Genetic Algorithms

Introduction

- Holland's GA
  - bitarray representation
  - fitness proportionate selection for mating
  - single point crossover, mutation
  - inversion operator introduced
- Currently there are many varieties with non-binary encodings, different selection strategies and operations.

Why it works?

- **Schema Theorem (Holland):** Short, low-order, above-average schemata (sequence of matching bits in a solution candidate) receive exponentially increasing trials in subsequent generations of a genetic algorithm.
- **Building Block Hypothesis (Goldberg):** A genetic algorithm seeks near-optimal performance through the juxtaposition of short, low-order, high performance schemata called the building blocks.

- If your problem is somewhat decomposable into small subproblems called building blocks, GA solves it better.
- Many design choices affect GA's capability of:
  - Introduction of new building blocks
  - Mixing existing building blocks from different individuals
  - Preserving existing building blocks.
- **Deceptive problems:** problems where combining high fit building blocks leads to a non-optimal solution.

Encoding

- **Genotype:** how it is represented, **Phenotype:** what it represents
- Example:
  Genotype: a binary sequence of size *n*
  Phenotype: an array of size *n/8.* Each byte represents an element of a vector of ASCII characters.
- Encoding should:
  - Favor building block growth
  - Preserve locality.
  - Be closed under genetic operators (crossover of two genotypes come up with a genotype representing a valid phenotype)

General GA Structure

I_λ: parent population, I_μ: offspring population,

p_t population at generation t

Θ_c: crossover probablity, Θ_m: mutation probability

$t \leftarrow 0$

$p_0 = randompopulation()$ ← Initialization

while $(\neg$ termination condition$)$ ← fixed # iterations or local optima

$I_\mu \leftarrow selecttomate(p_t, \Theta_c)$ ← sexual selection

$I_\kappa \leftarrow crossover(I_\mu)$ ← single/multi point, uniform

$I_\lambda \leftarrow mutate(I_\kappa, \Theta_m)$

$p_{t+1} \leftarrow selecttosurvive(I_\mu, I_\lambda)$ ← ecological selection,
elitism,
selection from λ only, or $(\mu+\lambda)$

$t \leftarrow t+1$

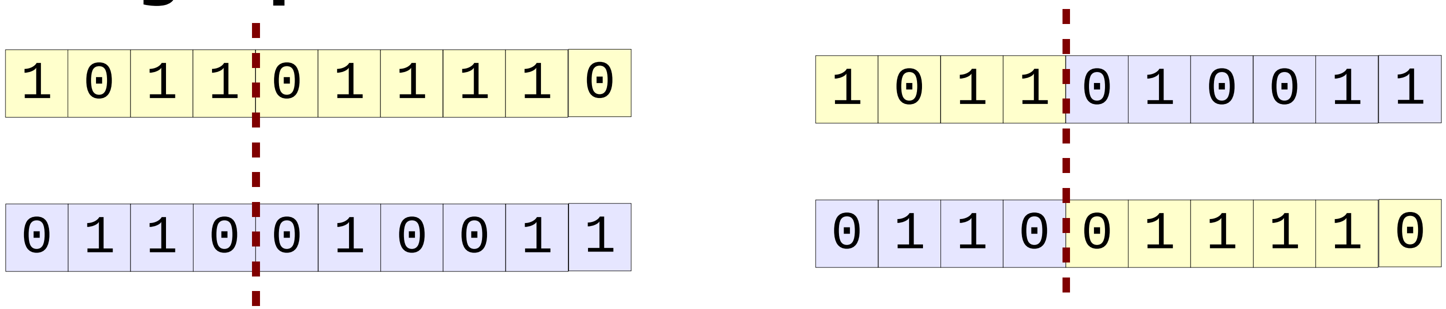
Initialization

- Create a random initial population
- Although mutation operation provide some genetic variety in the process, it cannot ensure that the parts of the solution exists in the population.
- Population size:
  - Sample quality of the search space
  - complexity of the GA
- Assure all building blocks of some degree are available in the initial population.

Crossover

- Combination of gene traits of two individuals to produce one or two offsprings. (Meiosis)
- single point, n-points (n>1), and uniform crossover.
- depending on encoding, different types of crossovers exists for encoding integrity. Like permutation preserving crossovers OX, PMX.

- **Single point crossover**

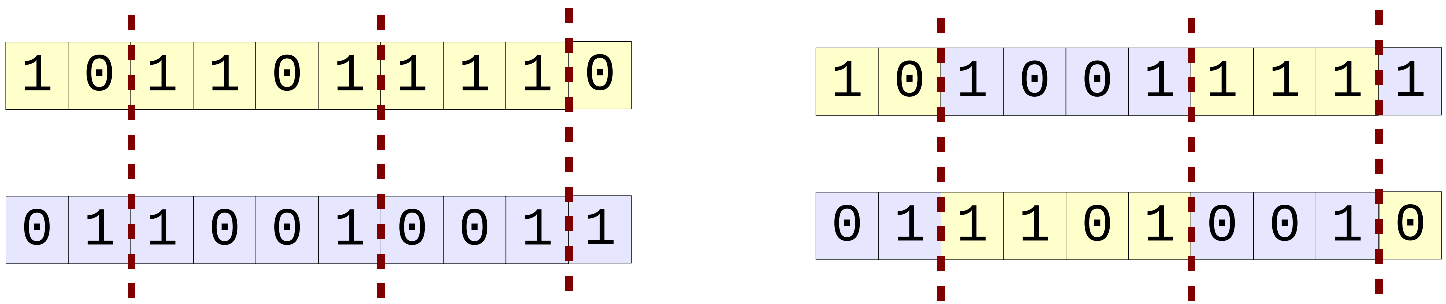


parents offsprings

$$k \leftarrow random(1..n)$$

$$\boldsymbol{o}_1[i] = \begin{cases} \boldsymbol{p}_1[i] & \text{if } i \le k \\ \boldsymbol{p}_2[i] & \text{otherwise} \end{cases}, \qquad \boldsymbol{o}_2[i] = \begin{cases} \boldsymbol{p}_2[i] & \text{if } i \le k \\ \boldsymbol{p}_1[i] & \text{otherwise} \end{cases}$$

- ***k*-points crossover**



parents

offsprings

$$\boldsymbol{s} \leftarrow randomset(1..n, k), \qquad sw = true$$

for $i = 1, 2, \ldots, n$

$\quad$ if $i \in \boldsymbol{s}$ then $sw \leftarrow \neg sw$

$$\boldsymbol{o}_1[i] = \begin{cases} \boldsymbol{p}_1[i] & \text{if } sw \\ \boldsymbol{p}_2[i] & \text{otherwise} \end{cases}, \qquad \boldsymbol{o}_2[i] = \begin{cases} \boldsymbol{p}_2[i] & \text{if } \neg sw \\ \boldsymbol{p}_1[i] & \text{otherwise} \end{cases}$$

- **uniform crossover**

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

parents

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

offsprings

for $i = 1, 2, \ldots, n$

$$sw \leftarrow random(\{true, false\})$$

$$\boldsymbol{o}_1[i] = \begin{cases} \boldsymbol{p}_1[i] & \text{if } sw \\ \boldsymbol{p}_2[i] & \text{otherwise} \end{cases}, \qquad \boldsymbol{o}_2[i] = \begin{cases} \boldsymbol{p}_2[i] & \text{if } \neg sw \\ \boldsymbol{p}_1[i] & \text{otherwise} \end{cases}$$

Mutation

- Due to errors occurred in meiosis, some codons copied with random values.
- Genetic algorithms, some gene positions are changed with a random probability. Some algoritms introduce their own controlled mutations like swaps, insertions, deletions.

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

$$\text{if } random(0..1) \le \Theta_m \text{ then } \boldsymbol{p}_i \leftarrow \neg \boldsymbol{p}_i$$

Selection

- Selection for mating vs. selection for survival.
- Mating strategies:
  - Truncation selection
  - Fitness proportionate selection (roulette wheel algorithm)
  - Rank selection
  - Tournament selection
- Survival based on either:
  - I_λ will be the new population
  - Selection based on $I_\mu + I_\lambda$,
    some best members of I_μ and members of I_λ,
    or simply selection among the mixed population

- **Fitness proportionate selection:**

  f_i: fitness of individual i, p_i: probability of individual i selected

  $$p_i = \frac{f_i}{\sum_{j=1}^{n} f_j}$$

- Roulette Wheel algorithm:
  - Vector based
  - Cumulative Distribution
- Selection pressure: ratio of best individual's selection probability to average selection probability.
- Selection pressure and probability distribution cannot be adjusted.

- Vector based roulette-wheel:

$\Delta p \leftarrow 0 ;\ j \leftarrow 1$

$\Delta M = \dfrac{1}{M}$

for $i = 1, 2, \ldots, N$

  $\Delta p \leftarrow \Delta p + p_i$

  while $\Delta p > \Delta M$

    $\boldsymbol{v}_j \leftarrow i$

    $\boldsymbol{\Delta} p \leftarrow \Delta p - \Delta M$

    $j \leftarrow j + 1$

$s \leftarrow random(1..M)$

select v_s

$f_1 = 6,\quad f_2 = 3,\quad f_3 = 12,\quad f_4 = 6,\quad f_5 = 3$

$p_1 = 0.2,\quad p_2 = 0.1,\quad p_3 = 0.4,\quad p_4 = 0.2,\quad p_5 = 0.1$

1	1	2	3	3	3	3	4	4	5

- Cumulative Distribution roulette-wheel:

$c \leftarrow 0\,;\, i \leftarrow 1$

for $i = 1, 2, \ldots, N$

$\quad c \leftarrow c + p_i$

$\quad v_i \leftarrow c$

$s \leftarrow random(0,1)$

search $j \ni \; v_{j-1} \leq s < v_j$

select j

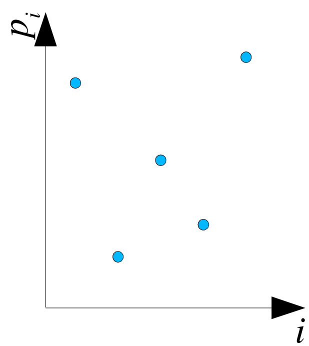
$f_1 = 6, \quad f_2 = 3, \quad f_3 = 12, \quad f_4 = 6, \quad f_5 = 3$

$p_1 = 0.2, \quad p_2 = 0.1, \quad p_3 = 0.4, \quad p_4 = 0.2, \quad p_5 = 0.1$

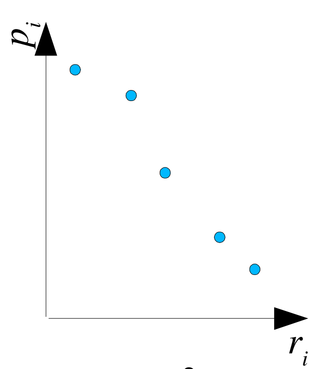
0.2	0.3	0.7	0.9	1.0

- **Rank selection:**
  individuals are sorted according to their fitness. selection probability of an individual is a function of their rank.

- The probability distribution is completely adjustable:



$f_1=7 \quad p_1=0.304$
$f_2=1 \quad p_2=0.043$
$f_3=4 \quad p_3=0.173$
$f_4=2 \quad p_4=0.086$
$f_5=8 \quad p_5=0.347$



$r_1=2$
$r_2=5$
$r_3=3$
$r_4=4$
$r_5=1$

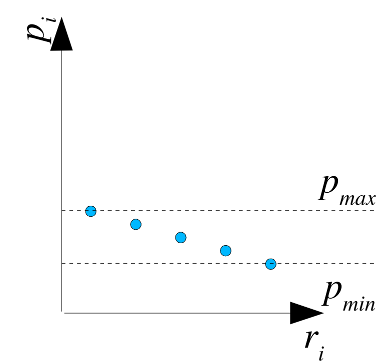
$$p_i = p_{min} + \frac{(p_{max} - p_{min})\ r_i}{N}$$

$$\sum_{i=1}^{N} p_i = 1$$

$$f_i = 2 - s + \frac{2(s-1)(i-1)}{N-1}$$

$s \in [1,2]$: fitness pressure

$$p_{min} = \frac{2-s}{N}, \quad p_{max} = \frac{s}{N}$$



- Also non-linear distributions are possible

- **Tournament selection:**
  Pick two individuals at random and choose the better one to mate.
- Simple, fitness pressure can be controlled by the size of the tournament (best of 3,4,5 etc.)