Evolution of Soft-Ware

The power of evolution

- Create machines that solve problems by means of evolution
  - Evolution creates a wide richness of solutions
  - Machines can adapt to changing environments
- Adaptation and richness of solutions is important for
  - Data analysis of company and customer data
  - Finding creative solutions in optimization tasks
    - Optimal usage of resources
    - Maximizing the income

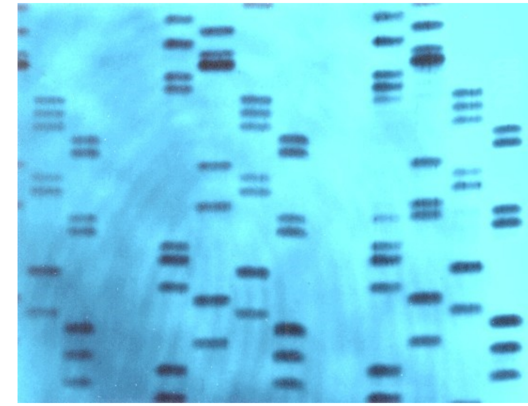
Terms

- Microscopic
  - Each gene encodes a trait (color of eyes)
    - The possible values of a trait are alleles (blue, brown)
  - Chromosomes consists of genes (blocks)
    - Here: one individual is one chromosome as each individual has only one chromosome
  - Genome - All Genes taken together
  - DNA consists of Chromosomes (an organism model)
- Macroscopic
  - Individual – a phenotype expression or instantiation according to the genotype model
  - Population – all individuals

History

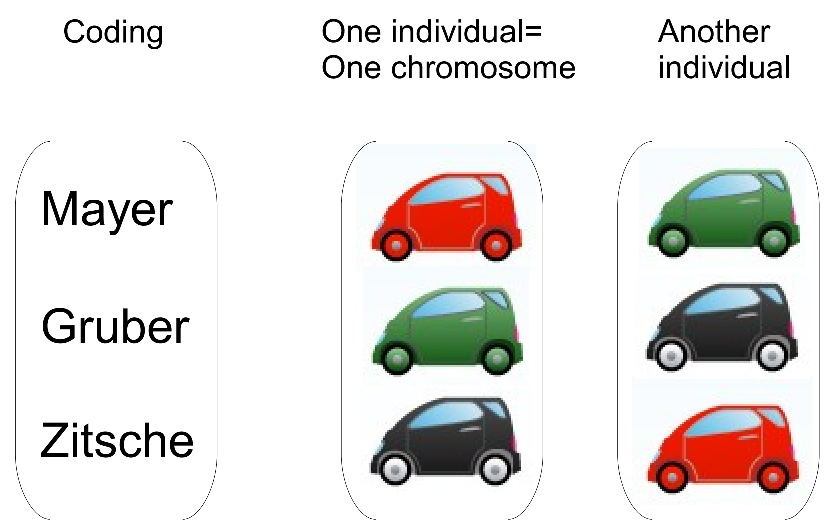
- Theory of Transmutation Lamark (1809)
- Selection of the fittest, On the Origin of Species Charles Darwin (1859) and Alfred Russel Wallace
- Genetics and Mendel's Law Gregor Mendel (posthumed after 1884)
- Evolutionary Computing by I. Rechenberg (1960)
- Genetic Algorithms John Holland (1975)
- Genetic programming from John Koza (1992)

Encoding the problem



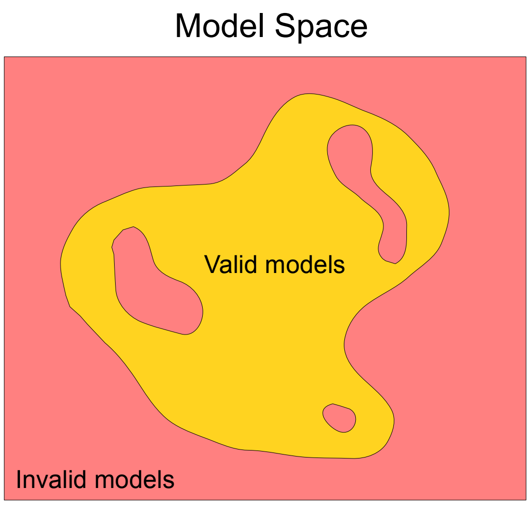
http://www.marlerblog.com/DNA-genetic-fingerprinting-on-fingerprint-blue-backdrop-1-AJHD%281%29.jpg

- Depends on the domain
- Is the toughest decision you have to take
- Influences the conversion speed and quality of solution
- Example
  - Allocation of cars to managers



http://www.dapino.nl/images/MiniCarIcons_all.jpg

Genes



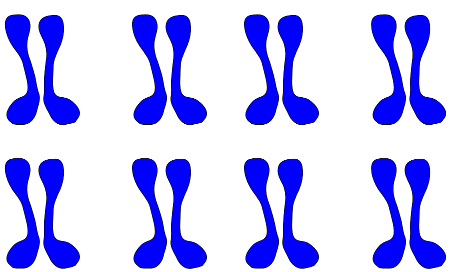
- Value encoding {'red','green','blue'}
- Real numbers {3.12, 3.34, 6.234}
- Binary Strings {10101000101110}
- Permutation {1 4 6 2 7 3 4 9 8}
  - Only unique numbers, require special consistency checkers
  - additional checks required
- Models can “leave” the valid model space
  - Operators may create “invalid” individuals

http://dvice.com/pics/garmin_car_icons.jpg
/www.dapino.nl/images/MiniCarIcons_all.jpg

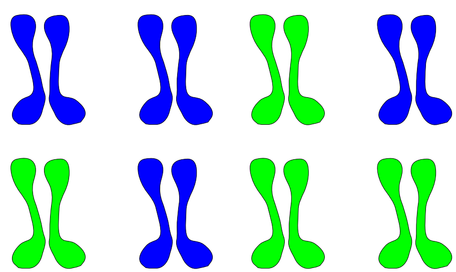
Learning algorithm

- Decide which individuals / chromosomes have the highest fitness (least error) to a given world
- Those individuals can recombinate and form

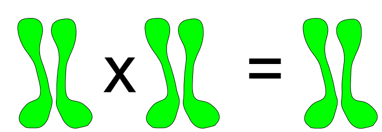
The Population
at evolution t



Select the
“best” individuals

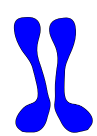


asexual and
sexual reproduction



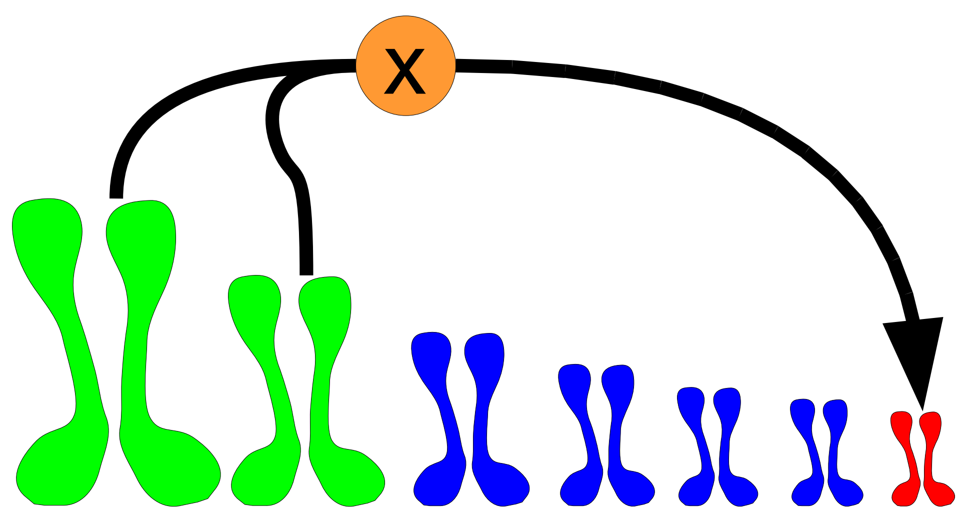
Fitness function

- Domain specific
- Used to compare the chromosomes and separate out-performers from under-performers
- Is related the the error-function
  - Chromosomes must maximize the fitness function

$f($  $) \rightarrow N$

Non-selected individuals

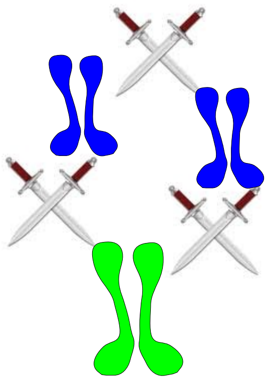
- Use a selection function that is based on the individuals fitness
  - As mostly the best fit organisms survive (Darwin)
- The least performant individuals are replaced by
  - randomly generated candidates
  - An offspring of two well-performers
  - An identical copy of a well performer



Selection

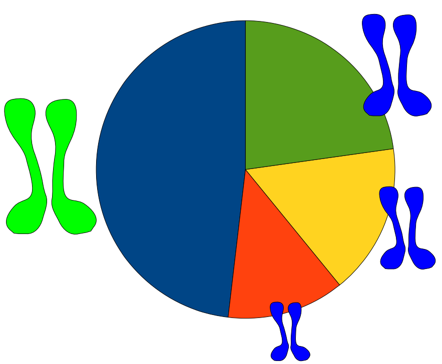
- Tournament selection
  - Select n individuals with uniform probability and let them compete with each other (the fittest enters the new generation)

Competition in Subgroups



- Roulette wheel
  - Select individuals with respect to fitness (areas on a roulette wheel increases with fitness)
  - A individual with 99% fitness is almost always selected

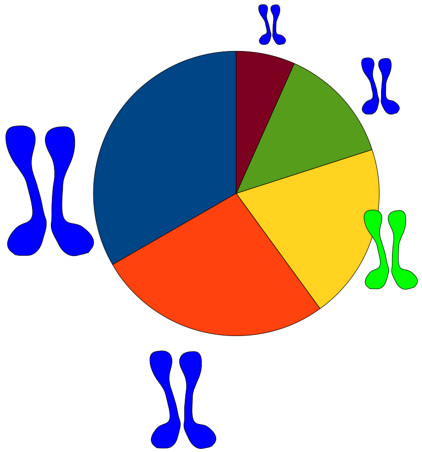
Roulette wheel



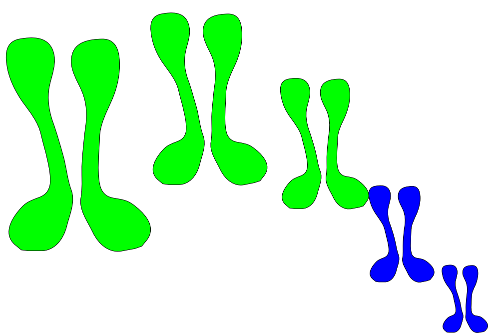
Selection

- Rank selection
  - Probability is a function of rank position (not prone to outperformers as in Roulette whele selection)
- Truncation selection
  - Selecting the first half or third of the best individuals

Ranking



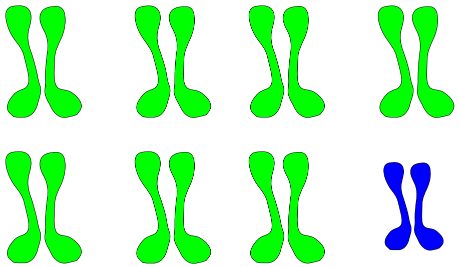
Truncation



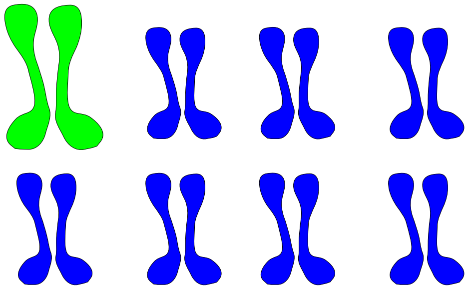
Selection

- Steady State selection
  - Take only some very tough individuals and replace only few low-performers
- Elitism
  - Always save the best individual for the next population (strongly recommended to be used)

Steady State

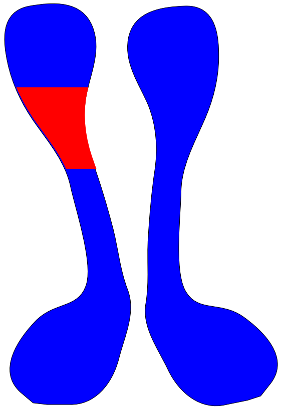


Elitism



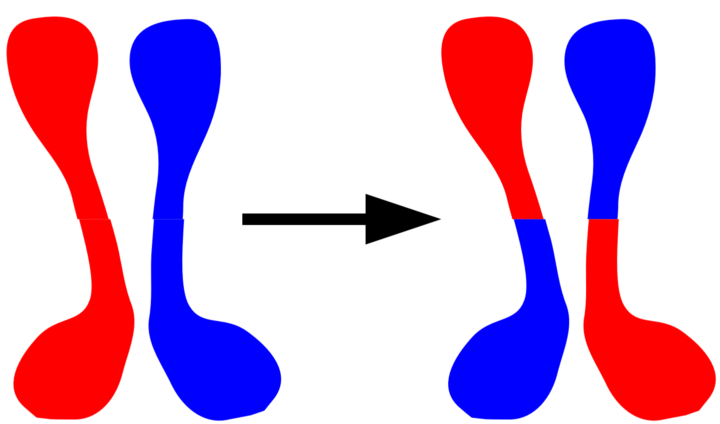
Mutation (unary operation)

- Mutations rate should be low (0.5% - 1%)
- Used for graduate changes exploring the search space
  - Keep non-mutated information intact
  - Changes are totally random
- Usually is applied after reproduction to simulate transcription errors

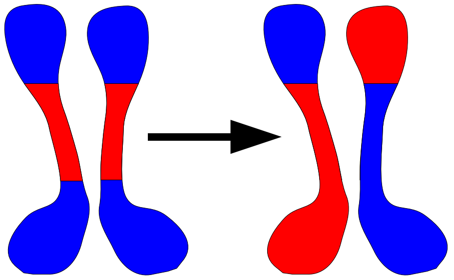


Cross Over (binary operation)

- The hope is the two normal performers are combined to a best-performer
- Crossover probability
  - 0% = just copy the parents
  - 100% = use crossover for each offspring
  - Recommended: 60%-95%
- Single point crossover



Crossover



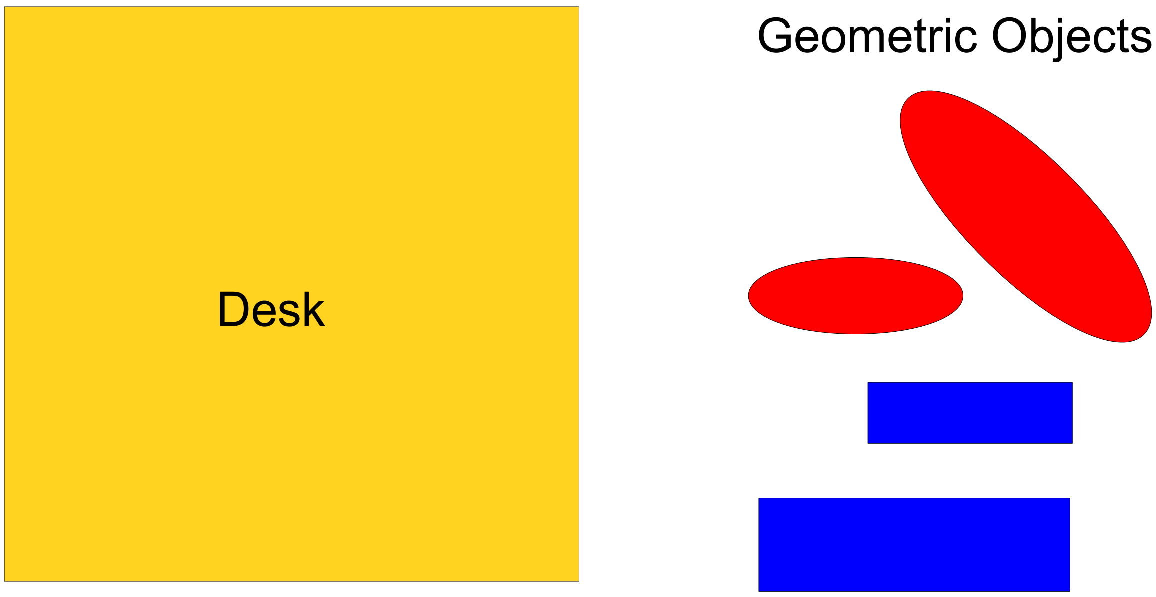
- Two point crossover
- Uniform crossover
- Arithmetic crossover
  - Mathematical functions applied (binary strings)

$$\begin{pmatrix}1\\0\\0\\1\\1\\1\\0\end{pmatrix} \times \begin{pmatrix}1\\0\\1\\0\\1\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\\1\\0\\0\end{pmatrix}$$

e.g. AND

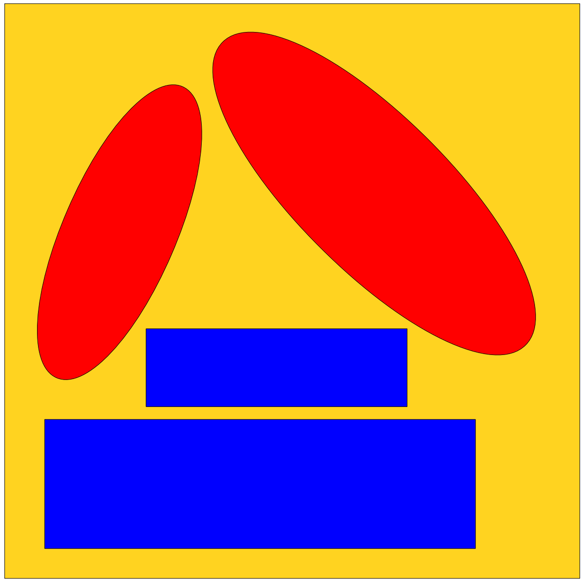
Example

- Domain: Tangram-Example
  - Each objects can be translated, rotated, scaled
  - Task: arrange all objects on the desk such that the desk is almost completely covered by the objects



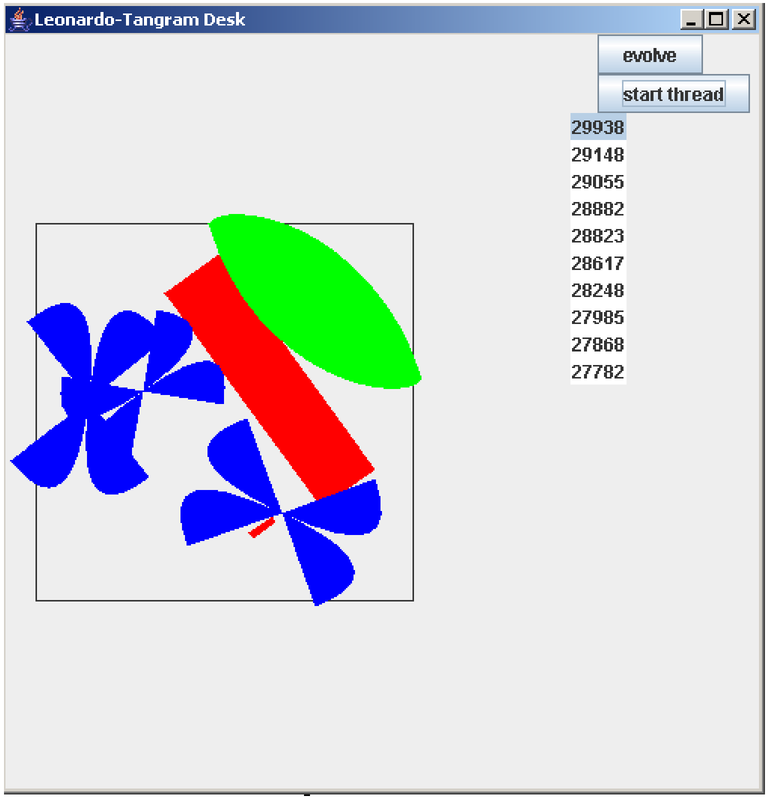
Encoding for Tangram-Domain

- One Gene
  - G={x,y,scale,rot}
- One individual
  - I={G;G;G;G} (for 4 geometric objects)
- The Population
  - P={I;I;I;I;I;I;I;I;....;I}
- Fitness function
  - Count the desk's hidden pixels

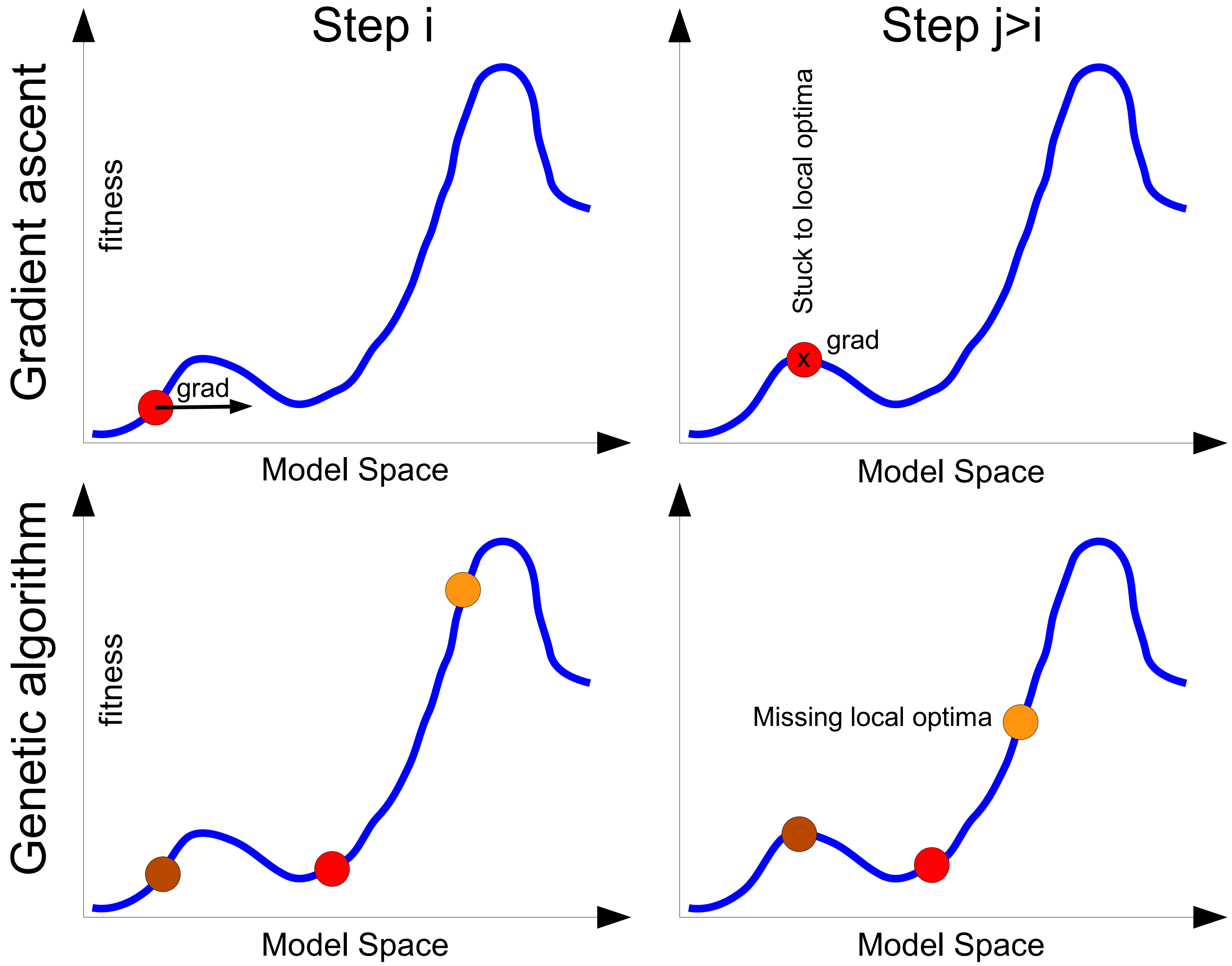


Practise

- Use the Tangram.jar Demo
  - Note the order of the individu-als from best to worst and see geometric representation
  - Watch what happens after one evolution (Turnament Selection+Elitism+Mutation)
  - A productive version is used to lay out electronic circuits as good as a human experts can do



Searching the model space



GA advantages

- In domains with many “step-like” dimensions
  - dimension-specific gradient descent does'nt work or is too slow
- As each organism is independent of others GA is suited for arallel processing in a cloud
- No specific criteria needed for the fitness function
  - Can be everything

GA Problems

- GA is is competitive to other algorithms only with good encoding
  - can reduce the processing time dramatically
- The search in the model space is not “directed” using a gradient, its just random -> slow convergence
  - (make a random walk and show how long it takes to get into the optimum)
- What is the difference to brute force?

Genetic programming

- Programs are encoded in Genes using a tree
- evolutionary operators are applied
- Very non-intuitive programs are created
- Problematic is the “porous” search space
  - More invalid solutions than valid ones