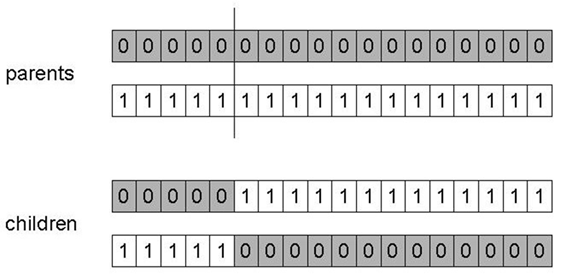
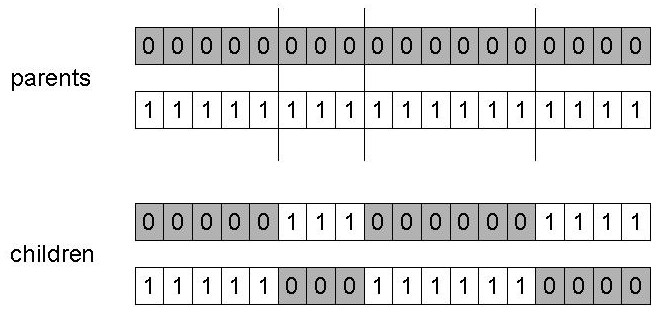


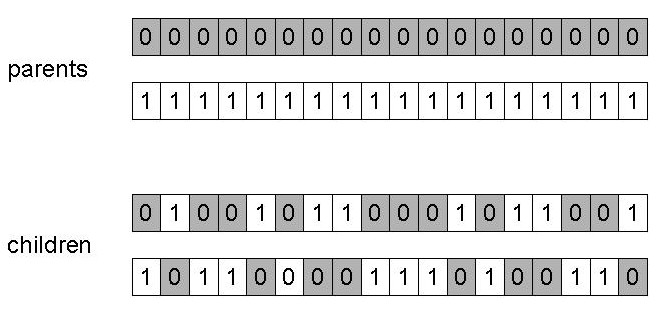
* + Binary strings : Genetic Algorithms
  + Real-valued vectors : Evolution Strategies
  + Finite State Machines: Evolutionary Programming
  + LISP trees: Genetic Programming

Binary Representation

Bit-flip mutation







**Exploration:** Discovering promising areas in the search space, i.e. gaining information on the problem

**Exploitation:** Optimising within a promising area, i.e. using information

**Integer Representation**

N-point / uniform crossover

random resetting

**Real-Valued Representation**

Uniform Mutation

Nonuniform Mutation

x’i = xi + N(0,σ)

σ: *mutation step size*

Self-Adaptive Mutation

〈 x1,…,xn, σ 〉

Uncorrelated mutation with one σ

σ’ = σ • exp(τ • N(0,1))

x’i = xi + σ’ • Ni(0,1)

learning rate τ ∝ 1/ n½

σ’ < ε0 ⇒ σ’ = ε0

Uncorrelated mutation with n σ’s

σ’i = σi • exp(τ’ • N(0,1) + τ • Ni (0,1))

x’i = xi + σ’i • Ni (0,1)

τ’ overall learning rate

τ coordinate wise learning rate

τ’ ∝ 1/(2 n)½  and τ ∝ 1/(2 n½) ½

σi’ < ε0 ⇒ σi’ = ε0

**Covariance Matrix Adaptation Evolution Strategy** **(CMA-ES)**

Correlated mutations

〈 x1,…,xn, σ1,…, σn ,α1,…, αk 〉

k = n • (n-1)/2

Covariance matrix:

cii = σi2

cij = 0 if i and j are not correlated

cij = ½•(σi2 - σj2 ) •tan(2 αij) if i and j are correlated

σ’i = σi • exp(τ’ • N(0,1) + τ • Ni (0,1))

α’j = αj + β • N (0,1)

*x* ’ = *x* + *N*(*0,C’*)

**x:** the vector 〈 x1,…,xn 〉

**C’**: the covariance matrix **C** after mutation of the α values

τ ∝ 1/(2 n)½  and τ ∝ 1/(2 n½) ½ and β ≈ 5°

σi’ < ε0 ⇒ σi’ = ε0 and

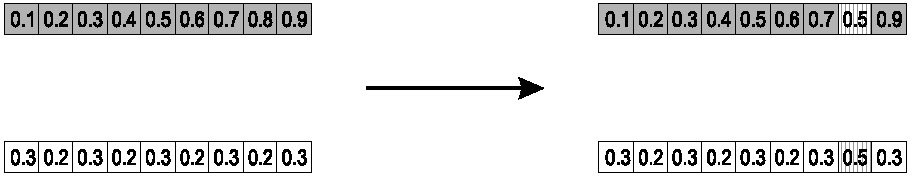
| α’j | > π ⇒ α’j =α’j - 2 π sign(α’j)

Recombination

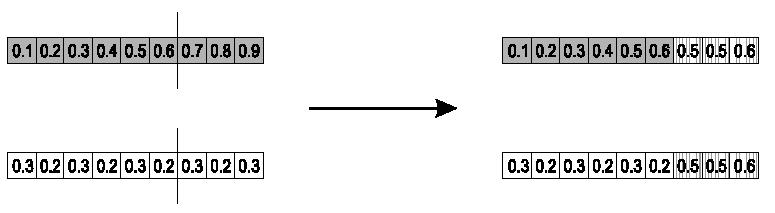
Discrete: n-point / uniform

Intermediate: *zi = α xi  +* (1 - *α) yi* where *α :* 0 ≤ *α* ≤1.

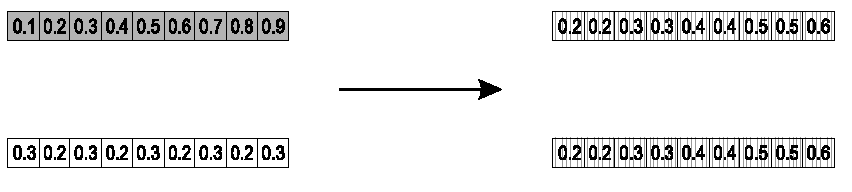
Single arithmetic crossover



Simple arithmetic crossover



Whole arithmetic crossover

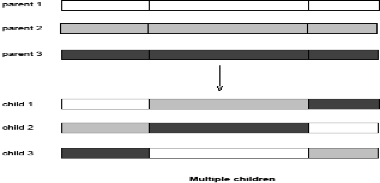


Blend Crossover

* Parents: 〈x1,…,xn 〉 and 〈y1,…,yn〉
* Assume xi < yi
* di = yi – xi
* Random sample zi= [xi, yi]
* Random sample zi= [xi – αdi, yi + αdi]

Original authors had best results with α = 0.5

Multi-parent recombination, type 1



Multi-parent recombination, type 2 (intermediary recombination)

* *i*-th allele in child is the average of the parents’ *i*-th alleles

**Permutation Representations**

Swap mutation



Insert mutation



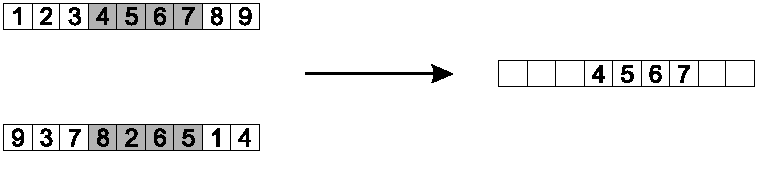
Scramble mutation

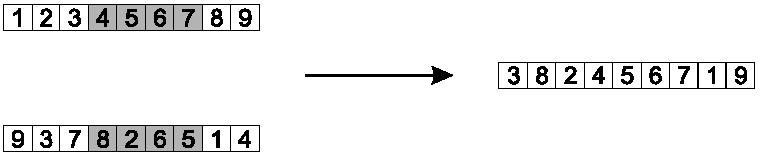


Inversion mutation

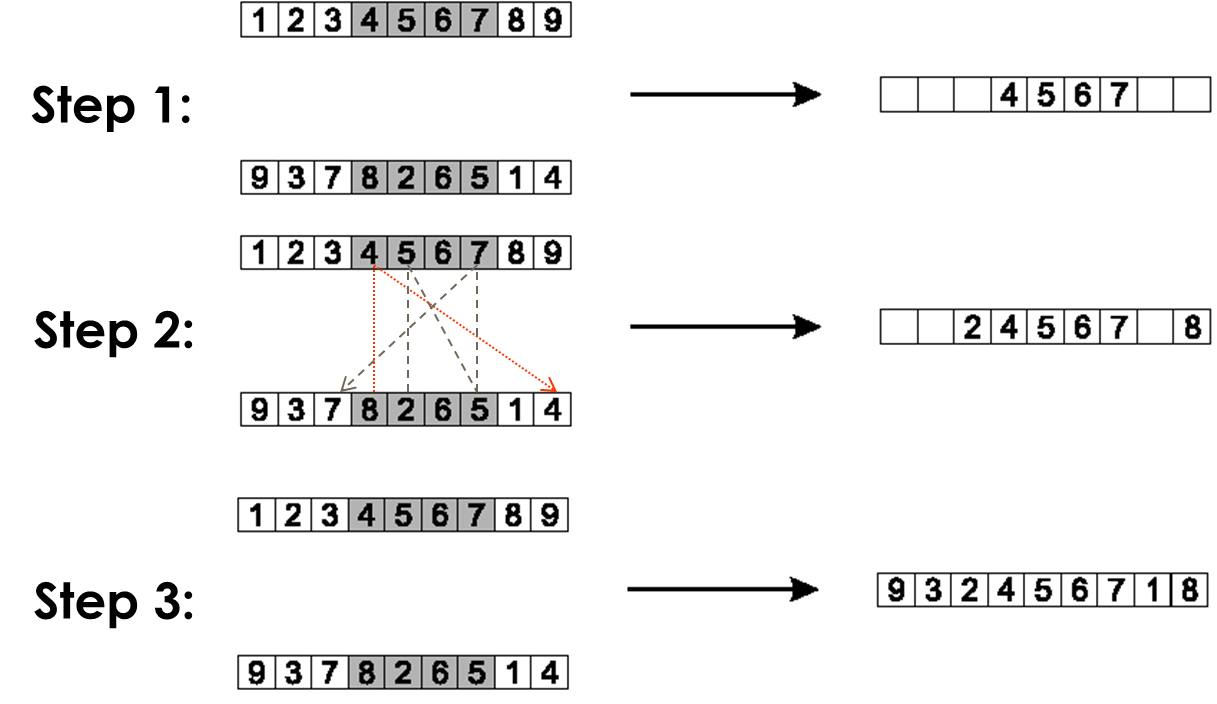


Order 1 crossover

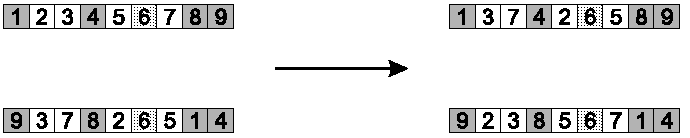
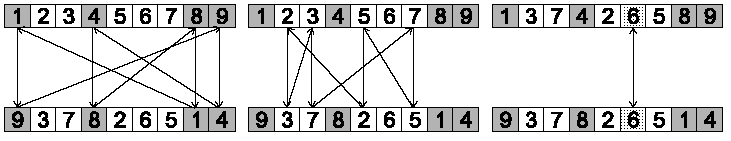




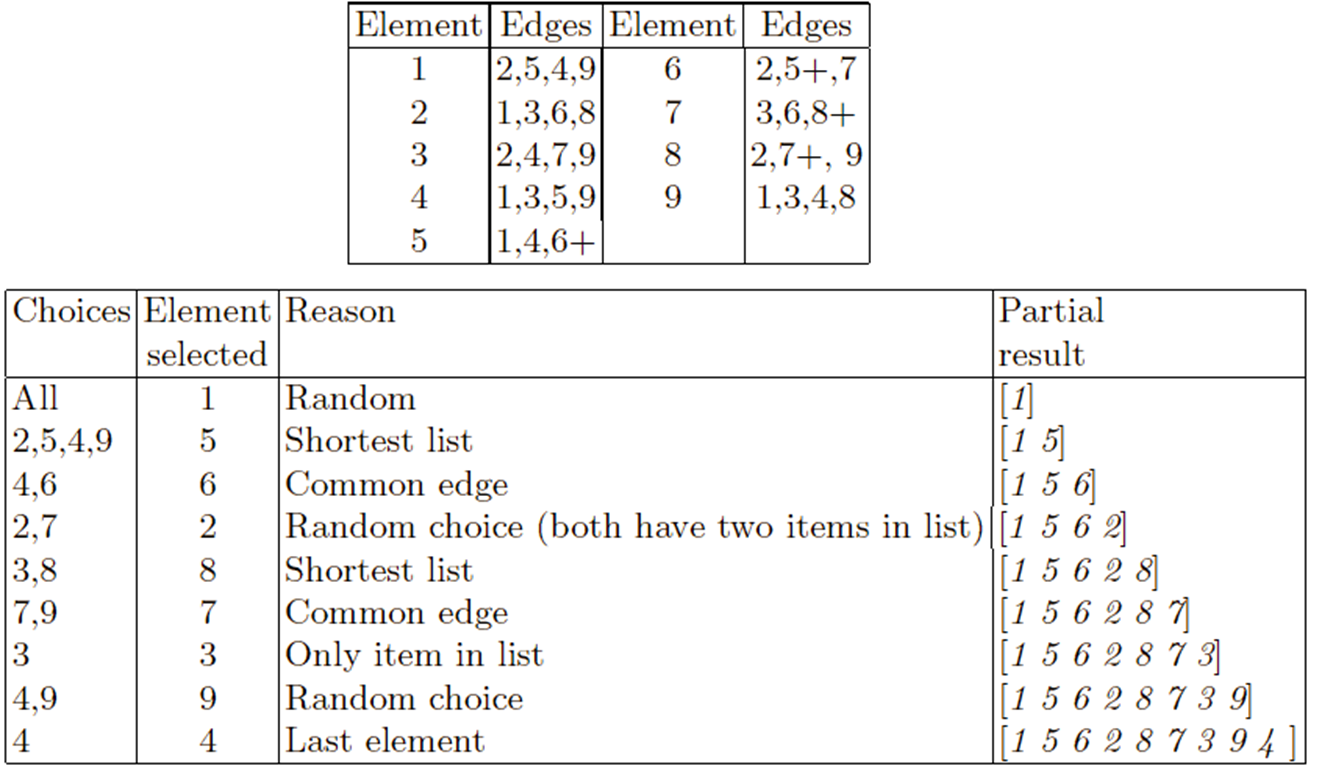
Partially Mapped Crossover (PMX)



Cycle crossover



Edge recombination



Tree Representation

Symbolic expression

Mutation: replace randomly chosen subtree by randomly generated tree

Probability pm to choose mutation

Probability to chose an internal point as the root of the subtree to be replaced

Recombination: exchange two randomly chosen subtrees among the parents

Population Management

Generation gap

= 1.0 means Generational EA

< 1.0 means Steady State EA

Fitness-Proportionate Selection (FPS)



Premature Convergence

Parent Selection: Scaling

Windowing: 

Sigma Scaling: 

Rank-based Selection

Linear Ranking



*s:* 1 < *s* ≤ 2 measures advantage of best individual

Exponential Ranking



c: normalise constant factor according to population size

Sampling algorithms

Roulette wheel alg.

Stochastic universal sampling alg.

All methods above rely on global population statistics

**Tournament Selection**

local fitness information

Pick *k* members at random then select the best of these

higher *k* increases selection pressure

Parent Selection: Uniform



Survivor Selection

Fitness-based survivor selection

Elitism: Always keep at least one copy of the fittest solution so far

Delete worst

Round-robin tournament:

* + P is the set of μ parents, O is the set of λ offspring
  + Pairwise competitions in round-robin format:
    - Each solution x from P ∪ O is compared with *q* other randomly chosen solutions
    - For each comparison, a "win" is assigned to x if it is better than its opponent
    - The μ solutions with the greatest number of wins are selected to be parents of the next generation
  + Parameter *q* allows tuning selection pressure
  + Typically *q* = 10

(μ,λ)-selection a.k.a. “comma strategy”, children only

(μ+λ)-selection a.k.a. “plus strategy”, parents and children

Selection pressure

*τ\**is a measure to quantify the selection pressure



Explicit Approaches for preserving diversity

Fitness sharing



Crowding

assumption that offspring will be close to parents

d(p1,o1) + d(p2,o2) < d(p1,o2) + d(p2,o1)

o1 compete with p1 and o2 compete with p2

Implicit Approaches for Preserving Diversity

Automatic Speciation

* Restrict mating to genotypically / phenotypically similar individuals
* Restrict mating to individuals that have the same (or very similar) tag, where

A tag is an extra bit (or bits) in the genotype is initialized randomly and is subject to recombination and mutation

Island Model Parallel EAs

Periodic migration of individuals between subpopulations

Cellular EAs

* Consider each individual to exist on a point on a grid (usually a rectangular toroid)
* Selection (hence recombination) and replacement happen using concept of a neighbourhood a.k.a. ***deme***
* Leads to different parts of grid searching different parts of space, good solutions diffuse across grid over a number of generations

GA

|  |  |
| --- | --- |
| Representation | Bit-strings |
| Recombination | 1-Point crossover |
| Mutation | Bit flip |
| Parent selection | Fitness proportional – implemented by Roulette Wheel |
| Survivor selection | Generational |

ES

|  |  |
| --- | --- |
| Representation | Real-valued vectors |
| Recombination | Discrete or intermediary |
| Mutation | Gaussian perturbation |
| Parent selection | Uniform random |
| Survivor selection | (μ,λ) or (μ+λ) |

EP

|  |  |
| --- | --- |
| Representation | Real-valued vectors |
| Recombination | None |
| Mutation | Gaussian perturbation |
| Parent selection | Deterministic (each parent one offspring) |
| Survivor selection | Probabilistic (μ+μ) |

GP

|  |  |
| --- | --- |
| Representation | Tree structures |
| Recombination | Exchange of subtrees |
| Mutation | Random change in trees |
| Parent selection | Fitness proportional |
| Survivor selection | Generational replacement |

Differential Evolution: Differential mutation





Population is a list. (ordering is not related to fitness value.)

1. Create a mutant vector population
2. Create trial vector population
3. Deterministic selection

|  |  |
| --- | --- |
| Representation | Real-valued vectors |
| Recombination | Uniform crossover |
| Mutation | Differential mutation |
| Parent selection | Given individual deterministically + Uniform random selection of the 3 necessary other vectors |
| Survivor selection | Deterministic elitist replacement (parent vs. child) |

PSO

Optimizing nonlinear functions

No crossover







Particle: 









|  |  |
| --- | --- |
| Representation | Real-valued vectors |
| Recombination | None |
| Mutation | Adding velocity vector |
| Parent selection | Deterministic (each parent creates one offspring via mutation) |
| Survivor selection | Generational (offspring replaces parents) |

Parameter-performance landscape or utility landscape for each { EA + problem instance + performance measure }

|  |  |  |
| --- | --- | --- |
|  | **LOWER PART** | **UPPER PART** |
| METHOD | EA | Tuner |
| SEARCH SPACE | Solution vectors | Parameter vectors |
| QUALITY | **Fitness** | **Utility** |
| ASSESSMENT | **Evaluation** | **Test** |

Average performance by solution quality, speed (MBF, AES)

Success rate = % runs ending with success

Robustness = variance in those averages over different problems

Parameter Control

Varing mutation step size







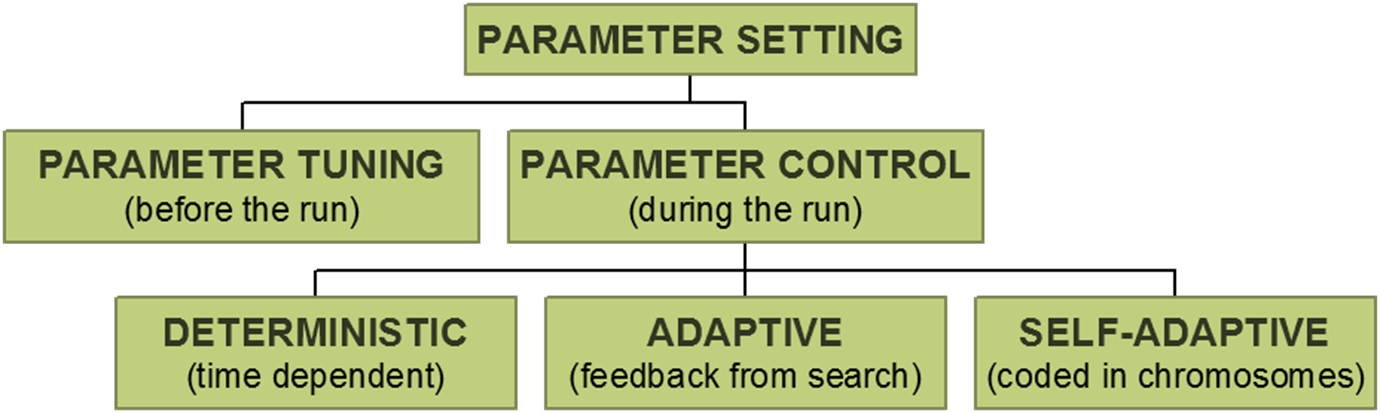


Varying penalties

*eval(x) = f(x) + W × penalty(x)*







T-test (and alike) indicate the chance that the values came from the same underlying distribution

Local search

*Neighbourhood N(x)* of point x is the set of points that can be reached from *x* with one application of a move operator

Pivot rule: condition for stopping neighbourhood search, e.g., greedy ascent

Greedy Ascent: stop as soon as a fitter neighbour is found

Steepest Ascent: stop after the whole set of neighbours examined and choose the best

Lamarckian

traits acquired by an individual during its lifetime can be transmitted to its offspring

EA implementation: replace individual with fitter neighbour (thus: use another genotype)

Baldwinian

* traits acquired by individual cannot be transmitted to its offspring
* EA implementation: individual receives the fitness of fitter neighbour (but keeps its own genotype)

Boltzmann selection operator in an EA



**Pareto-optimal set:** a set of non-dominated solutions in the solution space.

**Pareto-optimal front**: an image of the Pareto-optimal set in the objective space

Indirect constraint handling

a. penalty for violated constraints

b. penalty for wrongly instantiated variables

c. estimating distance/cost to feasible solution

1. eliminating infeasible candidates (very inefficient, hardly practicable)
2. repairing infeasible candidates (problem specific, may or may not be possible/easy)
3. preserving feasibility by special mutation / crossover operators  
   (requires feasible initial population)
4. decoding, i.e. transforming the search space  
   (allows usual representation and operators)

NEAT: NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

CPPNS: COMPOSITIONAL PATTERN PRODUCING NETWORKS

HYPERNEAT

Creating a neural network with a CPPN: making a neural network with another neural network.