Mutation for bits string genotypes

Most classical option: probabilistic bit flip mutation

1. copy parent genotype g_p as child genotype g_c
2. for each bit in the in g_c, flip it ($0 \rightarrow 1$ or $1 \rightarrow 0$) with p probability

Commonly, $\underline{p = 0.01}$

$$g_p = 001010011101010101100100101$$
$$g_c = 001010111101010101101100101$$

Crossover for (bits) string genotypes

$$\begin{tikzcd} g_1 \arrow[dr, dash] & \\ & g_c \\ g_2 \arrow[ur, dash] & \end{tikzcd}$$

Many options:

- one-point crossover
- two-points crossover
- *n*-points crossover
- uniform crossover
- ...

One-, two-, *n*-points crossover

Assume parents with equal genotype size:

1. choose randomly one (two, n) *cut points* in the genotype (indexes i such that $i < |g_{p_1}| = |g_{p_2}|$)
2. child bits before the cut point comes from parent 1, child bits after the cut point comes from parent 2

In general, jth bit comes from parent 1 iff closest larger cut point is even, from 2, otherwise.

One-, two-, *n*-point crossover

One-point:

$$
\begin{aligned}
g_{p_1} &= 0010100111010101 0|1100100101 \\
g_{p_2} &= 11101010101001010|0101110111 \\
g_c &= 00101001110101010\ 0101110111
\end{aligned}
$$

Two-points:

$$
\begin{aligned}
g_{p_1} &= 0010100|1110101010|1100100101 \\
g_{p_2} &= 1110101|0101001010|0101110111 \\
g_c &= 0010100\ 0101001010\ 1100100101
\end{aligned}
$$

Uniform crossover

1 0 0 1 0 0 1 1 0 1

0 0 0 0 0 1 0 1 1 1

1 0 0 1 0 1 1 1 0 1

A cut point is placed at each index with $p = 0.5$ probability

Crossover with variable length (bits) string genotype

Many variants:

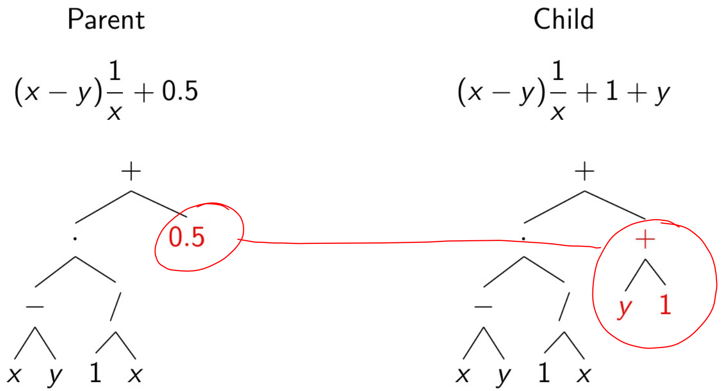
- one-, two-points crossover
  - cut points may be different within parents
  - child genotype size may be larger or smaller than parents sizes
- ...

One-point:

$$\begin{aligned} g_{p_1} &= 00101001110101010|1100100101 \\ g_{p_2} &= 111010101|01001010010111 0111 \\ g_c &= 00101001110101010\ 01001010010111 0111 \end{aligned}$$

Genotype-phenotype mapping must allow for variable length genotypes!

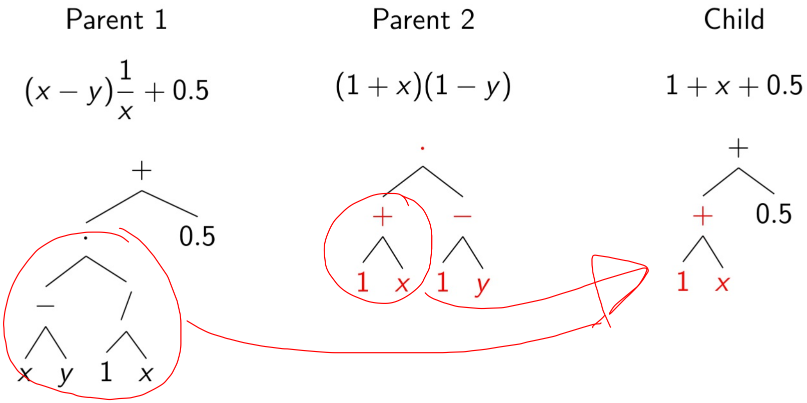
Mutation (trees)



1. choose a random subtree
2. replace with a randomly generated subtree

Usually, constraints on depth

Crossover (trees)



1. choose a random subtree in parent 1
2. choose a random subtree in parent 2
3. swap subtrees (child is copy of parent)

Usually, constraints on depth

Role of operators

Mutation (x)or crossover?

- mutation $\rightarrow$ exploitation
- crossover $\rightarrow$ exploration

Population initialization

- Totally random
- More specific approaches, dependent on genotype form

Fitness

Fitness of an individual = ability to solve the problem of interest

- errors on several fitness cases by execution/simulation/application

Common cases:

- one numerical index
- more than one numerical indexes
- ...

Closely related with selectors

Many indexes: $\underbrace{\text{multi}}_{\text{many}}$objective

$$f(i) = \langle f_1(i), \dots, f_n(i) \rangle$$

How to compare individuals i_1, i_2 ?

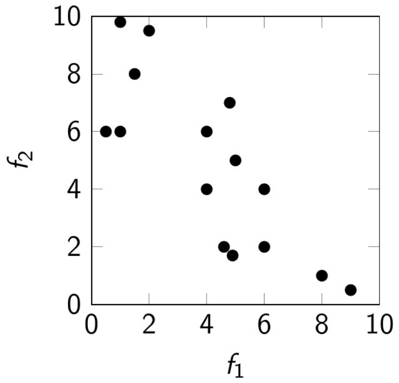
- linearization
  - $f(i) = \alpha_1 f_1(i) + \cdots + \alpha_n f_n(i)$
- lexicographical order
  - compare $f_1(i_1) \overset{?}{>} f_1(i_2)$; if tie, $f_2(i_1) \overset{?}{>} f_2(i_2)$; ...
- Pareto dominance
- ...

Q: with which selectors?

Pareto dominance

i_1 dominates i_2 iff:

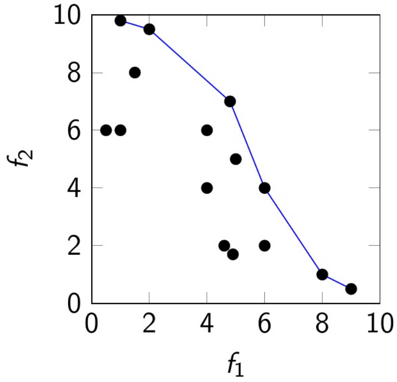
$$\forall j, f_j(i_1) \geq f_j(i_2) \wedge \exists k, f_k(i_1) > f_k(i_2)$$



Pareto dominance

i_1 dominates i_2 iff:

$$\forall j, f_j(i_1) \geq f_j(i_2) \wedge \exists k, f_k(i_1) > f_k(i_2)$$

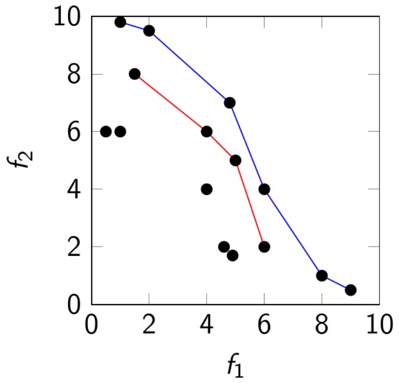


- 1st Pareto front: undominated solutions

Pareto dominance

i_1 dominates i_2 iff:

$$\forall j, f_j(i_1) \geq f_j(i_2) \wedge \exists k, f_k(i_1) > f_k(i_2)$$

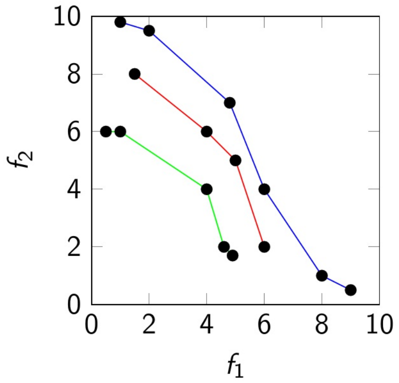


- 1st Pareto front: undominated solutions
- 2nd Pareto front: undominated solutions, while not considering 1st front

Pareto dominance

i_1 dominates i_2 iff:

$$\forall j, f_j(i_1) \geq f_j(i_2) \wedge \exists k, f_k(i_1) > f_k(i_2)$$



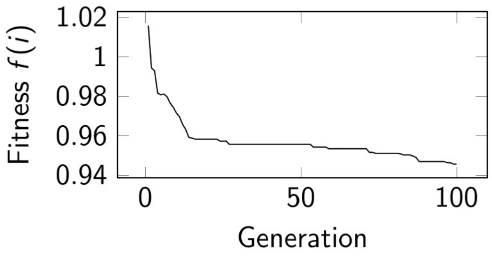
- 1st Pareto front: undominated solutions
- 2nd Pareto front: undominated solutions, while not considering 1st front
- ...

In practice

- Is my EA working?
- When to stop evolution?
- How to choose value for parameter X?

In practice

- Is my EA working?
- When to stop evolution?
- How to choose value for parameter X?



On many (≥ 30) runs!

Issues

- Diversity
- Variational inheritance
- Expressiveness . . .

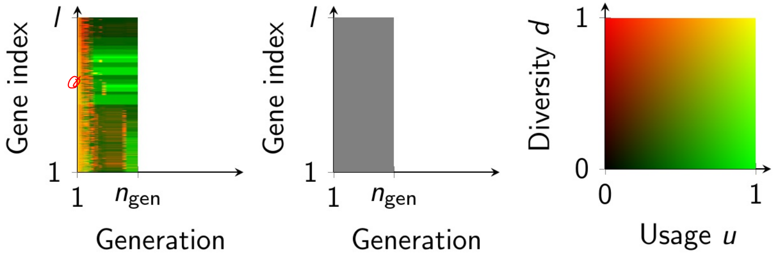
Diversity

Is the population diverse enough?

- "No" $\rightarrow$ too much exploitation $\rightarrow$ local minimum
- "Yes" $\rightarrow$ in principle, no drawbacks
    - how to measure diversity?
    - how to enforce/promote diversity?

Giovanni Squillero and Alberto Tonda. "Divergence of character and premature convergence: a survey of methodologies for promoting diversity in evolutionary optimization". In: *Information Sciences* 329 (2016), pp. 782–799

Diversity: visualization



Eric Medvet and Tea Tušar. "The DU Map: A Visualization to Gain Insights into Genotype-phenotype Mapping and Diversity". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. GECCO '17. Berlin, Germany: ACM, 2017, pp. 1705–1712

Variational inheritance

Are children similar but not identical to parents?

- "Too much similar" $\rightarrow$ too much exploitation $\rightarrow$ local minimum, no/slow evolution
- "Too much different" $\rightarrow$ no exploitation, just coarse exploration (random walk)

- How to measure? (locality, redundancy, degeneracy, uniformity, . . .)
- How to tackle? Operators, mapping, both?

Expressiveness

Is the representation (phenotype) expressive enough?

- "Low expressiveness" $\rightarrow$ good/optimal solution might not be representable, or might not be reachable
- "Large expressiveness" $\rightarrow$ large search space $\rightarrow$ very long or infiniti convergence time

Fitness landscape

- How are genotype and fitness spaces related?
- What does a small step on one correspond to on the other?

Q: is phenotype space relevant?

Genetic Algorithms (GA)

- Genotype = phenotype = bits string
- $m = n \approx 1000$, no overlapping
- Fitness-proportional selection, or multiobjective (Pareto-based) selection

- Most widely used/studied
- Genotypes often encodes numerical parameters

Genetic Programming (GP)

POPULAR FOR
SYMBOLIC REGRESSION (FIND FORMULA)

Focus: individuals are programs

- Genotype = phenotype = tree (tree-based GP) or list of instructions (linear GP)
- $m = n \approx 1000$, overlapping
- Tournament selection

- Syntactic/semantic validity?

Grammatical Evolution (GE)

A form of GP based on GA, given a context-free grammar $\mathcal{G}$

- Genotype $=$ bits string, phenotype $=$ string $\in \mathcal{L}(\mathcal{G})$, by means of a mapping procedure
- steady state ($m \approx 500, n = 1$, overlapping) or $m = n$, overlapping
- Tournament selection

GE (standard) genotype-phenotype mapping

✓

$$g = 01101001\ 00001101\ 01011000\ 00000011\ 11000110\ 01111101 \qquad \text{(bits)}$$
$$= 105\ 13\ 88\ 3\ 198\ 125 \qquad \text{(integers)}$$

104

i	g_i	$\lvert r_s \rvert$	j	w	Phenotype p
					`<expr>`
0	105	$\textcircled{3}$	0	0	`( <expr> <op> <expr> )`
1	13	3	$\textcircled{1}$	0	`( <var> <op> <expr> )`
2	88	2	0	0	`( x <op> <expr> )`
3	3	4	3	0	`( x / <expr> )`
4	198	3	0	0	`( x / ( <expr> <op> <expr> ) )`
5	125	3	2	0	`( x / ( <num> <op> <expr> ) )`
0	105	10	5	1	`( x / ( 5 <op> <expr> ) )`
1	13	4	1	1	`( x / ( 5 - <expr> ) )`
2	88	3	1	1	`( x / ( 5 - <var> ) )`
3	3	2	1	1	`( x / ( 5 - y ) )`

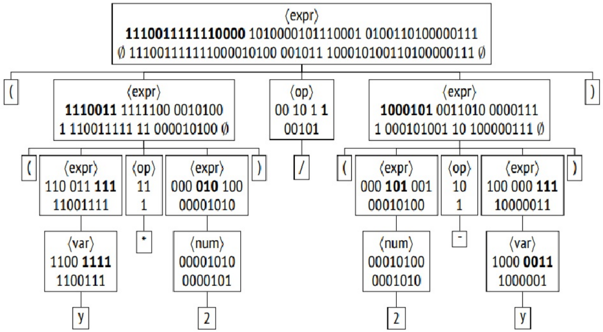
$$G=(T,N,n_0,R)$$

```
<expr> ::= ( <expr> <op> <expr> ) | <var> | <num>
<op> ::= + | - | * | /
<var> ::= x | y
<num> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

An alternative: WHGE genotype-phenotype mapping

$g = 111001111111000010100001011100010100110100000111$

$p = ((y * 2)/(2 - y))$



Weighted

Eric Medvet. "Hierarchical Grammatical Evolution". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. GECCO '17. Berlin, Germany: ACM, 2017, pp. 249–250