

Evolvability in Grammatical Evolution

Eric Medvet¹, Fabio Daolio², Danny Tagliapietra¹

¹: DIA, University of Trieste, Italy

²: University of Stirling, Scotland



MACHINE
LEARNING
LAB

UNIVERSITY of
STIRLING



GECCO, 17/7/2017, Berlin (Germany)

<http://machinelearning.inginf.units.it>

Table of Contents

- 1 Motivation
- 2 Context: GE and evolvability
- 3 Experimental analysis

Grammar-based GP

- grammar-based GP approaches are popular
 - user just needs to provide the grammar and the fitness function
 - ensure valid individuals

Grammar-based GP

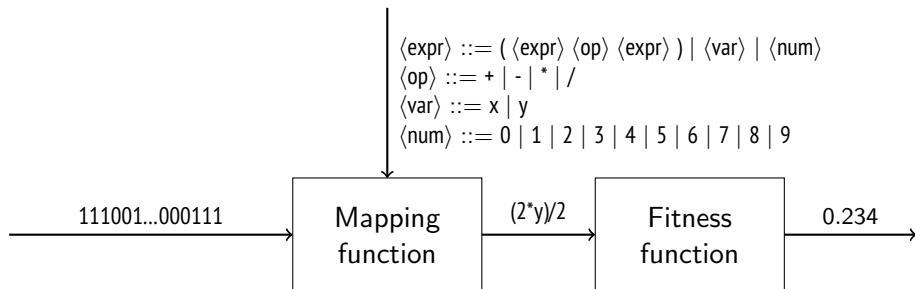
- grammar-based GP approaches are popular
 - user just needs to provide the grammar and the fitness function
 - ensure valid individuals
- among them, Grammatical Evolution (GE) widely used

The screenshot shows a Semantic Scholar search interface. At the top, the Semantic Scholar logo is on the left, and a search bar contains the text 'grammatical evolution'. Below the search bar, the results are displayed. On the left side, there are filter options: 'Filter Results:' with a checkbox for 'Full text PDF available (2.807)', 'Publication Year' with a bar chart showing an increasing trend from 1964 to 2017, and other filters for 'Publication Type', 'Author', and 'Publication Venue'. The main content area shows 'About 3,220 results'. The top result is titled 'Grammatical evolution' by Conor Ryan and GECCO, published in 2001. The abstract states: 'Grammatical Evolution is an automatic programming system that is a form of Genetic Programming structures. These structures can be in any form that can be specified using a grammar, in computer networks. When evolving computer languages, multiple types can be handled in a complete manner.' Below this, there are links for 'View PDF' and 'Related Publications'. The second result is 'Grammatical Evolution Evolutionary Automatic Programming Edition' by Maria Adler, published in 2016. The abstract for this result is partially visible: 'Number of pages: 197 pages Thank you for reading grammatical evolution evolutionary automatic programming in an arbitrary language 1st edition. Maybe you have knowledge that, people have look numerous times l evolution evolutionary automatic programming in an arbitrary language 1st edition, but en'.

GE

Three representations for the individual:

- genotype: a bit string
- phenotype: a string of the language defined by the CFG
- fitness



Genotype-phenotype mapping

Salient trait:

- allows using (standard) bit string genetic operators
- indirect mapping, poor variational inheritance
 - many variants proposed (BGE, π GE, SGE, ...)
 - large/hot debate: “GE mapping is bad!”¹, “No! It’s good!”²³

¹Whigham, Dick, and Maclaurin, “On the mapping of genotype to phenotype in evolutionary algorithms” .

²Squillero and Tonda, “(Over-)Realism in evolutionary computation: Commentary on “On the mapping of genotype to phenotype in evolutionary algorithms” by Peter A. Whigham, Grant Dick, and James Maclaurin” .

³Ryan, “A rebuttal to Whigham, Dick, and Maclaurin by one of the inventors of grammatical evolution: Commentary on “On the mapping of genotype to phenotype in evolutionary algorithms” by Peter A. Whigham, Grant Dick, and James Maclaurin” .

Mapping and properties

Mapping:

- invalidity: how often the mapping “fails”?
- redundancy⁴: is part of the genotype not used?
- degeneracy⁴: how often different genotypes are mapped to the same phenotype?

⁴Often, redundancy is used for degeneracy

Mapping and properties

Mapping:

- invalidity: how often the mapping “fails”?
- redundancy⁴: is part of the genotype not used?
- degeneracy⁴: how often different genotypes are mapped to the same phenotype?

Mapping + genetic operators:

- locality: are genotypic neighbors also phenotypic neighbors?
- neutrality: how often the child and parent with different genotypes share the same phenotype?

⁴Often, redundancy is used for degeneracy

Mapping and properties

Mapping:

- invalidity: how often the mapping “fails”?
- redundancy⁴: is part of the genotype not used?
- degeneracy⁴: how often different genotypes are mapped to the same phenotype?

Mapping + genetic operators:

- locality: are genotypic neighbors also phenotypic neighbors?
- neutrality: how often the child and parent with different genotypes share the same phenotype?

Mapping + genetic operators + fitness:

- evolvability: is the child fitter than its parents?

⁴Often, redundancy is used for degeneracy

Mapping and properties

Mapping:

- invalidity: how often the mapping “fails”?
- redundancy⁴: is part of the genotype not used?
- **degeneracy**⁴: how often different genotypes are mapped to the same phenotype?

Mapping + genetic operators:

- **locality**: are genotypic neighbors also phenotypic neighbors?
- neutrality: how often the child and parent with different genotypes share the same phenotype?

Mapping + genetic operators + fitness:

- **evolvability**: is the child fitter than its parents?

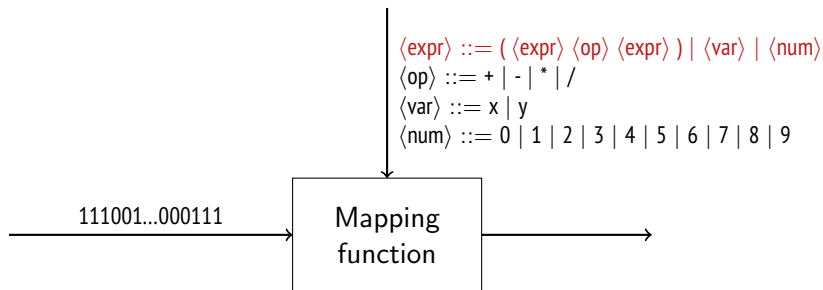
Many studies on locality/degeneracy, **none** on evolvability \Rightarrow our paper!

⁴Often, redundancy is used for degeneracy

Table of Contents

- 1 Motivation
- 2 Context: GE and evolvability
- 3 Experimental analysis

GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

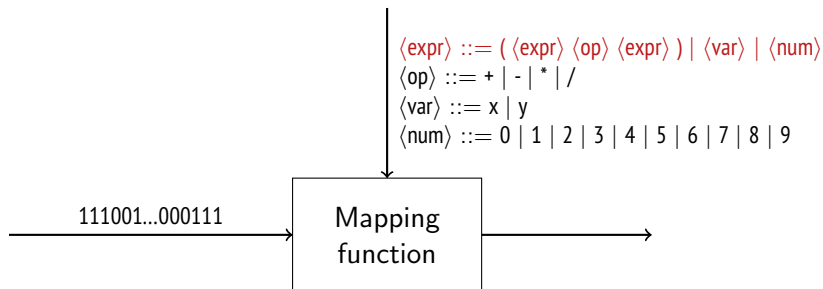
i=0

w=0

p=⟨**expr**⟩

$$231 \bmod 3 = 0$$

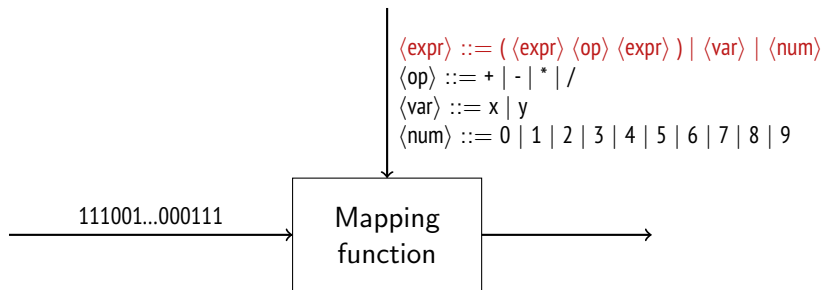
GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=1$
 $w=0$
 $p = (\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle)$
 $15 \bmod 3 = 0$

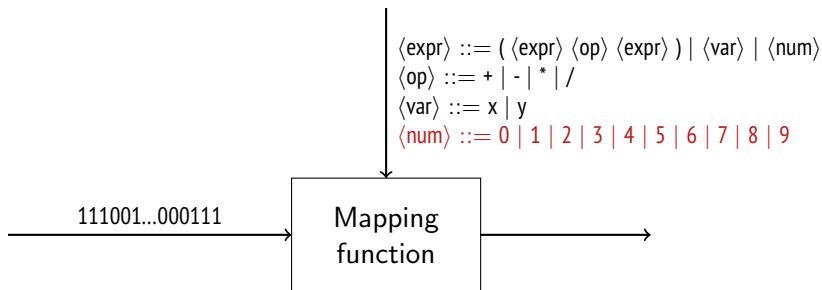
GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=2$
 $w=0$
 $133 \bmod 3 = 1$
 $p = ((\langle \mathbf{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle) \langle \text{op} \rangle \langle \text{expr} \rangle)$

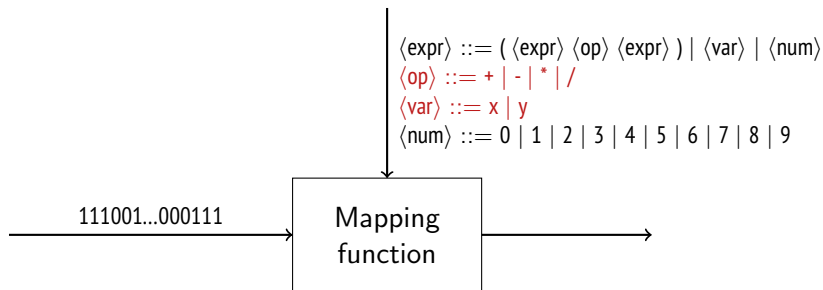
GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=3$
 $w=0$
 $142 \bmod 10 = 2$
 $p= ((\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle) \langle \text{op} \rangle \langle \text{expr} \rangle)$

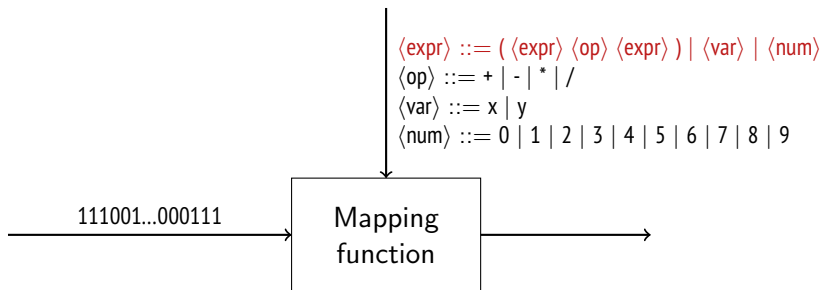
GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=4$
 $w=0$
 $178 \bmod 4 = 2$
 $p = ((2 \langle \text{op} \rangle \langle \text{expr} \rangle) \langle \text{op} \rangle \langle \text{expr} \rangle)$

GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

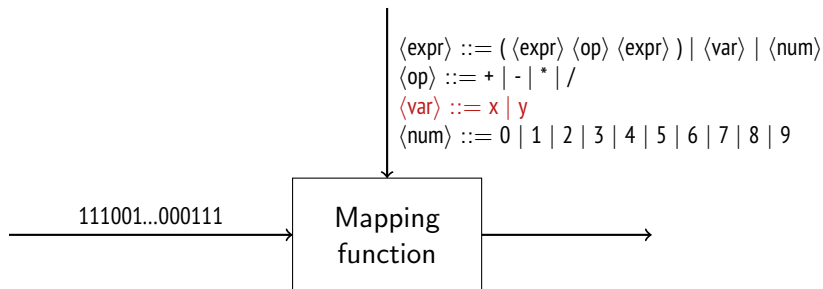
i=5

w=0

p= $((2 * \langle \text{expr} \rangle) \langle \text{op} \rangle \langle \text{expr} \rangle)$

224 mod 3 = 2

GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

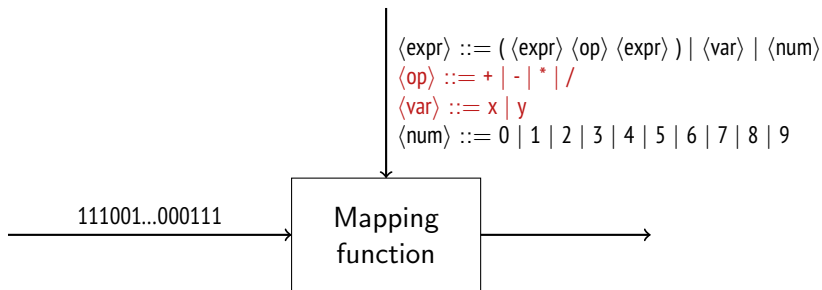
i=0

w=1

231 mod 2 = 1

p= ((2 * **var**) <op> <expr>)

GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

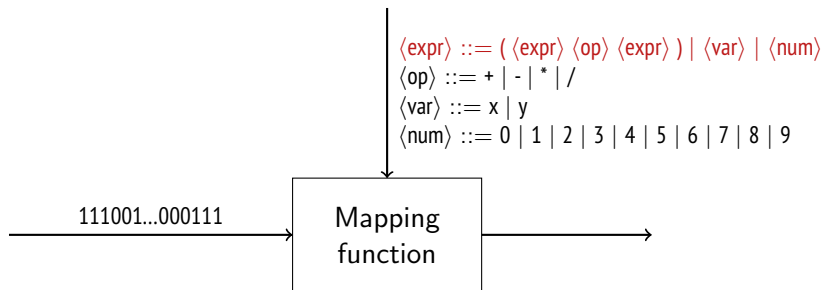
i=1

w=1

p= ((2 * y) <op> <expr>)

$$15 \bmod 4 = 3$$

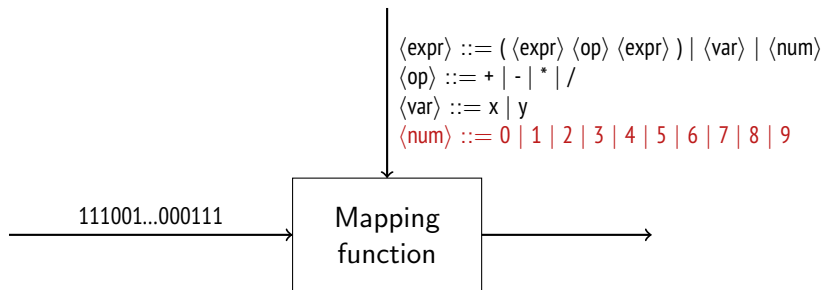
GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=2$
 $w=1$
 $p= ((2 * y) / \langle \text{expr} \rangle)$
 $133 \bmod 3 = 1$

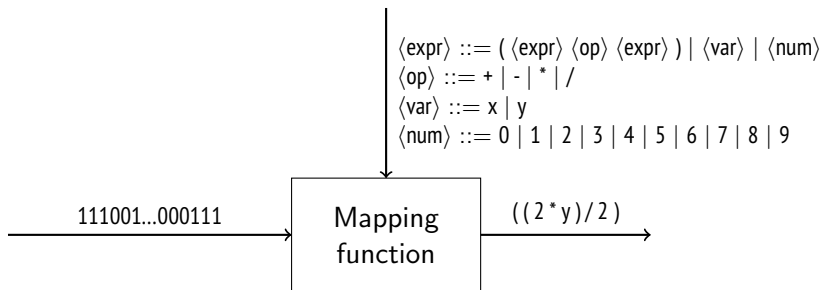
GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=3$
 $w=1$
 $p= ((2 * y) / \langle \text{num} \rangle)$
 $142 \bmod 10 = 2$

GE mapping in brief



Genotype (bits)	11100111	11110000	10100001	01110001	01001101	00000111
Genotype (ints)	231	15	133	142	178	224

 $i=3$
 $w=1$
 $p= ((2 * y) / 2)$

GE variants

- BGE: breadth first instead of leftmost
- π GE: non-terminal to expand chosen using the genotype
- SGE: structured genotype (int string), no genotype reuse, no modulo rule, custom operators
 - designed for better locality and lower degeneracy

Evolvability

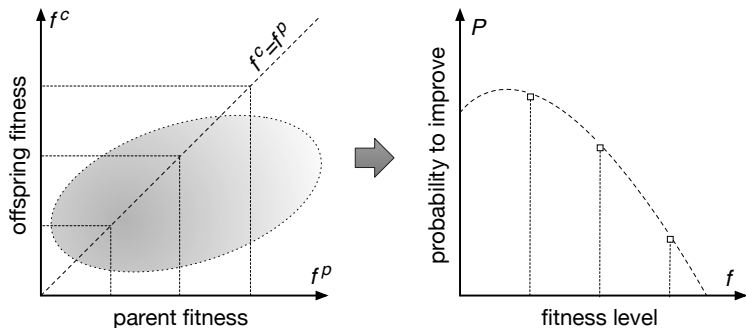
“is the child fitter than its parents?”

- a tool for *fitness landscape analysis*
 - problem hardness from point of view of search heuristic
- chosen metric: fitness cloud⁵ → **fitness-probability cloud (FPC)**⁶ → escape probability → accumulated escape probability (AEP)

⁵Verel, Collard, and Clergue, “Where are bottlenecks in nk fitness landscapes?”

⁶Lu, Li, and Yao, “Fitness-probability cloud and a measure of problem hardness for evolutionary algorithms”.

Fitness-probability cloud



- AEP: average of escape probability across bins

Table of Contents

- 1 Motivation
- 2 Context: GE and evolvability
- 3 Experimental analysis**

Procedure

Problems and variants:

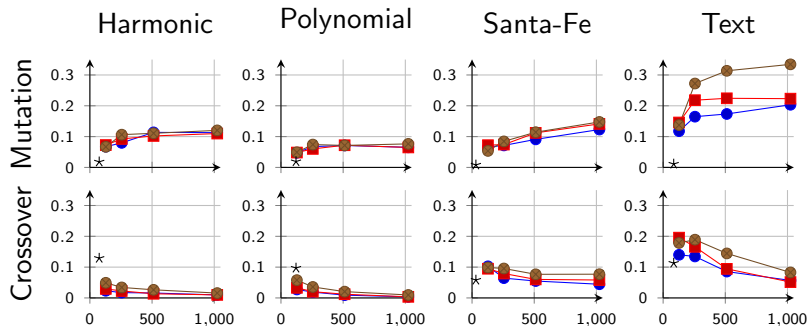
- 4 benchmark problems: harmonic, polynomial, Santa-Fe, text
- GE, BGE, π GE, SGE
 - 4 genotype sizes: 128, 256, 512, 1024 bit (except SGE)

Procedure:

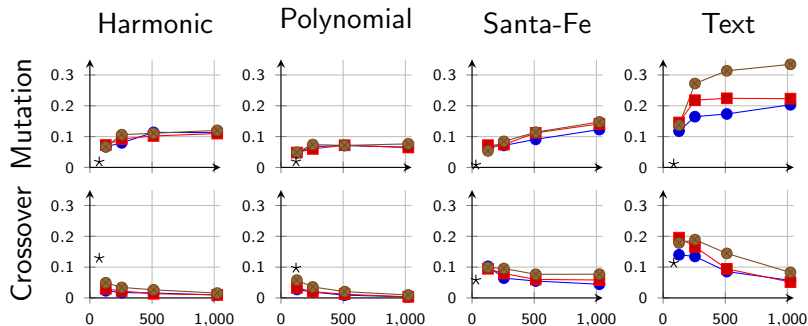
- 1 randomly generate 300 (pair of) parents
- 2 for each, apply 30 times the genetic operator
- 3 for each \langle parents, child \rangle , map, compute fitness, compute FPC and AEP
 - one FPC and one AEP value comes from 300×30 “points”

No dynamical aspects (selection, replacement)!

Results: AEP vs. genotype size

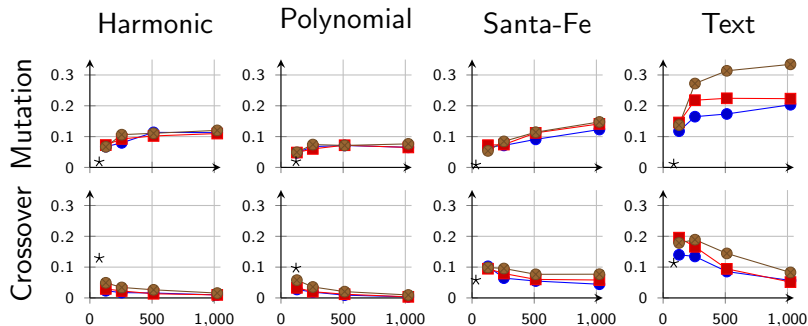


Results: AEP vs. genotype size



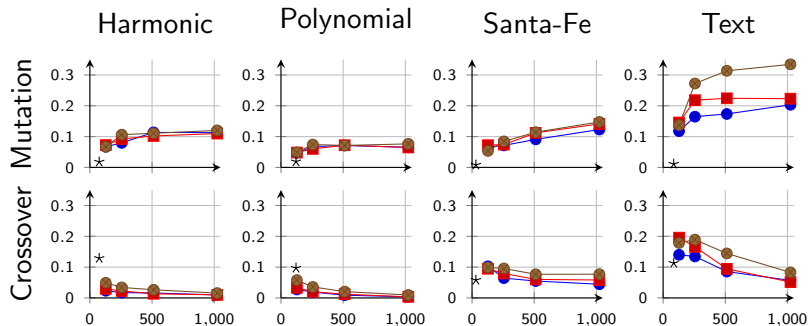
- SGE different from GE, BGE, π GE

Results: AEP vs. genotype size



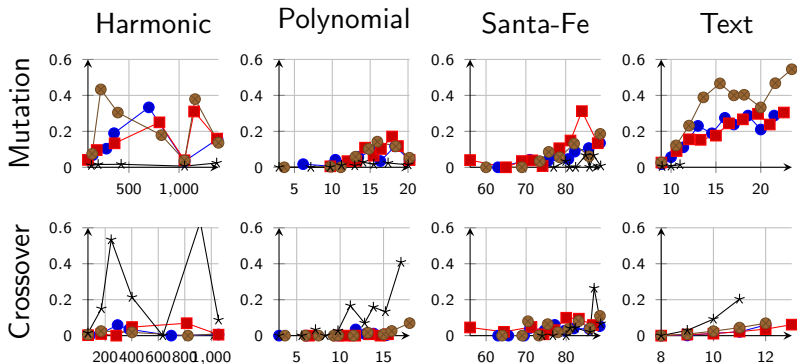
- SGE different from GE, BGE, π GE
- impact of size opposite for mutation and crossover

Results: AEP vs. genotype size

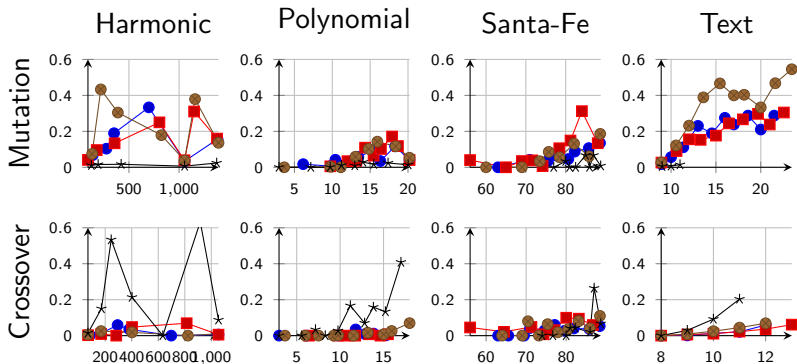


- SGE different from GE, BGE, π GE
- impact of size opposite for mutation and crossover
- differences among problems

Fitness-probability cloud

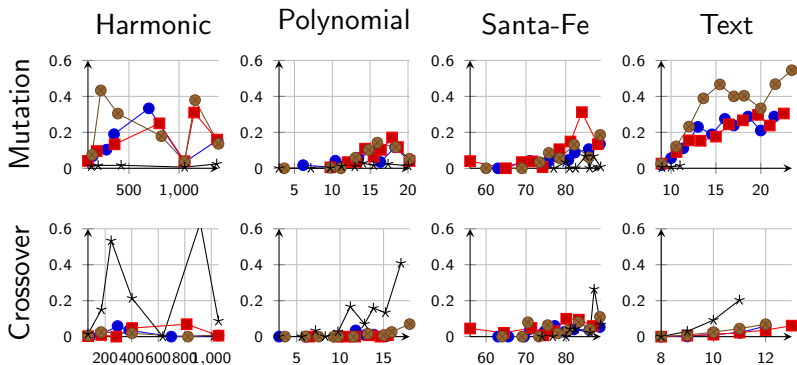


Fitness-probability cloud



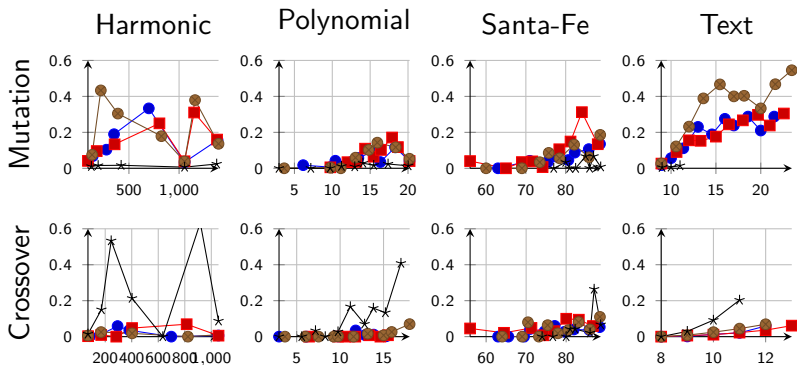
- the better the fitness, the lower the escape probability

Fitness-probability cloud



- the better the fitness, the lower the escape probability
- SGE different from GE, BGE, π GE

Fitness-probability cloud



- the better the fitness, the lower the escape probability
- SGE different from GE, BGE, π GE
- differences among problems

Explaining evolvability

Can mapping + genetic operators explain evolvability?

- locality

Pearson correlation between the shortest parent-child genotype distance $\min(d_g(g_p^1, g_c), d_g(g_p^2, g_c))$ and the shortest parent-child phenotype distance $\min(d_p(p_p^1, p_c), d_p(p_p^2, p_c))$

- neutrality

Explaining evolvability

Can mapping + genetic operators explain evolvability?

- locality

Pearson correlation between the shortest parent-child genotype distance $\min(d_g(g_p^1, g_c), d_g(g_p^2, g_c))$ and the shortest parent-child phenotype distance $\min(d_p(p_p^1, p_c), d_p(p_p^2, p_c))$

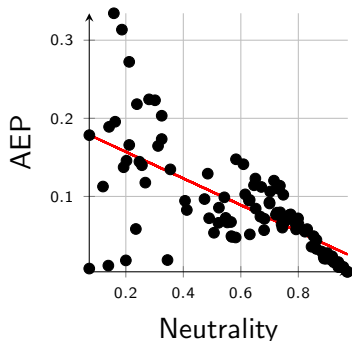
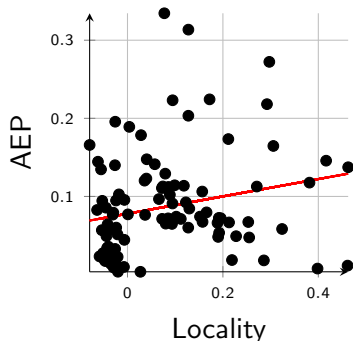
- neutrality

percentage of cases in which the child genotype is different from both parent genotypes (that is, $g_p^1 \neq g_c \neq g_p^2$) and the child phenotype is equal to at least one parent phenotype (i.e., $g_c = g_p^1 \vee g_c = g_p^2$)

Locality/neutrality and evolvability

One point:

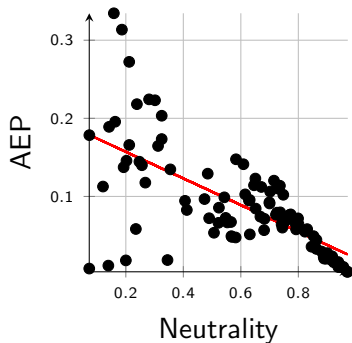
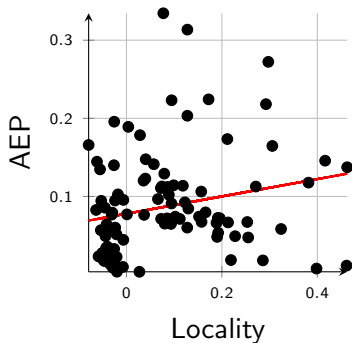
- a combination of: variant, problem, genotype size, operator
- coordinates come from 300×30 numbers



Locality/neutrality and evolvability

One point:

- a combination of: variant, problem, genotype size, operator
- coordinates come from 300×30 numbers

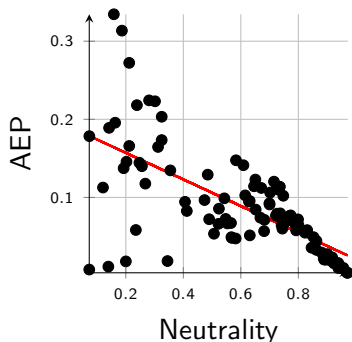
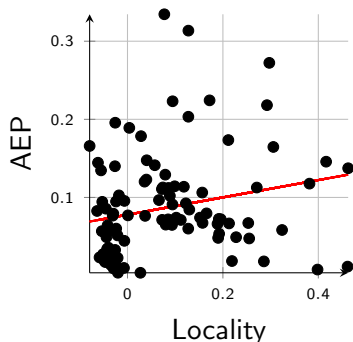


- neutrality more (neg.) correlated than locality

Locality/neutrality and evolvability

One point:

- a combination of: variant, problem, genotype size, operator
- coordinates come from 300×30 numbers



- neutrality more (neg.) correlated than locality
- intuition: no different child, no fitter child

Conclusions

First study of evolvability in GE:

- none among problem, mapping, genotype size, genetic operator strongly affects evolvability
- locality weakly correlated with evolvability
- neutrality well correlated with evolvability, in particular for higher values

Thanks!