undesirable features
- Can seriously cut down the size of the search space
- Slow interpretation
- Neighbourhoods are highly rugged

Population Size and Initialisation Strategy

Commonly, μ denotes number of parents and λ denotes number of offspring, the new population is obtained from the μ + λ individuals.

Common initialisation strategies:

- Uniformly at random
- Incorporating problem domain knowledge
- Incorporating some elite solutions

The number of parents λ can be seen as the degree of parallelism of the evolutionary search.

The number of offspring μ can be seen as the degree at which the parents as used as the basis to generate new individuals.

The difficulty of the fitness landscape must be taken into account when setting the population size parameters μ and λ.

Evaluation Function

A good evaluation function assesses the quality of evolved solutions and helps to assess the effectiveness of the reproduction operators.

Usually, evaluating fitness is a time-consuming process in EAs so delta and/or approximate evaluation might be useful.

Moreover, it might be enough to have a way to discriminate between individuals without conducting an exact fitness evaluation.

Reproduction Operators

The design of reproduction operators must be done according to the solution representation and the fitness landscape.

Some common operators (recombination and mutation):

- Changing solution component values
- Perturbing solution component values
- Combining complete solutions
- Blending solution component values

When designing reproduction operators ⇒ allow infeasibility?

Mutation: some variation of the parent after cloning.

Recombination: components of parents are cloned then combined.

Note: reproduction operators within an EA can be applied to produce 1 or more offspring at the same time.

Mutation on String Encodings

Single-gene Mutation

Choose a gene at random, and change it to a random new value, e.g. 352872 → 312872

M-gene Mutation

Multiple instances of single-gene mutation

Swap Mutation

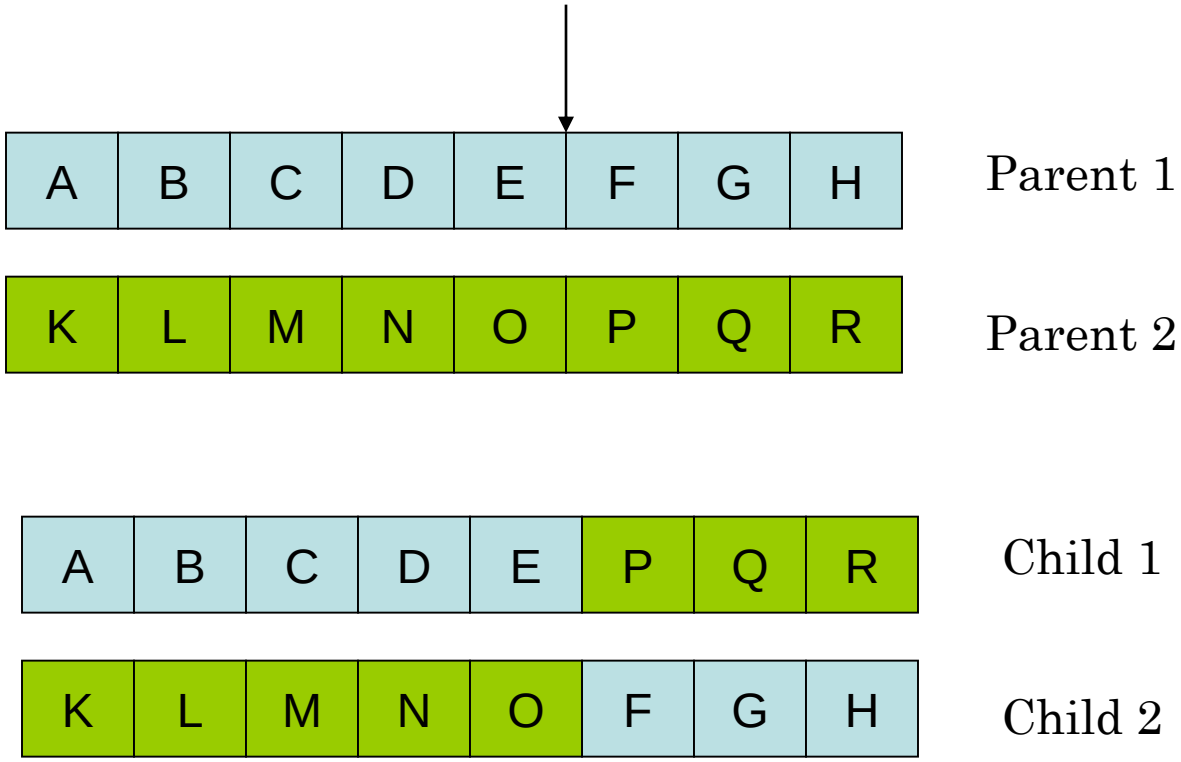
Choose two genes at random, and swap them,

e.g. 352872 → 372852

Why is this probably not much good in this context?

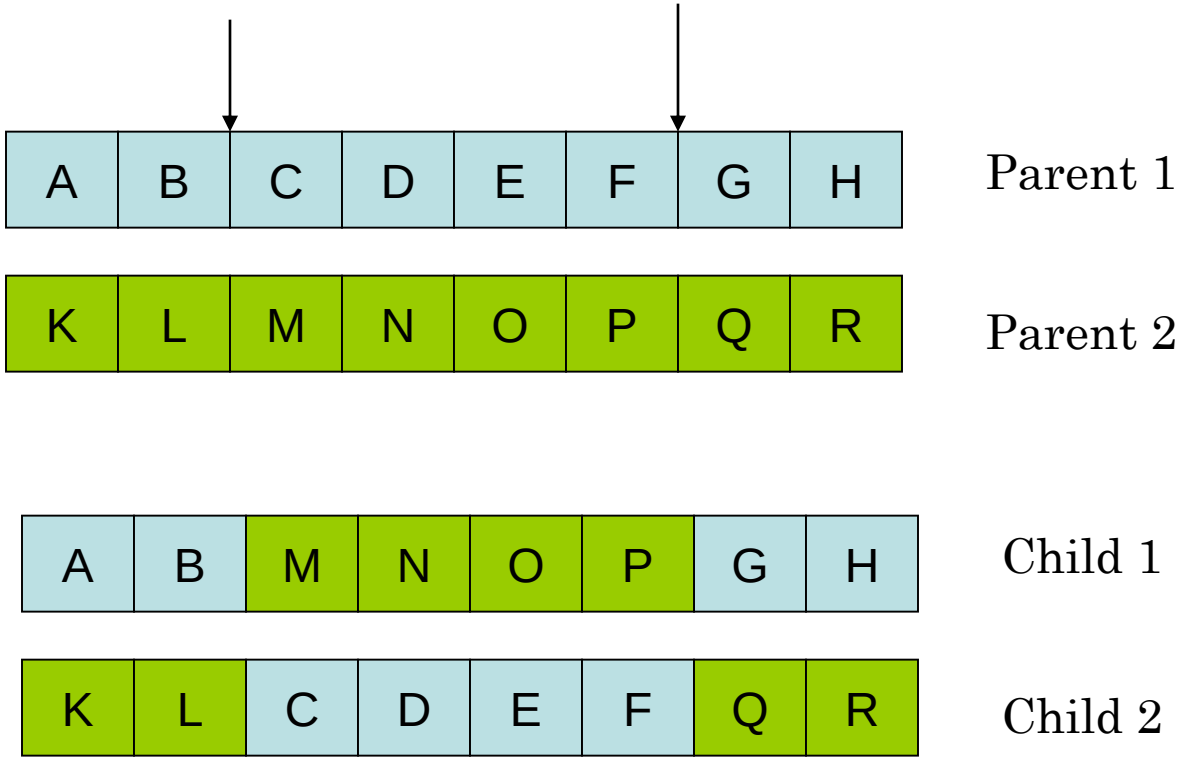
Standard Recombination Operators

1-point crossover



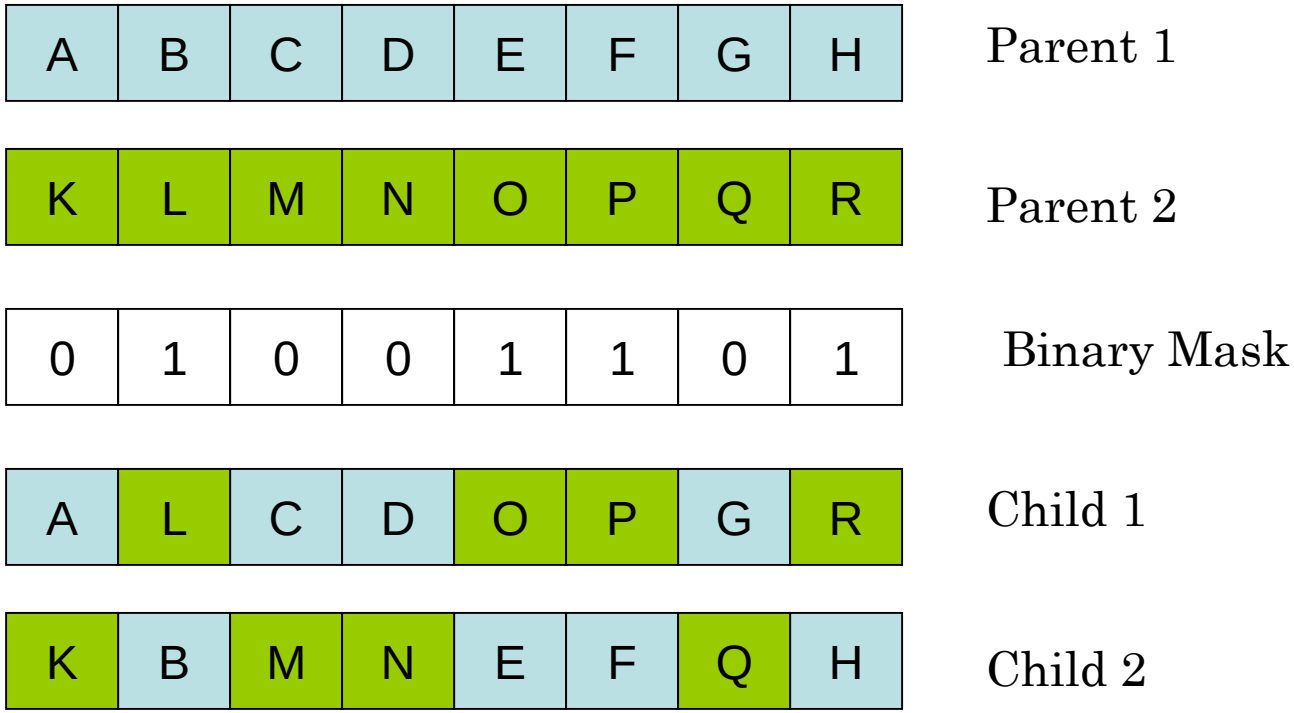
Standard Recombination Operators

2-point crossover
k-point crossover is the generalisation



Standard Recombination Operators

Uniform crossover
Generate a random binary 'mask'
Used to decide which genes are swapped and which stay



Selection Mechanism

Selection is applied in two contexts: for deciding which individuals will reproduce to generate new solutions (selection for reproduction) and for deciding which individuals will survive to the next generation (selection for survival).

Some issues to consider in the selection mechanism:

- Sampling with or without replacement?
- Generational or steady-state?
- Elitism strategy?
- Deterministic of stochastic?
- In stochastic selection: proportional, tournament or ranking?

Common notation: μ denotes number of parents and λ denotes number of offspring. Some selection strategies: $(\mu+\lambda)$, (μ,λ)

In terms of selection pressure, deterministic selection is the most stringent while proportional selection is the least.

Fitness Proportionate Selection

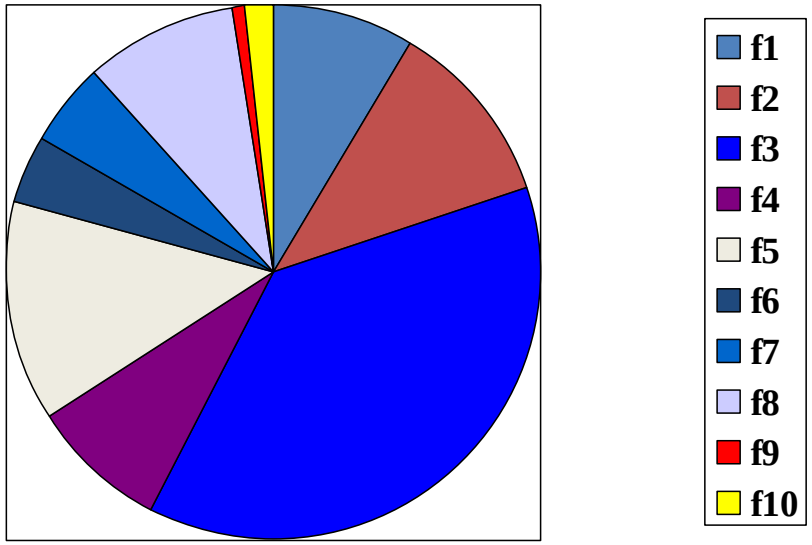
This is also called roulette wheel selection

Suppose there are P individuals with fitness f1, f2, …, fP

The probability of selecting individual i is simply:

$$\frac{f_i}{\sum_{k=1}^{P} f_k}$$

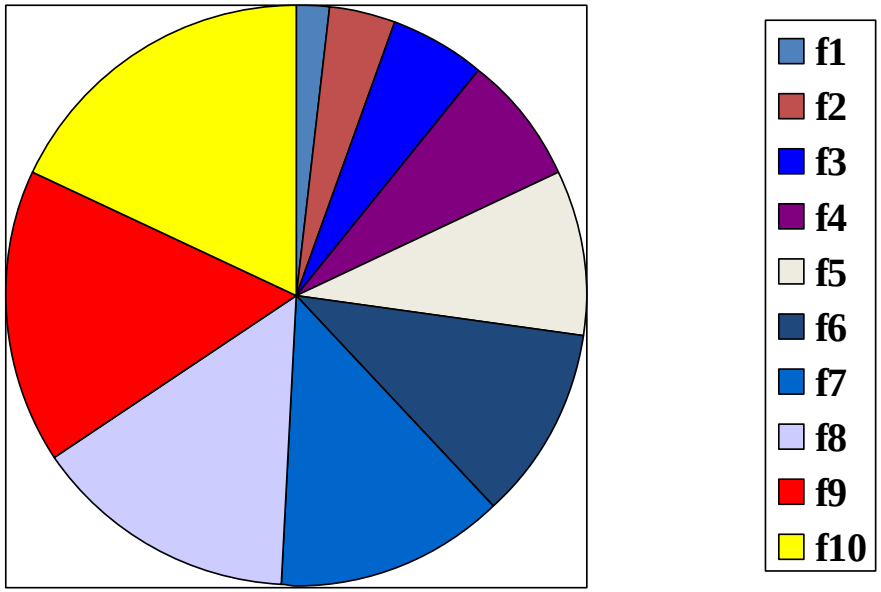
This is equivalent to spinning a roulette wheel with sectors proportional to fitness.



Rank Based Selection

The fitness values in the population are ranked from PopSize (fittest) down to 1 (least fit). The selection probabilities are proportional to rank.

There are variants where the selection probabilities are a function of the rank. For example $rank^{0.5}$ (low bias) or $rank^{2}$ (high bias).



Tournament Selection

Tournaments of size = t are executed to select individuals that compete between them.

```
Repeat t times
    choose a random individual from the pop
    and remember its fitness
Return the best of these t individuals
```

- Very Tunable Method
- Avoids the problems of super-fit or super-poor solutions
- Very simple to implement
- Requires a parameter (tournament size t)
- Tournaments can be with or without replacement

Examples of Evolutionary Algorithms

- Genetic Algorithms (GA)
- Evolutionary Strategies (ES)
- Genetic Programming (GP)
- Ant Colony Optimisation (ACO)
- Memetic Algorithms (MA)
- Particle Swarm Optimisation (PSO)
- Differential Evolution (DE)
- Estimation of Distribution Algorithm (EDA)
- Cultural Algorithms (CA)

Since this a very active research area, new variants of EAs and other population-based meta-heuristics are often proposed in the specialised literature.