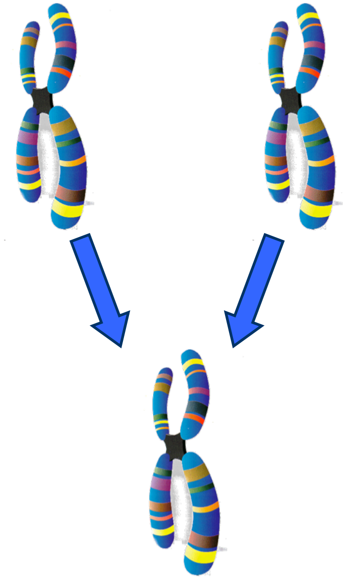
Genetic Algorithms: part II

How to improve the performance of your GA



Contents

- Genome representations
- Crossover
- Mutation
- Fitness assignments
- Different type of GA:s

Representations

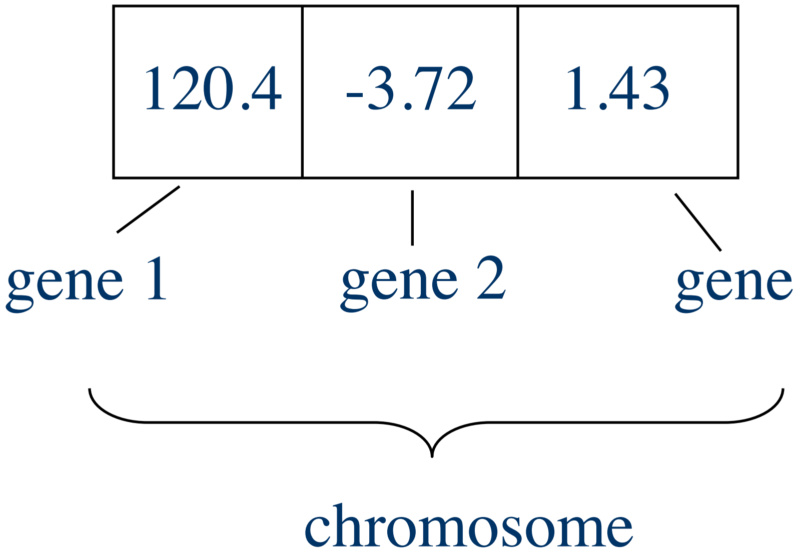
Let the representation be as close to the real problem as possible

- Real coded GA
- Mixed chromosomes
  - e.g. real and integer elements
- Multi-dimensional arrays
- Trees

Real number representation

If your problem concerns real numbers you are probably better of using a real encoding.

Three optimization variables: Real encoded



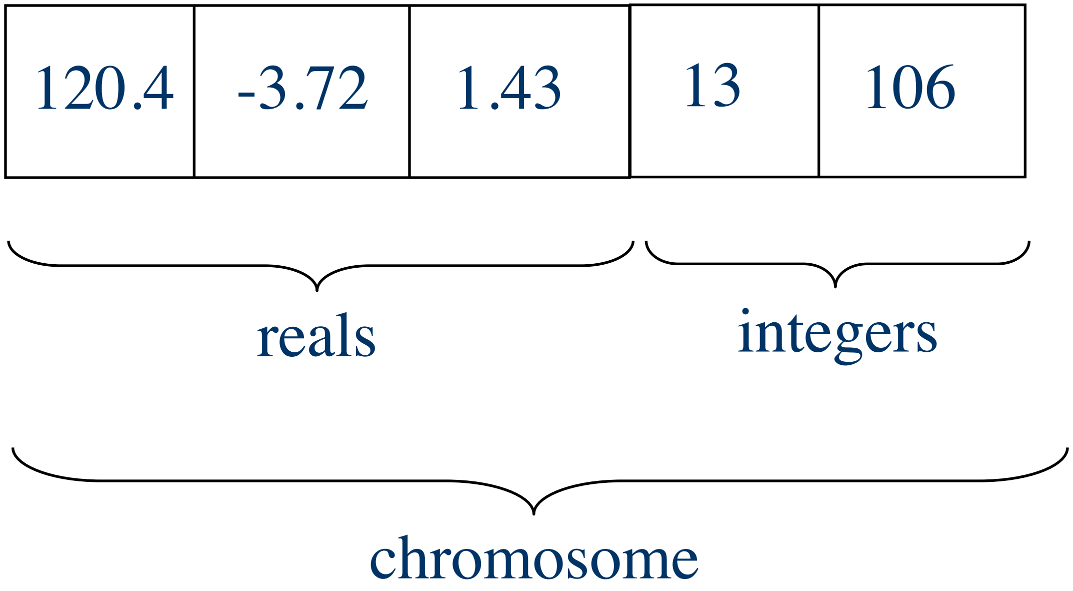
120.4 -3.72 1.43

gene 1 gene 2 gene 3

chromosome

Mixed real and integer representation

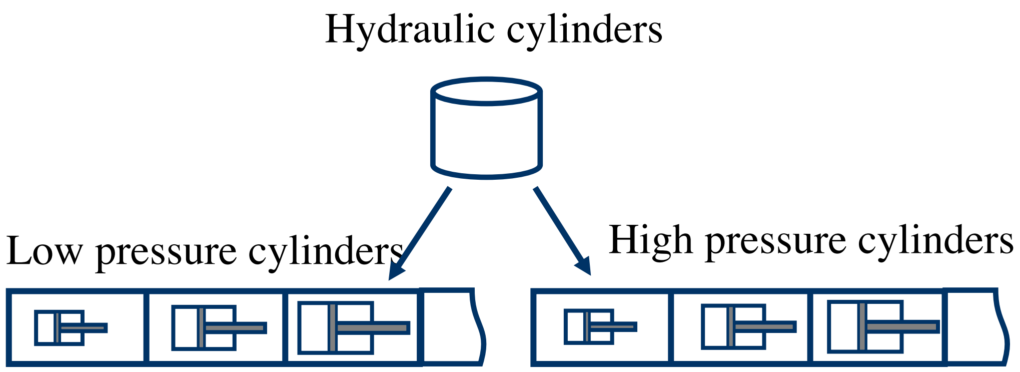
Three real variables and two integers



120.4	-3.72	1.43	13	106

Catalogs

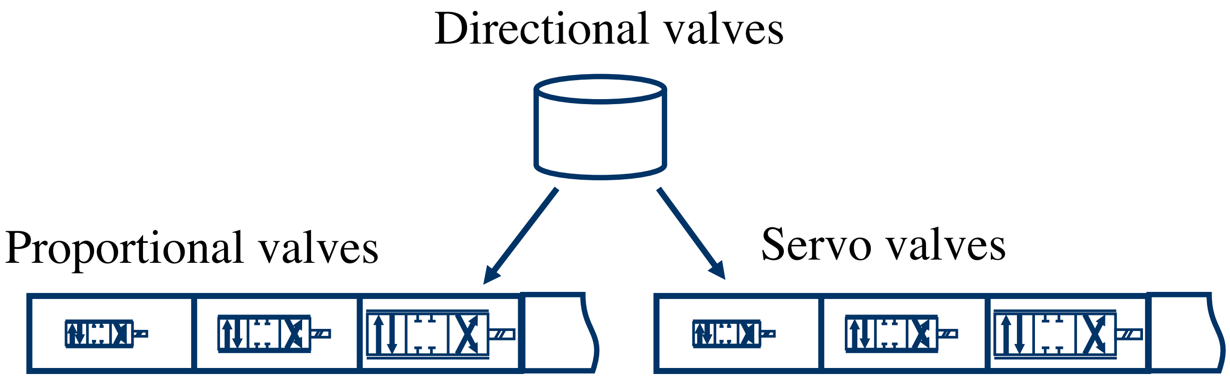
Discrete elements that are sorted into a hierarchy



Hydraulic cylinders

Low pressure cylinders

High pressure cylinders



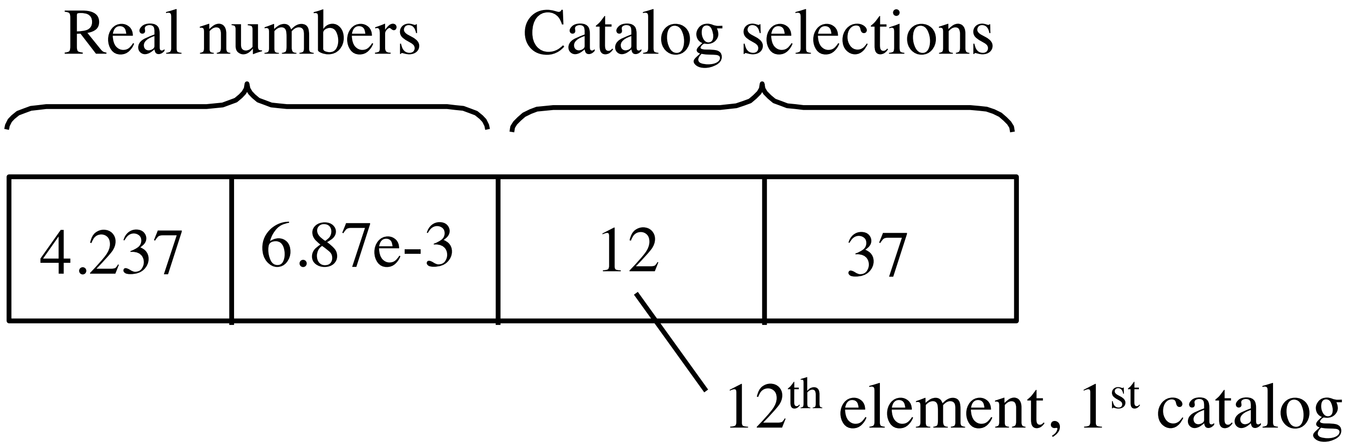
Directional valves

Proportional valves

Servo valves

Mixed real and catalog representation

Two real numbers and two catalog selections



Real numbers

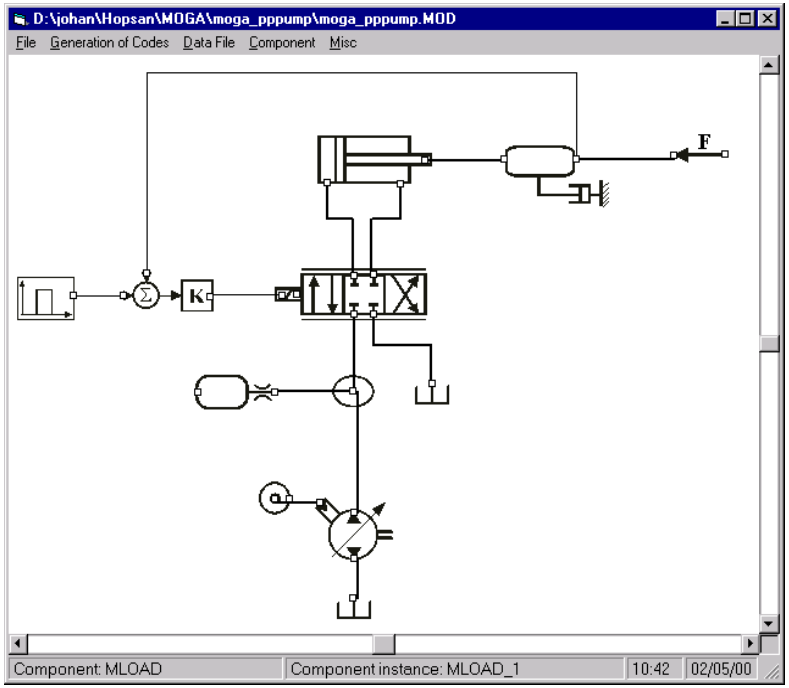
Catalog selections

4.237	6.87e-3	12	37

12th element, 1st catalog

Discrete optimization

Component databases



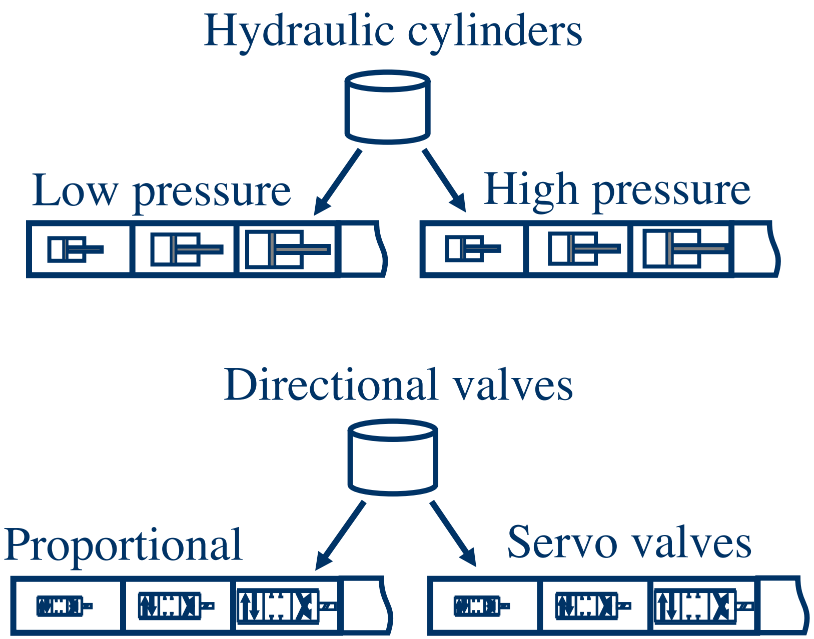
D:\johan\Hopsan\MOGA\moga_pppump\moga_pppump.MOD

Component: MLOAD

Component instance: MLOAD_1

10:42

02/05/00



Hydraulic cylinders

Low pressure

High pressure

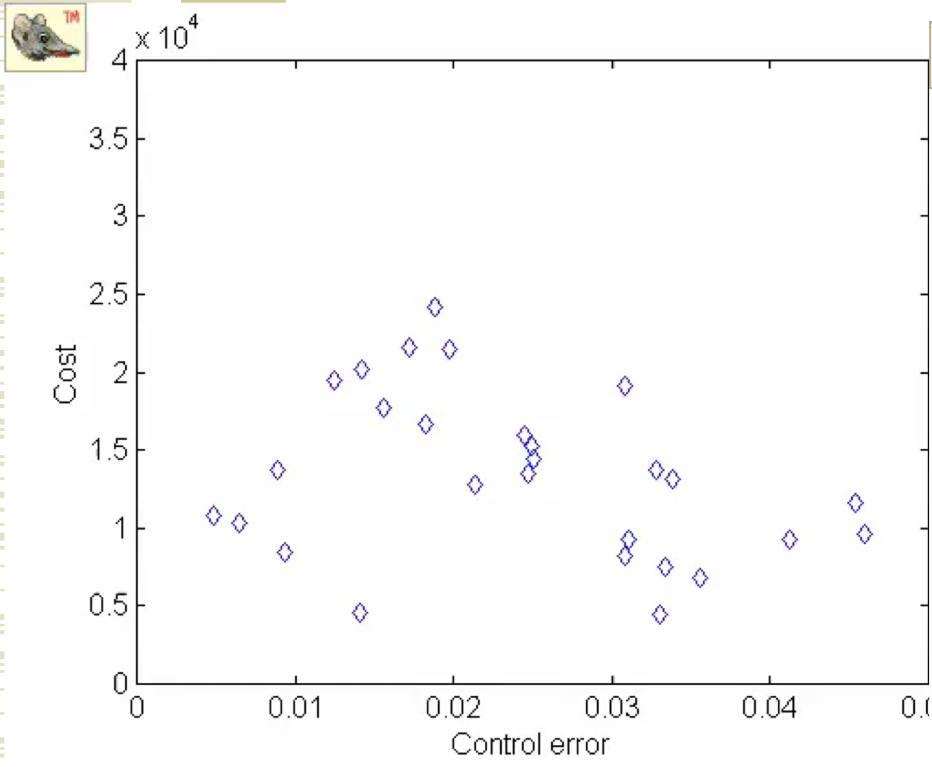
Directional valves

Proportional

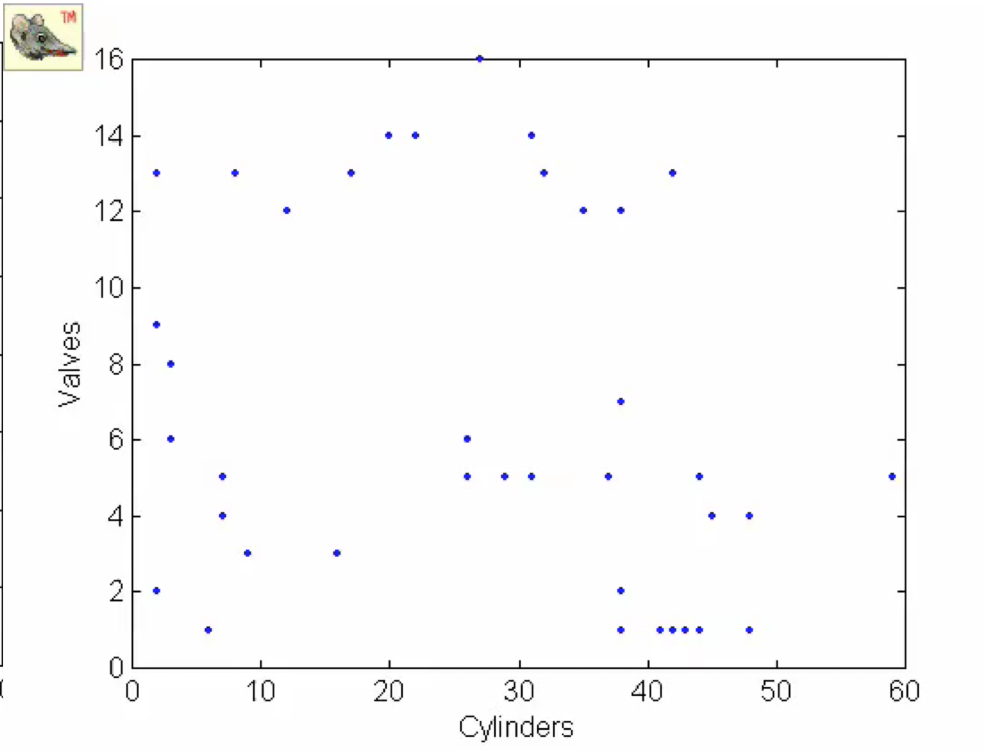
Servo valves

Valve system

The optimization progress

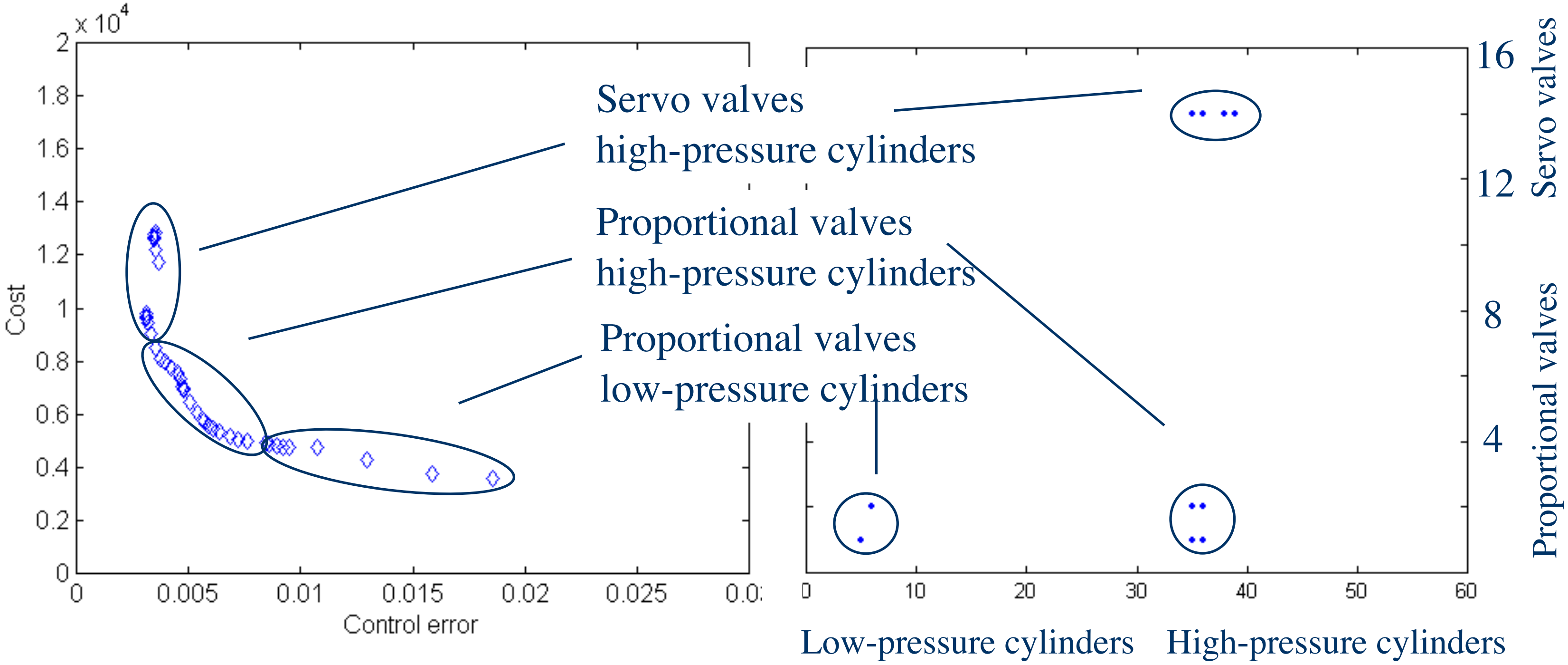


The population in objective space



The population in parameter space: Valves and Cylinders

Optimization result



Servo valves
high-pressure cylinders

Proportional valves
high-pressure cylinders

Proportional valves
low-pressure cylinders

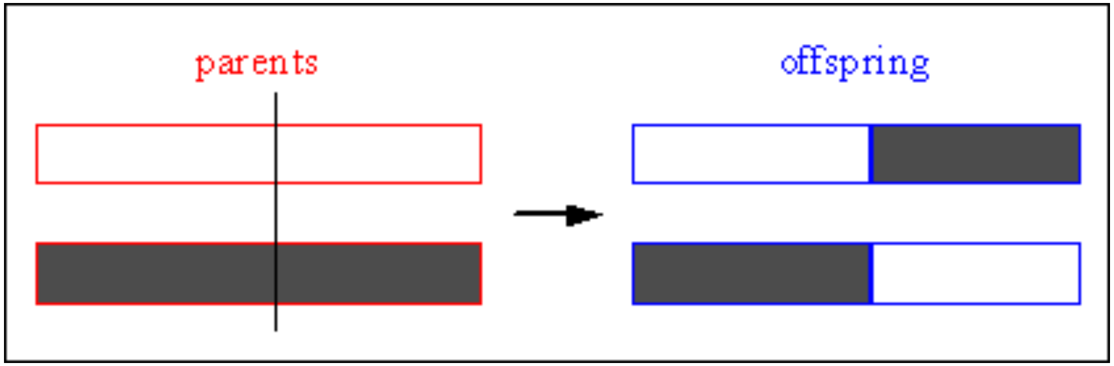
Pareto front

Valve and cylinder selection

Reproduction – crossover

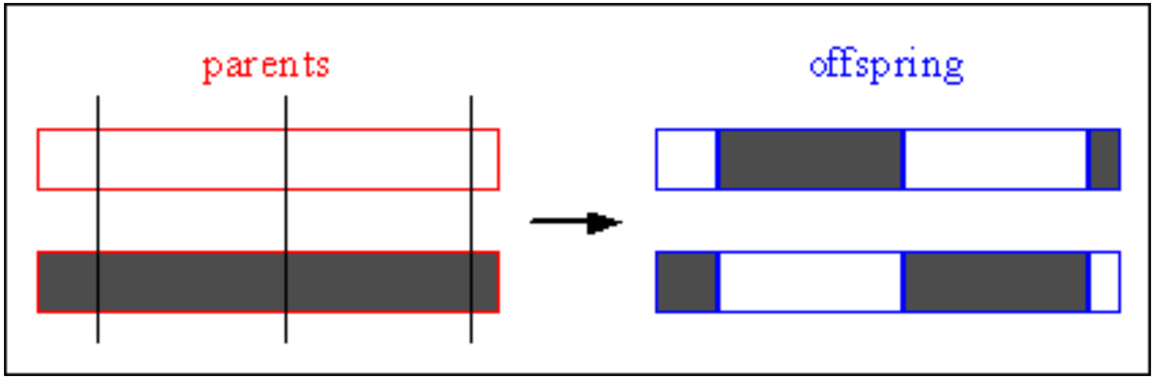
- The crossover operator should match the representation and the problem
- Make sure that only “valid” individuals can be represented and created.
- The crossover operator could be very crucial to the performance of the optimization

Example - array crossover



Single Point

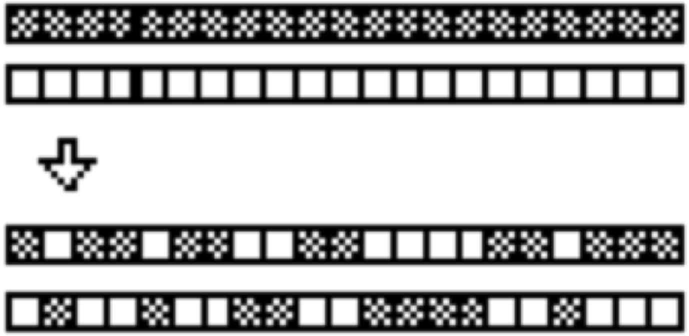
Crossover



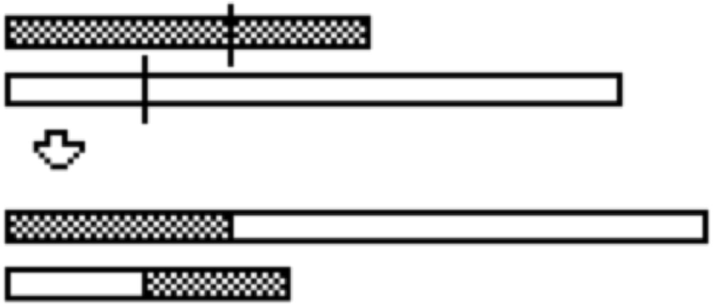
Multiple Point

Crossover

Example - array crossover

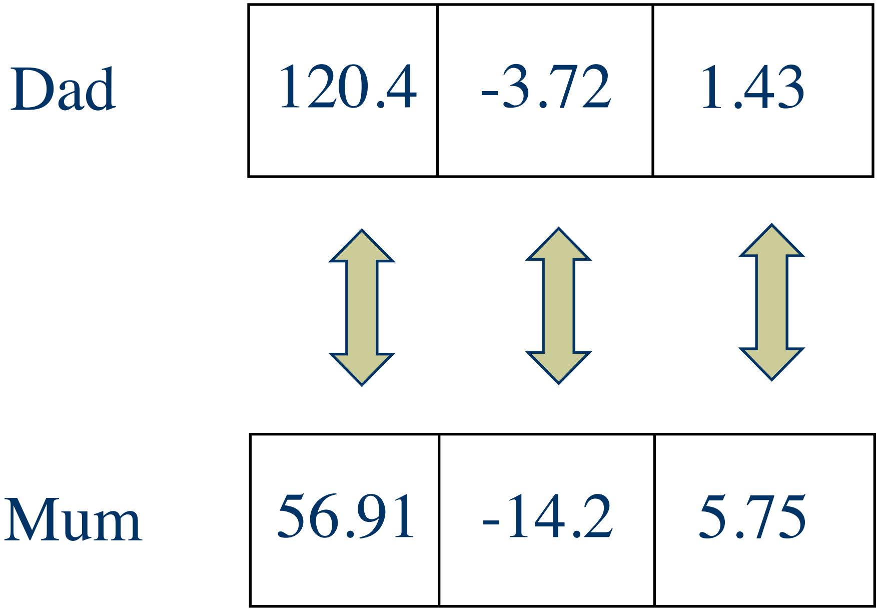


Array Uniform Crossover



Variable-Length Array
Single Point Crossover

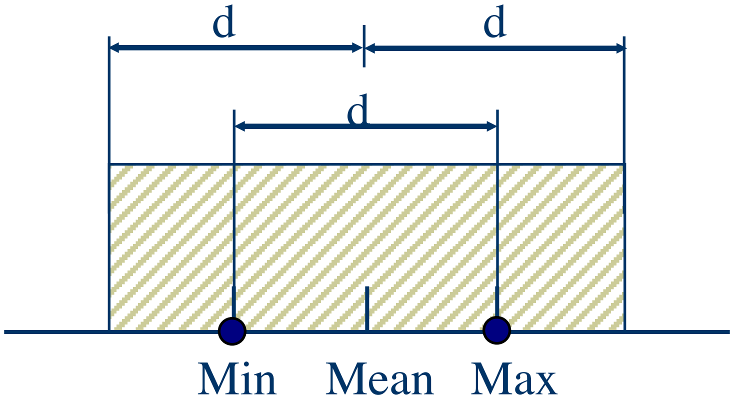
Uniform crossover



Dad	120.4	-3.72	1.43
Mum	56.91	-14.2	5.75

Each gene from one parent are combined with the corresponding gene from the other parent.

Blend crossover
(Extended intermediate crossover)

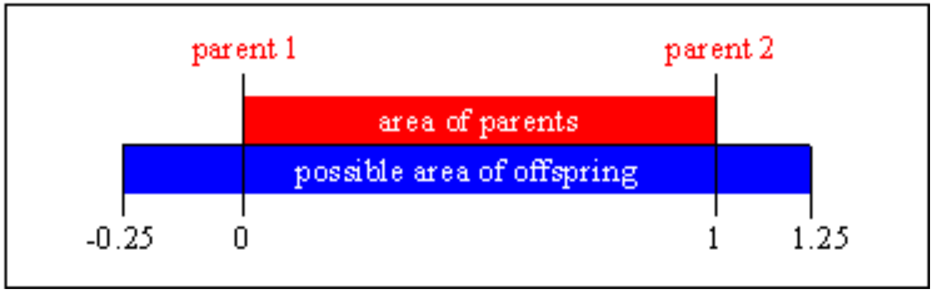


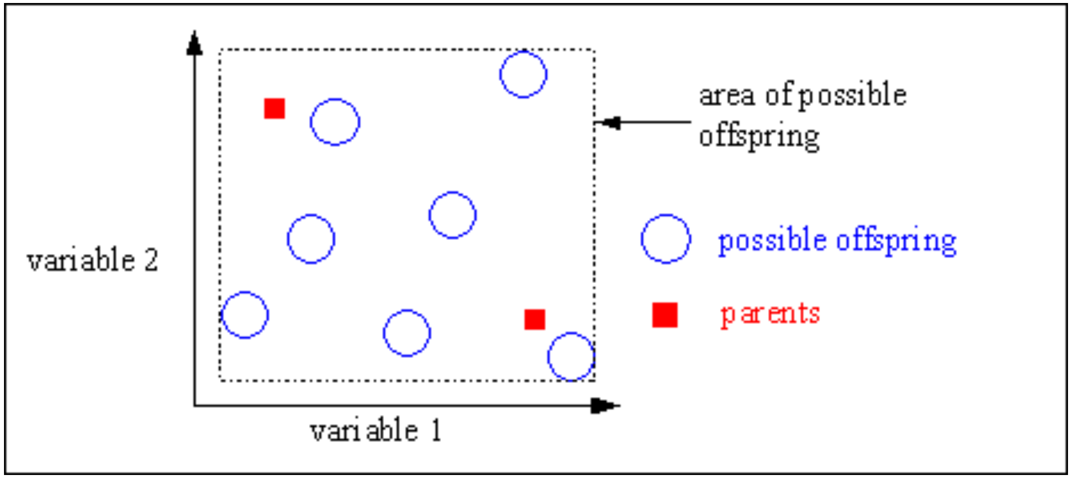
Dad	120.4	-3.72	1.43

Child	115.3	-4.42	0.52

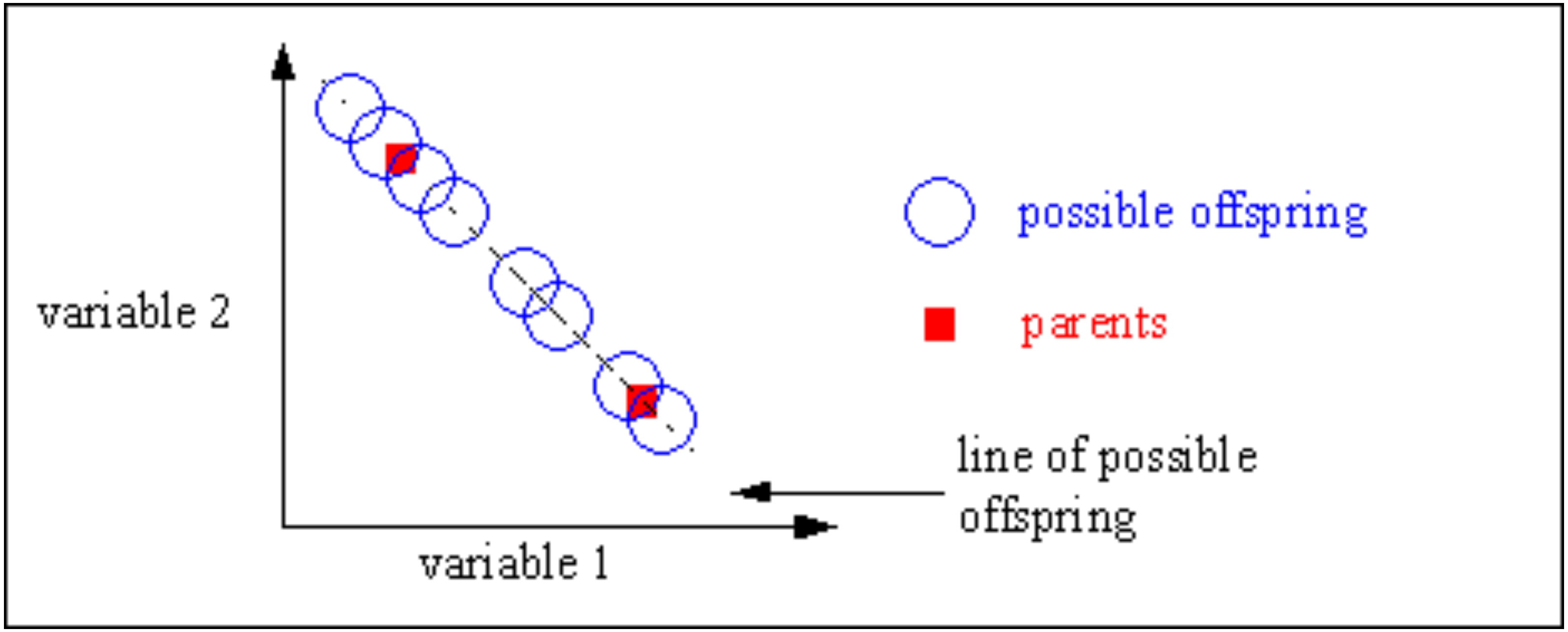
Mum	91.0	-4.32	0

Blend crossover cont.

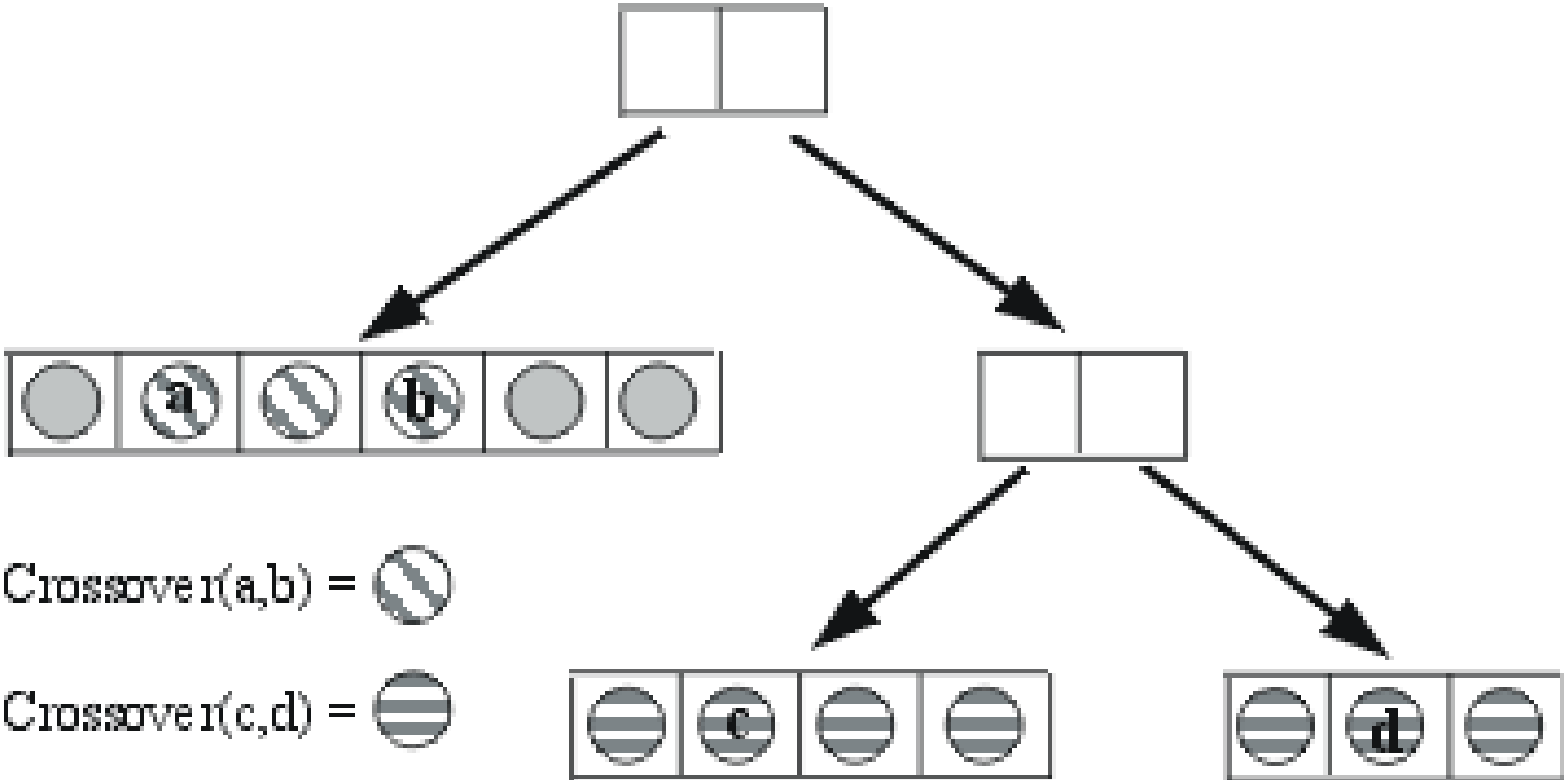




Line recombination



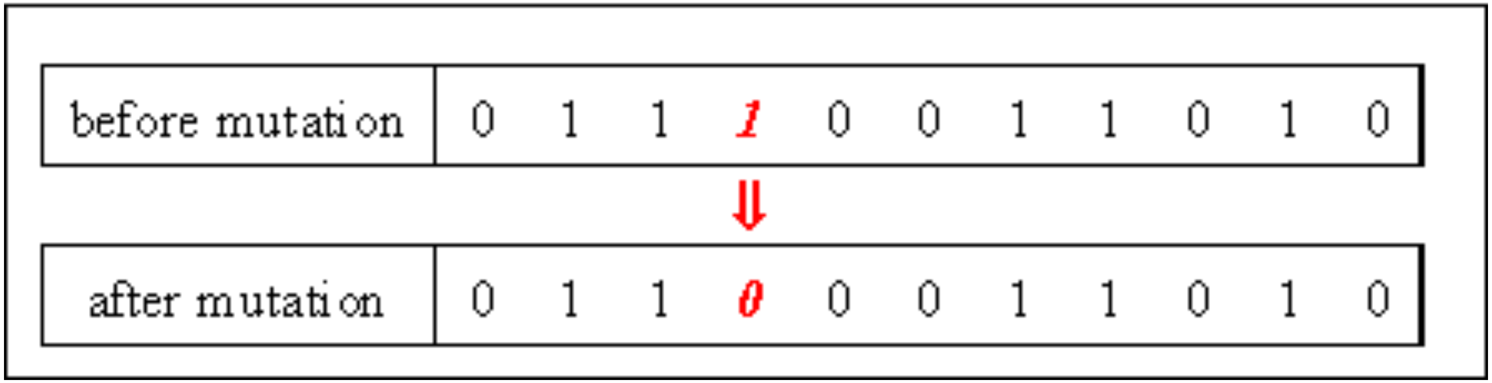
Catalog crossover



Reproduction – mutation

- Mutation is also dependent on the representation.
- Depending on the crossover, there might be no need for mutation.
- Examples:
    - Binary mutation
    - Gausian mutator

Example mutations

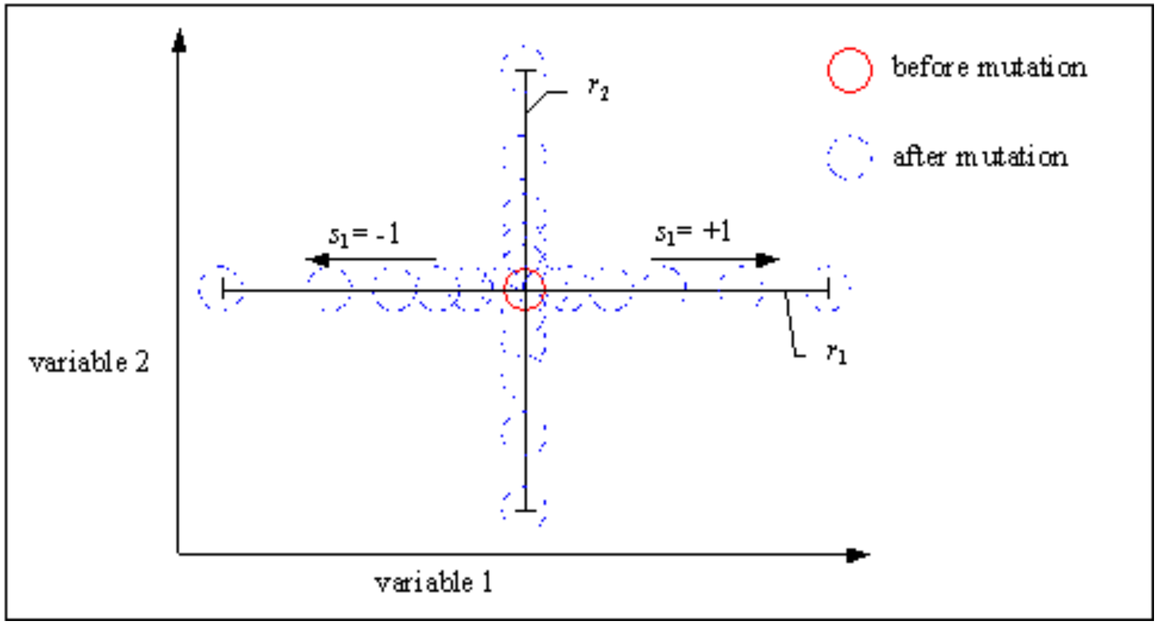


before mutation	0 1 1 1 0 0 1 1 0 1 0
	⇓
after mutation	0 1 1 0 0 0 1 1 0 1 0

Binary mutation

For each bit there is the probability *pmut* that it is changed

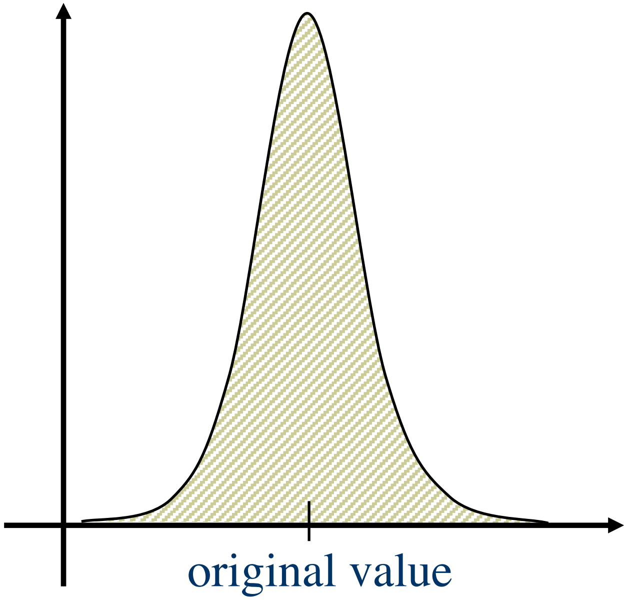
Example mutations



Real number mutation

Large probability for small mutation step and low probability for large steps

Example mutations



Gaussian mutator

Problems a GA might encounter

- Premature convergence
- Slow convergence
- The optimization problem could have:
  - Multiple optima
  - Isolated optima
  - Noisy objective function
  - Deceptive

Fitness assignment

How objective score becomes fitness values

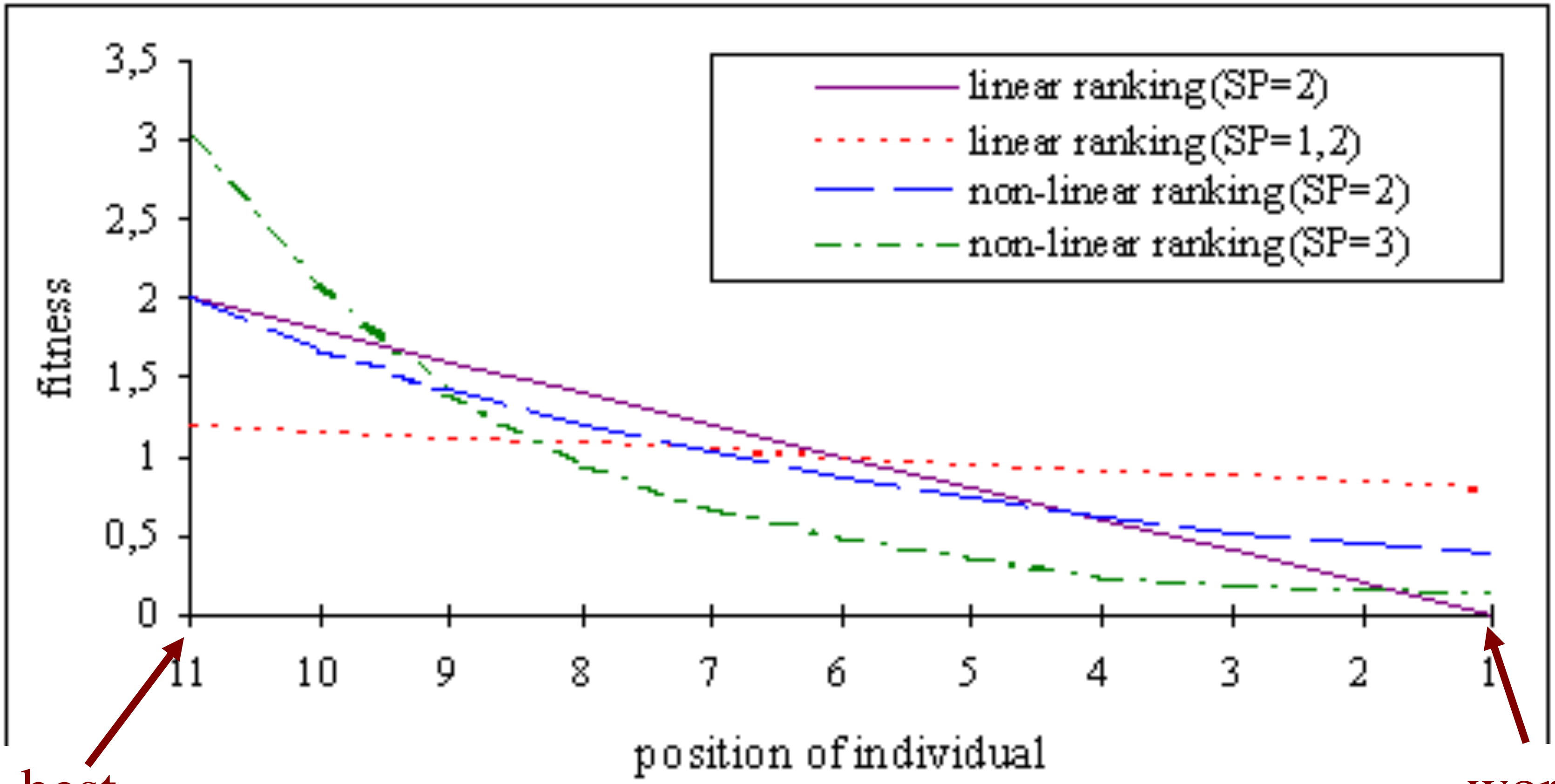
- In the start of a GA run the extraordinary may take over => premature convergence
- In the end of the run we might have genetic difference but similar fitness => slow convergence

Solution: Rank-based fitness assignment

Rank based fitness assignment

		fitness value (parameter: selective pressure, SP) SP=2 & 1.1 for linear and 3 and 2 for non-linear				
		No ranking	linear ranking		non-linear ranking	
Individual no.	objective value	1,0	2,0	1,1	3.0	2.0
1	1	1,0	2.0	1.10	3.00	2.00
2	3	1,0	1.8	1.08	2.21	1.69
3	4	1,0	1.6	1.06	1.62	1.43
4	7	1,0	1.4	1.04	1.99	1.21
5	8	1,0	1.2	1.02	0.88	1.03
6	9	1,0	1.0	1.00	0.65	0.87
7	10	1,0	0.8	0.98	0.48	0.74
8	15	1,0	0.6	0.96	0.35	0.62
9	20	1,0	0.4	0.94	0.26	0.53
10	30	1,0	0.2	0.92	0.19	0.45
11	95	1,0	0.0	0.90	0.14	0.38

Fitness assignment

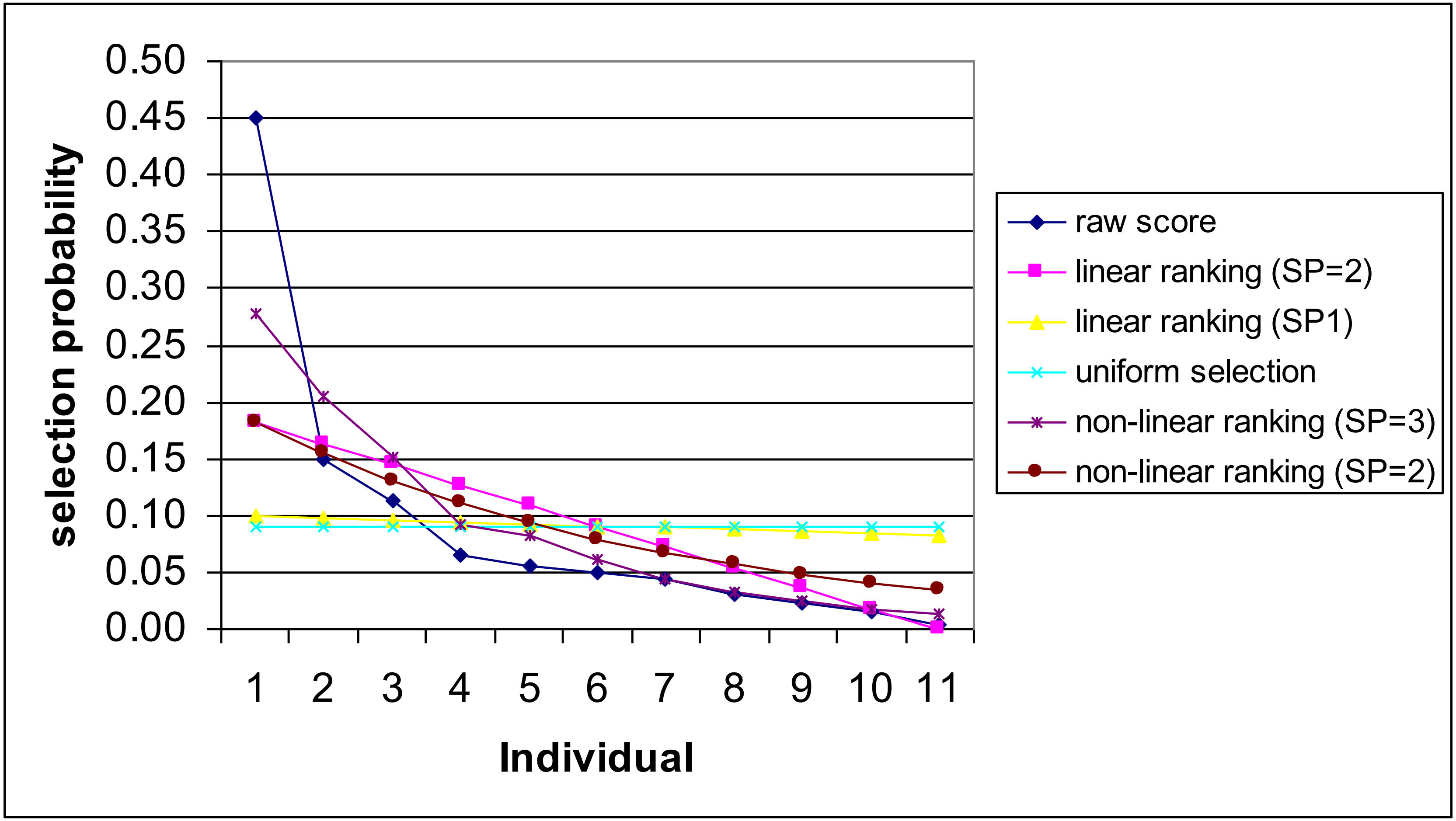


best ——— worst

Selection probability

		Probability of selection					
		Based on raw objective score	No ranking	linear ranking		non-linear ranking	
Individual no	objective value		1,0	2,0	1,1	3	2
1	1	0.45	0,09	0.18	0.10	0.28	0.18
2	3	0.15	0,09	0.16	0.10	0.21	0.15
3	4	0.11	0,09	0.15	0.10	0.15	0.13
4	7	0.06	0,09	0.13	0.09	0.09	0.11
5	8	0.06	0,09	0.11	0.09	0.08	0.09
6	9	0.05	0,09	0.09	0.09	0.06	0.08
7	10	0.04	0,09	0.07	0.09	0.04	0.07
8	15	0.03	0,09	0.05	0.09	0.03	0.06
9	20	0.02	0,09	0.04	0.09	0.02	0.05
10	30	0.01	0,09	0.02	0.08	0.02	0.04
11	95	0.00	0,09	0.00	0.08	0.01	0.03
	SUM	1	1	1	1	1	1

Selection probability



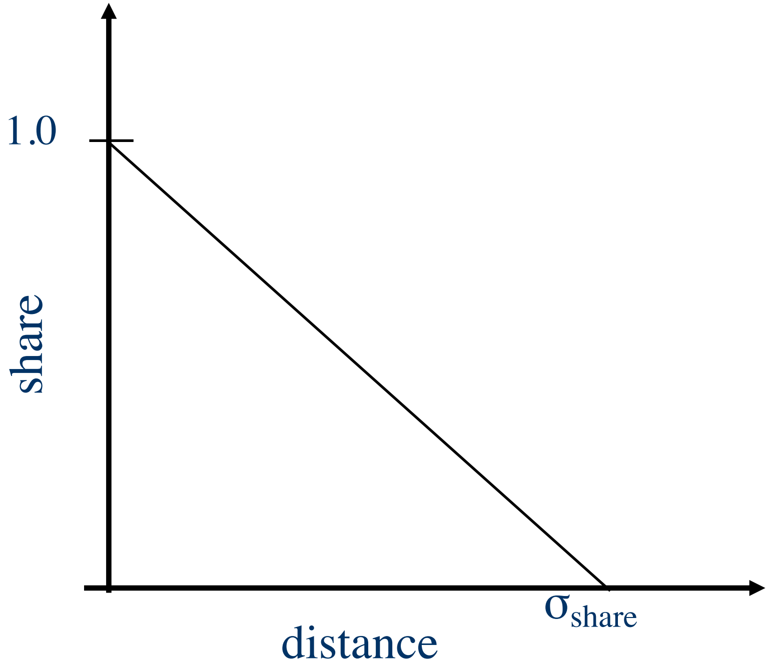
Niche and speciation

Ability to detect and maintain multiple optima

- Fitness assignment
    - sharing
- Replacement strategies
    - crowding
    - struggle
- Multiple populations

Fitness sharing

Individuals close to each other have to share fitness



$$f_{share}(\mathbf{x}_i) = \frac{f(\mathbf{x}_i)}{\sum_{j=1}^{n} s\left(d\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)}$$

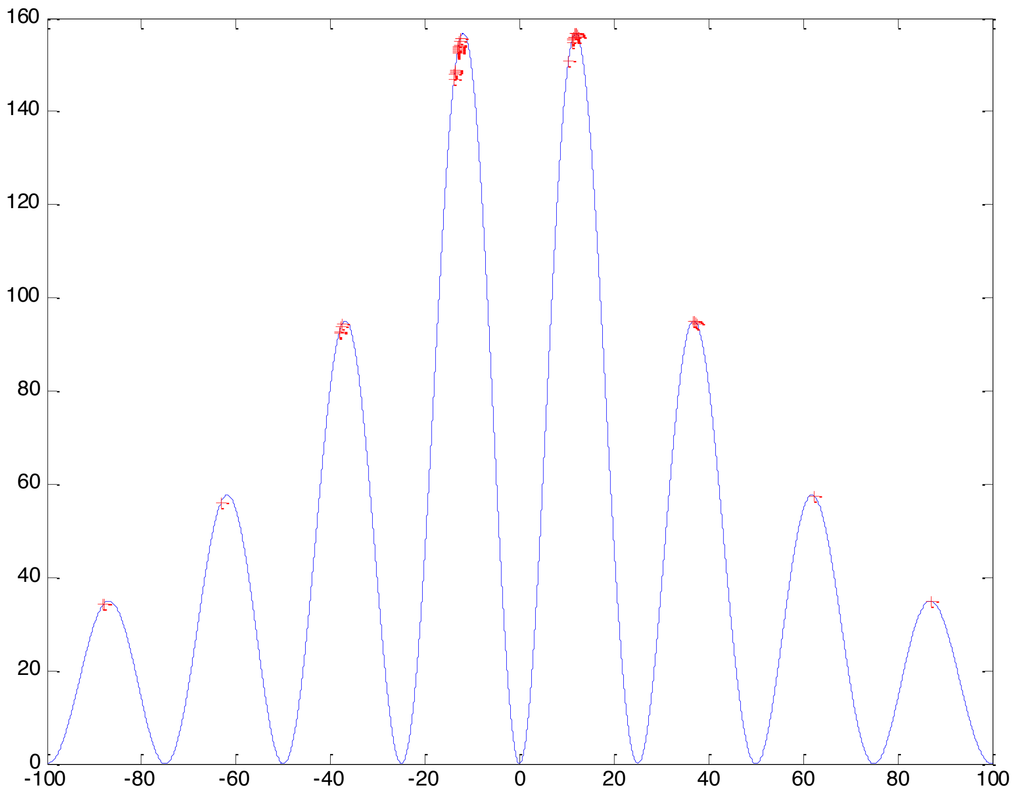
s = sharing function

d = distance function

Triangular sharing function according to Goldberg

Example - fitness sharing

A multimodal function where the GA have identified multiple peeks



Distance functions

Many operators needs a distance function to determine similarity

- Genotype distance
  - how many bits differ
- Phenotype distance
  - Euclidean distance of decoded values
- Attribute distance
  - Euclidean distance between the objective values

Genotype

This is the "internally coded, inheritable information" carried by all living organisms. This stored information is used as a "blueprint" or set of instructions for building and maintaining a living creature.

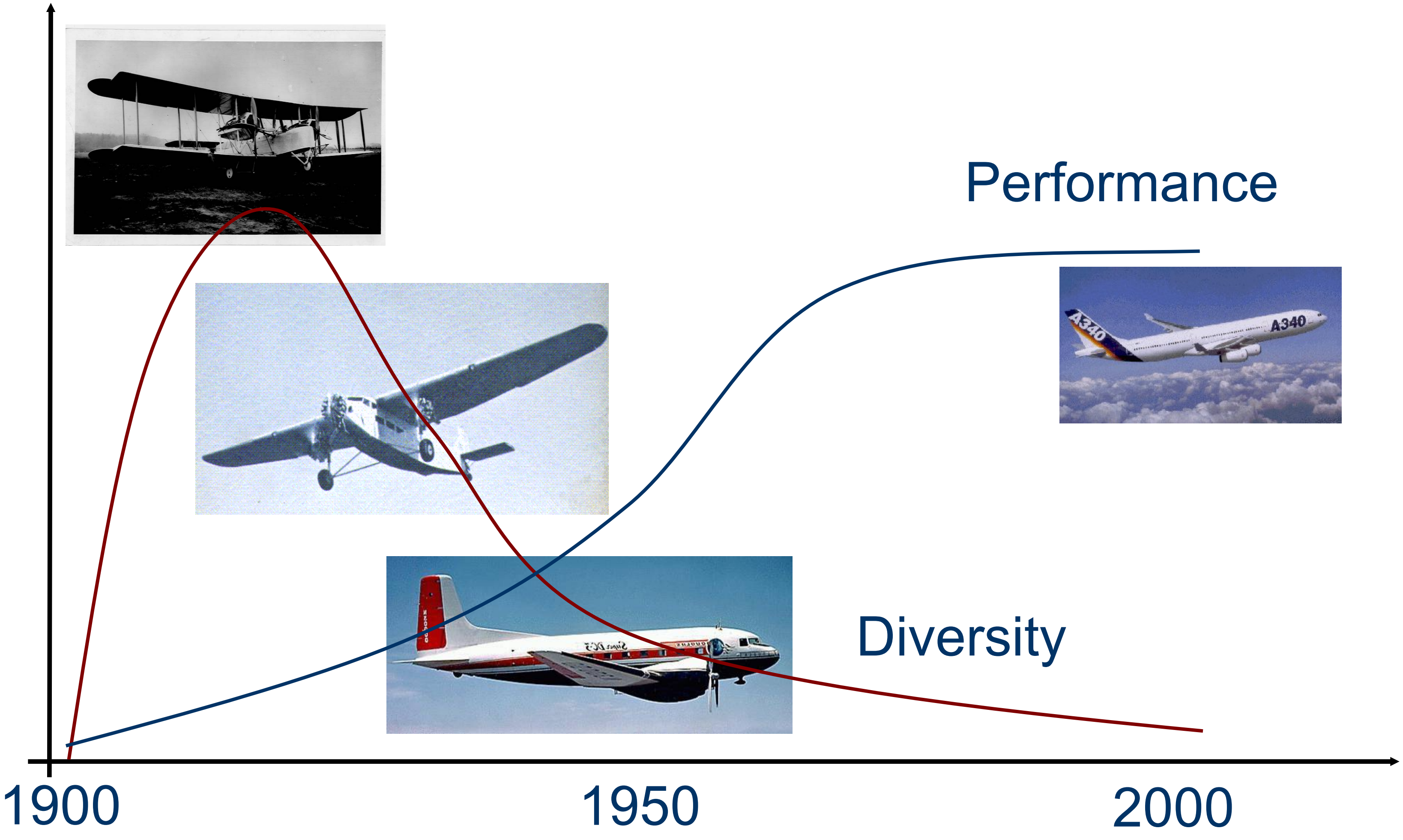
Phenotype

This is the "outward, physical manifestation" of the organism. These are the physical parts, anything that is part of the observable structure, function or behaviour of a living organism.

Richard Dawkins

- Evolutionary pattern resembling biological evolution can be found in technical development
- Culture is a continuation of biological evolution

Evolution of passenger aircraft



Distance functions

Many operators needs a distance function to determine similarity

- Genotype distance
  - how many bits differ
- Phenotype distance
  - Euclidean distance of decoded values
- Attribute distance
  - Euclidean distance between the objective values

Distance functions:
Attribute/objective space

The distance between two individuals *a* and *b* is calculated as the Euclidean distance in attribute space

$$\text{Distance}(a,b) = \sqrt{\sum_{i=1}^{k} \left(\frac{f_{ia} - f_{ib}}{f_{i\max} - f_{i\min}} \right)^2 \frac{1}{k}}$$

Distance functions:
Real numbers

The distance between two real numbers is calculated as the normalized Euclidean distance.

$$\text{Distance}(a,b) = \sqrt{\left(\frac{a-b}{\text{max distance}}\right)^2}$$

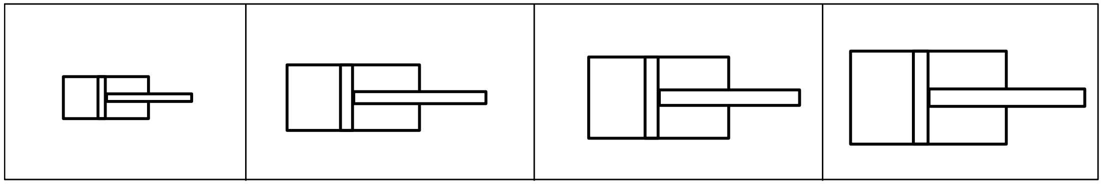
Distance functions:
Mixed chromosome distance

The overall distance between two individuals *a* and *b* is calculated as the sum of the distances between all design variables.

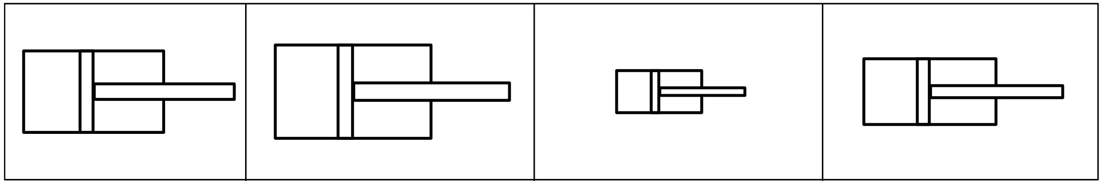
$$\text{Distance}(a,b)=\sum_{i=1}^{n}\frac{\text{Distance}(\text{DV}_{\text{i}})}{n}$$

Distance functions:
Catalog distance

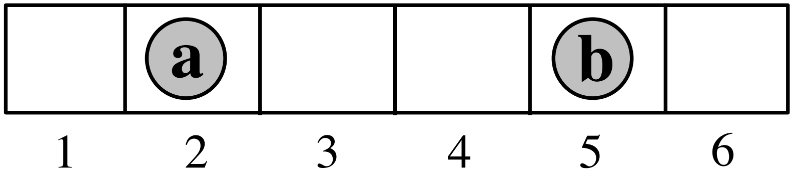
Ordered catalog of hydraulic cylinders



Unordered catalog of hydraulic cylinders



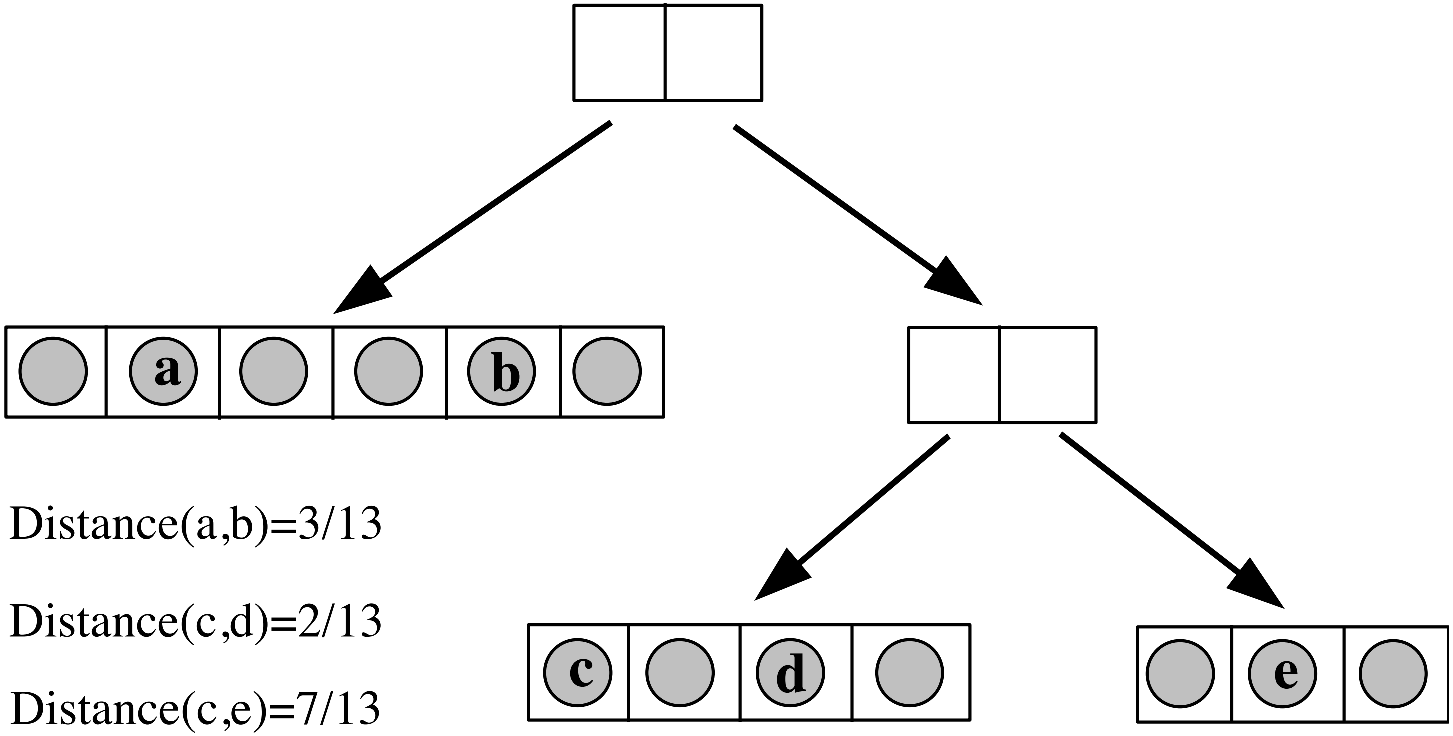
Distance(a,b)=3/5



$$\text{Distance}(a,b) = \frac{pos(a) - pos(b)}{\text{max distance}}$$

Distance functions:
Hierarchical catalog

The distance between two catalog elements is based on the smallest common hierarchy.



Distance(a,b)=3/13

Distance(c,d)=2/13

Distance(c,e)=7/13

Replacement strategies

- Simple
  - The whole population is replaced each generation
  - Not like in nature
- Steady state
  - A certain percentage of the population is replaced
- Incremental
  - One or two children are inserted into the population each time.
  - The children replaces the worst, there parents, a random individual, the most similar etc…

Replacement strategies *cont*.

- produce as many offspring as parents and replace all parents by the offspring (pure reinsertion).

Generation gap

- produce less offspring than parents and replace parents uniformly at random (uniform reinsertion).
- produce less offspring than parents and replace the worst parents (elitist reinsertion).

- produce more offspring than needed for reinsertion and reinsert only the best offspring (fitness-based reinsertion).

DeJong style Crowding

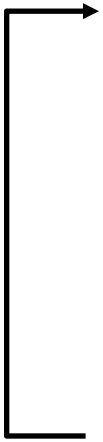
Avoid genetic drift by performing replacement based on similarity measures

- Select parents (according to fitness)
- Create children
- Compare the children to a random subpopulation of CF members.
- Let the children replace the most similar individual in CF

DeJong 1975

Struggle crowding

Let the children compete with the most similar individual in the entire population



- Select parents (randomly)
- Create one child
- Find the most similar individual in the entire population.
- Let the child replace the most similar individual.

Grüninger and Wallce, 1996

Other “natural” tweaks

- Diploidy (in contrast to haploid chromosomes)
    - two pair of chromosomes
    - dominance
- Sex
- Restricted mating

Hybrid algorithms

- Additional operators
  - Complex mutation
- Pipelining hybrids
  - First GA – then Complex or a derivative method
- Hierarchical hybrids
  - GA on a system level – derivative on low level
- Asynchronous hybrids
  - Different methods on different sub areas

GA pros and cons

- It requires many function evaluations – computational expensive

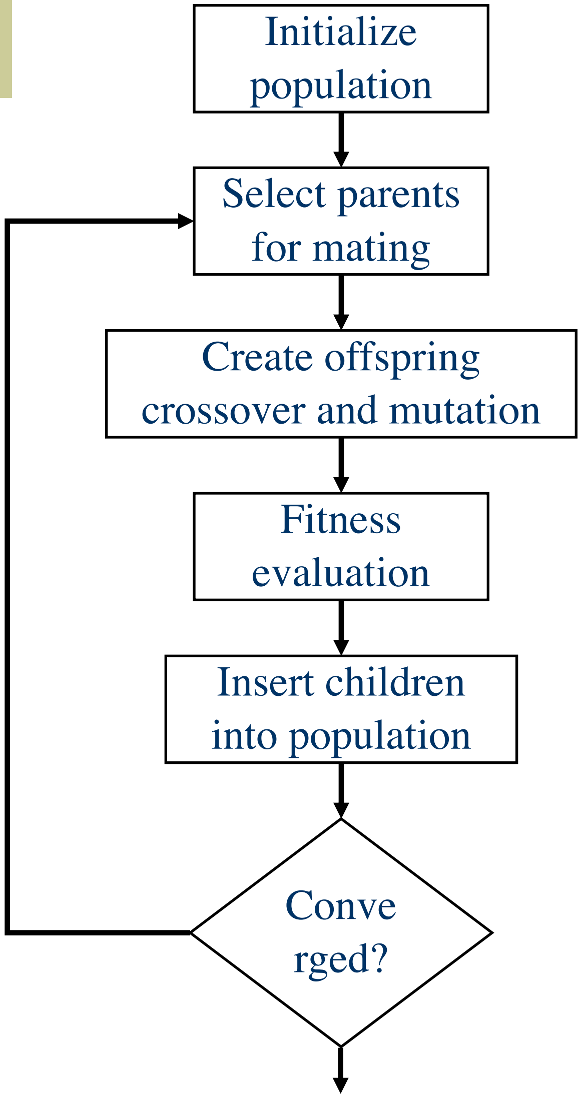
- We can not guarantee that we have found the true optima (you can very seldom do that anyhow)

+ It is more likely to find the global optima on hard problems than many other methods.

+ It could be applied to a very large range of problems

+ As it uses a population of individuals it can identify multiple solutions in one optimization run. Good for **multi modal** functions and **multi objective** optimization.

GA – the principle revisited



Initialize population

Select parents for mating

Create offspring crossover and mutation

Fitness evaluation

Insert children into population

Conve rged?

representation, *popsize*

selection method

crossover method, *pcross*

mutation method, *pmut*

fitness assignment

replacements strategy, *gengap*

termination criteria, *nogen*, *convergence*, *stagnation*

Final comments

- ➢ There are many GAs out there
- ➢ There is no “best” GA for all problems
- ➢ The operators are problem specific
- ➢ The GA parameters are problem specific
  - Optimization of the GA parameters
- ➢ This has just been a scratch on the surface