# Fitness Landscapes and Problem Hardness in Evolutionary Computation


**Leonardo Vanneschi**
Dipartimento di Informatica, Sistemistica
e Comunicazione (DISCo)
University of Milano-Bicocca, Italy
vanneschi@disco.unimib.it
http://personal.disco.unimib.it/Vanneschi



**Sébastien Verel**
Laboratoire I3S
University of Nice-Sophia Antipolis /CNRS, France
verel@i3s.unice.fr
http://www.i3s.unice.fr/~verel

# Dishes of the day

- **Part I - Fitness Landscapes (Sébastien Verel)**

  - Definition of Fitness Landscape
  - Types of Fitness Landscapes
    - Multimodal
    - Rugged
    - Neutral
  - Measures to quantify multimodality and ruggedness
  - Measures to study neutrality

- **Part II - Problem Difficulty in GP (Leonardo Vanneschi)**

  - Binding between Fitness Landscapes and Problem Difficulty
  - Measures of difficulty applied to GP

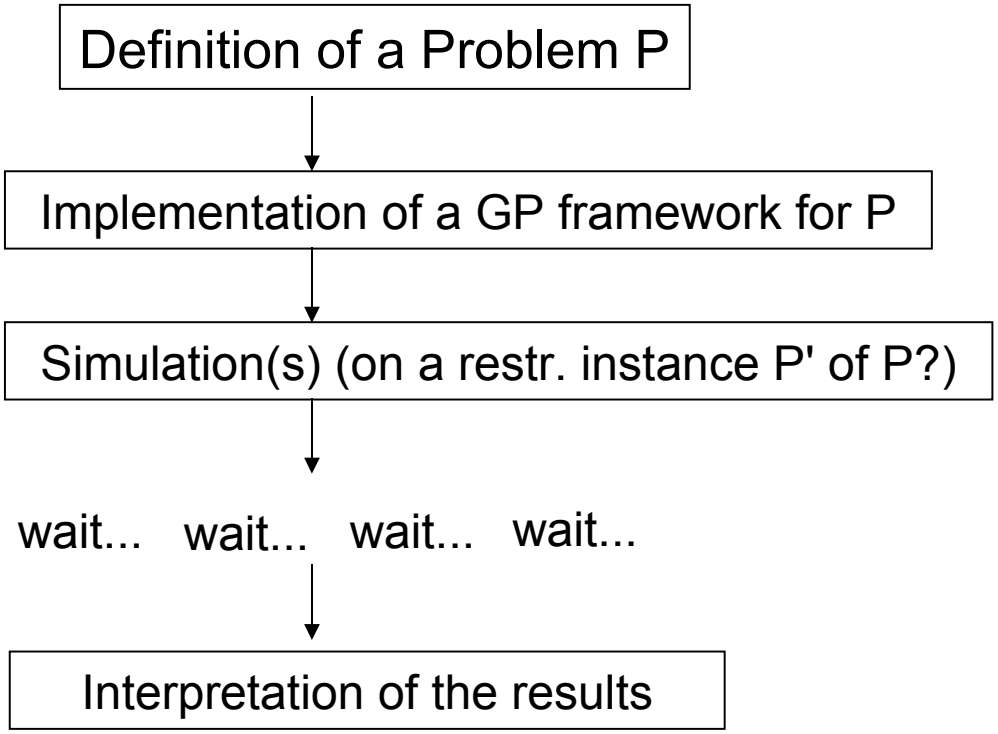
# PART II

# Problem Hardness in Genetic Programming

**Leonardo Vanneschi**
Dipartimento di Informatica, Sistemistica
e Comunicazione (DISCo)
University of Milano-Bicocca, Italy
vanneschi@disco.unimib.it
http://personal.disco.unimib.it/Vanneschi

Why predicting the difficulty of a problem is important?

# Is GP the good technique to solve my problem?




***Is there a better way?***

*Define some measures to quantify the ability of GP to solve a problem from its high level specification!*

Not obvious:

- GP is stochastic
- GP *works well* on P'... but how does it work on P?

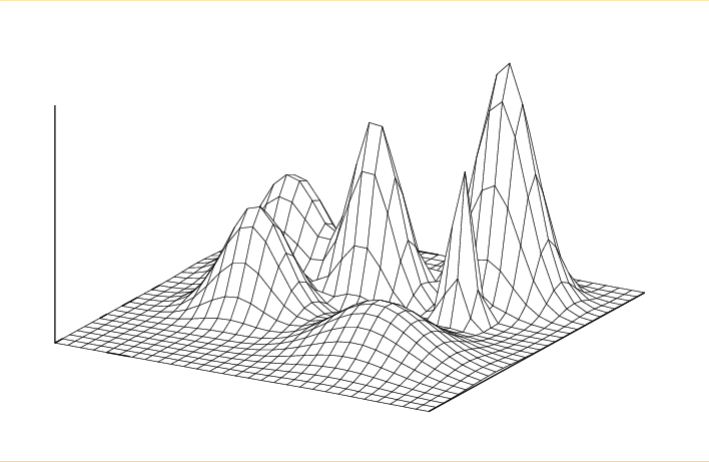
# The first step: J. R. Koza, 1992

Proposed "measure" of problem hardness: number of individuals that have to be sampled by GP before finding a solution with a given probability $p$ (usually $p = 0.99$).

## Remarks

- It can't be calculated without executing (many times!) GP
- It can be used to ***confirm*** the results of another hardness measure

# Fitness Landscapes in GP



- Huge search spaces!
- Multidimensionality of neighborhoods!

Very complex neighborhood structures (genotypes = trees, strings of dynamic size, graphs, ...)

Impossible to draw a Fitness Landscape also for simple problems!

We look for measures able to catch some interesting properties of Fitness Landscapes

# Autocorrelation. Kinnear, 1994

Proposed measure of problem hardness for GP: ***autocorrelation function*** (Weinberg in 1990 and Manderick in 1991 had studied the same measure for GAs).

Basically no clear relationship between autocorrelation values and problem hardness was observed for GP

# Fitness Landscapes in GP are very complex, but...

"Why Ants are Hard?" **Langdon, Poli, 1998**

Enumeration of a small fraction of the total search space and random sampling characterise it as rugged with many multiple plateaus split by deep valleys and many local and global optima. This suggests it is difficult for hill climbing algorithms.

Many other similar studies in "Foundations of Genetic Programming", **Langdon, Poli, 2002**.

This book also contains an important first step towards the study of problem hardness using the results obtained for the Schema Theorem.

# Relationship between Neutrality and Evolvability
# T. Yu, J. Miller 2001

Neutrality is particularly interesting in GP since functional redundancy and introns naturally foster neutrality

```
G1: nor (and x1 x2) (nor x1 x2)
G2: nor (nand (nand x1 x2) (or x1 x2)) (nor x1 x2)
G3: nor (and x1 x2) (nor (nor x1 x2) (nand x1 x2))
```

different programs (genotypes), same functional behavior (phenotype)

*Implicit neutrality*

Yu and Miller introduce *explicit neutrality* and a way to measure it for Cartesian GP.

# Neutrality Measured with Hamming distance

Let G be an individual in the population at a certain time step.

Let G1 be an individual obtained by mutating G.

If G and G1 have the same fitness (the mutation is neutral), Yu and Miller accept G1 as a legal offspring (and thus allow him to take part in the evolution) only if G and G1 have a smaller Hamming distance than a given constant k.

Changing this constant k (Hamming distance threshold) allows us to control the ***amount of allowable neutral mutations***, i.e. the ***amount of neutrality***.

# Yu and Miller Results (Synthesis)

Larger amount of neutrality allow GP to generate fitter individuals

(results criticized by Collins, 2005)

# Relationship between Code Growth and Problem Difficulty: Gustafson, Ekárt, et al., 2004

They used two different types of symbolic regression increased instance difficulty.

## Results

Increased difficulty induces higher selection pressure and less genetic diversity, which both contribute toward an increased rate of code growth

# Discussion

Problem difficulty is

- bound to neutrality
- bound to code growth
- bound to tree-shapes (Daida et al., 2001)
- ...

but....

we still miss mathematical measures
of problem hardness

# Hardness Measures for GP

- Fitness-Distance Correlation (*fdc*)
- Negative Slope Coefficient (*nsc*)

Collaborators:

- Marco Tomassini (University of Lausanne, Switzerland)
- Philippe Collard (University of Nice-Sophia Antipolis, France)
- Manuel Clergue (University of Nice-Sophia Antipolis, France)
- Sébastien Verel (University of Nice-Sophia Antipolis, France)

# Fitness Distance Correlation (*fdc*) *[T. Jones, 1995]*

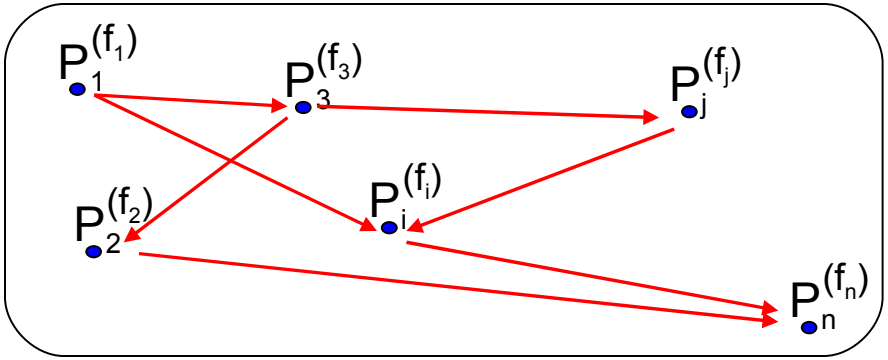
Given a sample of *n* individuals, let's suppose to know:

- the set $F = \{f_1, f_2, \ldots, f_n\}$ of the individual fitnesses
- the genotype of the global optimum (individual with the best fitness)
- a measure to express the genotypic distance between individuals

Let $D = \{d_1, d_2, \ldots, d_n\}$ be the *n* distances to the global optimum, then

***fdc* is the correlation between sets *F* and *D***

## Main idea




**Space of programs**

- Notion of **distance**.
- Relationship between **fitness** and **distance to the goal**.

# First use of the fdc in GP: Nicolaev and Slavov, 1998

The used the *fdc* to chose a mutation operator among a set of given ones

In 2005 we tried to use the fdc for GP much more in the same way Jones intended to use it for GAs.

# *fdc* as tool for problem hardness *[T. Jones, 1995]*

For GAs, problems can be classified in three classes:

- **Misleading** (*fdc* 0.15) in which fitness increases with distance.
- **Difficult** (-0.15 < *fdc* < 0.15) in which there is no correlation between fitness and distance.
- **Straightforward** (*fdc* -0.15) in which fitness increases as the global optimum approaches.

**To (experimentally) verify if the same property is also valid for GP:**

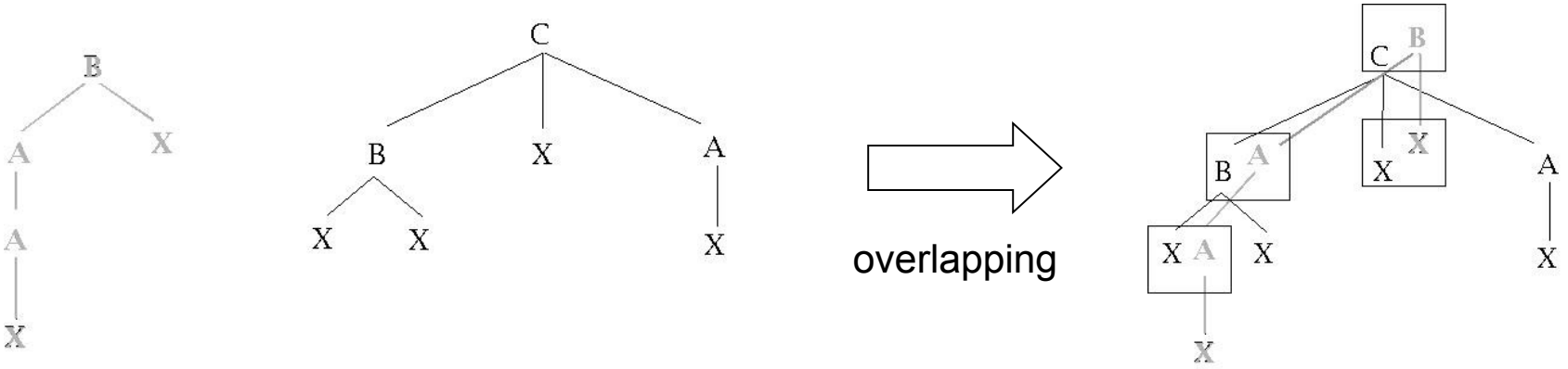
First step: to choose a distance between genotypes (trees!)

# Our approach

- To chose a distance between genotypes to calculate fdc
- To define some genetic operators ***consistent*** with this distance
- To test fdc on a set of functions

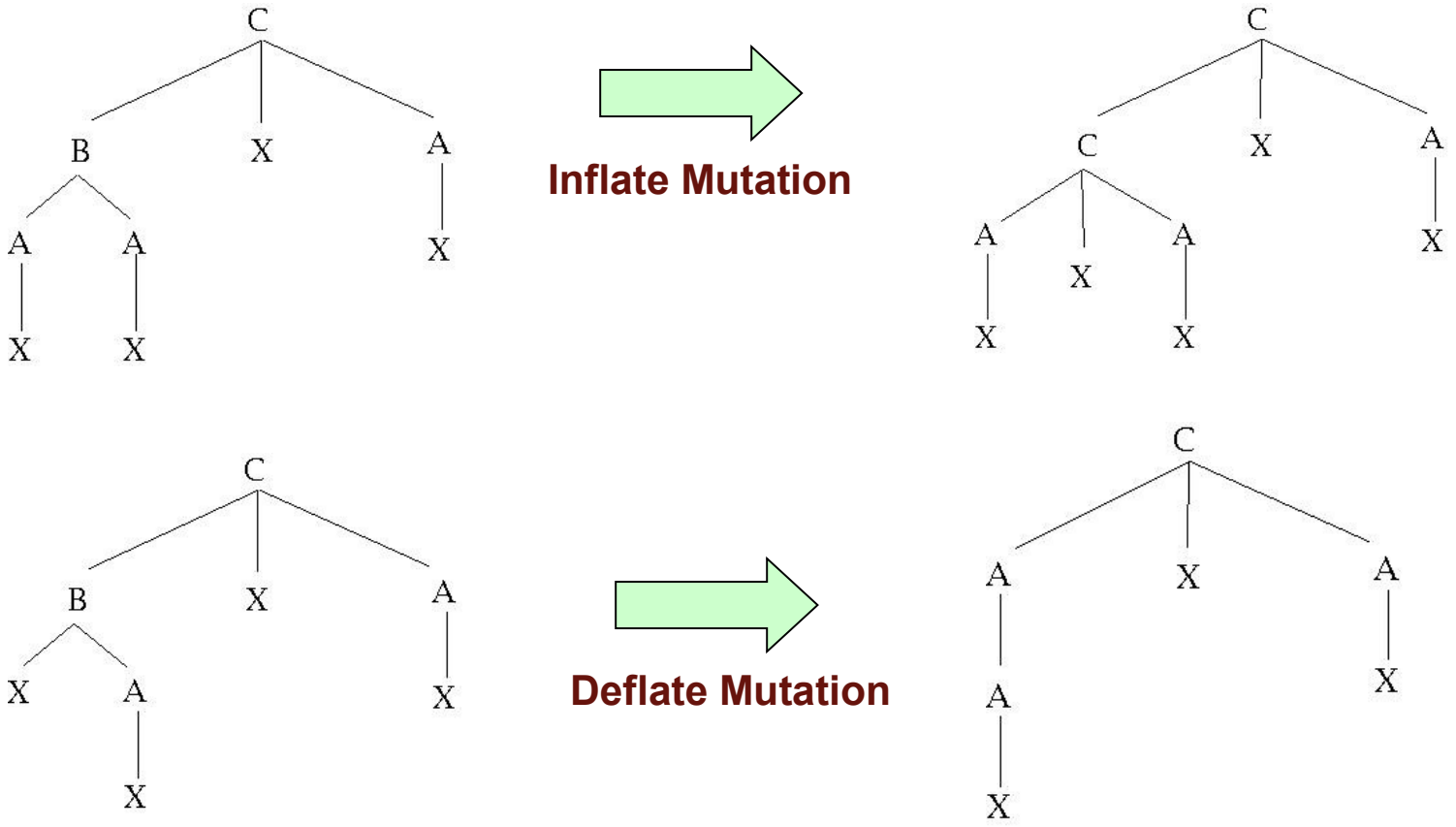
# Structural Distance (Intuition)

[Ekàrt-Németh 2002]




- We assign a *weight* to each node
- We calculate the difference of the weights of nodes at corresponding positions
- The distance is the weighted sum of these differences

# Operators of Structural Mutation




GP based only on these operators:
***S*tructural *M*utation *G*enetic *P*rogramming (*SMGP*).**

# Property (Distance/Operators Consistency)

Let:

- F = {A, B, C, ...}  T = {X}
- s t.q. s  {F  T} : c(s) = arity(s) + 1
- $T_1$ et $T_2$ two trees composed by symbols  {F  T}
- k = 1, z = 1

**If**

*dist* ($T_1$, $T_2$) = *D*

**then**

*$T_2$ can be obtained from $T_1$ with a sequence of D/2 operations of structural mutation*

# Summary of *fdc* results

*Fdc* correctly measures the difficulty of:

- Unimodal and Multimodal Trap Functions (Deb, Goldberg)
- Royal Trees (Punch)
- Max Problem (Gathercole)

Are we happy ?

# *fdc* drawbacks

- Existence of counterexamples

  Ridged Royal Trees
  (inspired by the counterexample for GAs of [Quick *et al*., 1998])

- Not a predictive measure

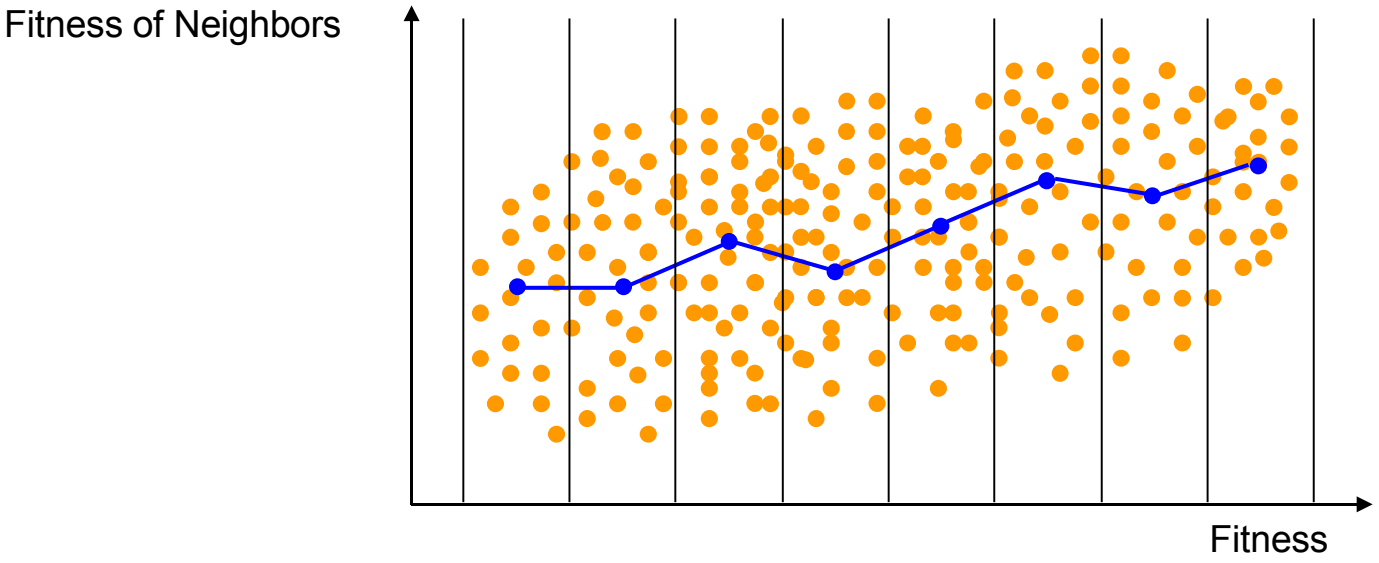
  Optima must be known "a priori"
  (this drawback makes *fdc* "*almost*" unusable in practical cases)

A new measure is needed to quantify the difficulty of "*real*" problems.

The measure we have proposed is based on the concept of ***fitness clouds***.

# Measure of Problem Hardness Based on Fitness Clouds Negative Slope Coefficient (*nsc*)

- A fitness cloud is partitioned into *n* bins
- For each bin, a point is calculated, such that its abscissa is the average of the abscissas and its ordinate is the average of the ordinates.
- All these points are joined by segments $\{S_1, S_2, \ldots, S_{n-1}\}$



# Negative Slope Coefficient Definition

$$nsc = \sum_{i=1}^{n-1} p_i$$

where, $i \in [1, n-1]$

$p_i = min\{0, \text{slope}(S_i)\}$

**Hypothesis**:

- $nsc = 0$ the problem is ***easy***
- $nsc < 0$ the problem is ***difficult*** and the magnitude of *nsc* quantifies the difficulty

**Idea**:

If $nsc < 0$ then there is at least one area of the fitness landscape where evolvability is bad.

# Sampling the search space and the neighborhoods

Main idea:

Evolvability makes sense if it is calculated on "good" individuals ("bad" ones are probably discarded by selection).

Sampling the search space:

Importance sampling (Metropolis-Hastings technique)

Sampling the neighborhoods:

selection (tournament selection of size 10).

# Summary of *nsc* results

- Good hardness indicator for:
  - Trap Functions
  - Royal Trees
  - Binomial-3 Problem [Daida *et al.,* 2001]
  - Even Parity Problem [Koza, 1992]
  - Artificial Ant on the Santa Fe Trail [Koza, 1992]
- Many ways of calculating the *nsc* have been used:
  - Number of neighbors for each sampled individual
  - Number of mutations to generate neighbors
  - Different types of mutations to generate neighbors
  - Different techniques to partition the fitness clouds into bins
- *nsc* is predictive it can be used on *any* problem
- *nsc* has not been normalized yet into a given range (classification of different problems by their difficulty)
- *nsc* lacks formal/theoretical justification

# A first step towards a theoretical justification of *nsc*

R. Poli and L. Vanneschi. Fitness-proportional negative slope coefficient as a hardness measure for genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2007*. ACM Press, 2007. Nominated for the *Best Paper Award* for the Genetic Algorithms track.

Presentation:
Monday 9 July
at 10:40
Room: Roberts G06

# What about Crossover?

# Modeling/Studying GP Crossover

- Schema Theorem [R. Poli and coworkers]
- Geometric Crossover [R. Poli and A. Moraglio]
- Homologous Crossover [M. Defoin-Platel, P. Collard et al.]
- Crossover (pseudo-)distance [S. Gustafson and L. Vanneschi]

# Crossover Distance

Collaborator:

Steven Gustafson (GE Global Research, Niskayuna, NY, USA)

# The (Basic) Idea

We don't have to count *how many crossovers* it takes to transform a tree $T_1$ into another tree $T_2$, but *how probable it is* to obtain $T_2$ by applying crossover to $T_1$ (in just one step!).

[S. Gustafson, L. Vanneschi, *Operator based distance for Genetic Programming: Subtree Crossover Distance*, EUROGP 2005]

*Subtree Crossover Distance* (SCD)
between two trees $T_1$ and $T_2$
=
*Probability of:*
- Selecting a subtree $S_{T1}$ from $T_1$, and
- Finding a subtree $S_{T2}$ in the population P

*Such that:*
Replacing $S_{T1}$ with $S_{T2}$ in $T_1$ we get $T_2$

# Terminology

- SCD is a *probability*!
- SCD between two trees $T_1$ and $T_2$ is a function of $T_1$, $T_2$ *and the population* (P) in which $T_1$ and $T_2$ are!

Thus

SCD is *NOT* a distance (metric) !!

We need a *similarity / dissimilarity measure (for subtree crossover)*, not necessarily an (Euclidean) distance metric.

The term *pseudo-distance* would be more appropriate.

# SCD Definition

$$
\begin{aligned}
&\text{func SCD}(T_1, T_2, P)\{\\
&\quad S = diff(T_1, T_2)\\
&\quad res = 1\\
&\quad \forall (s^i_{T_1}, s^i_{T_2}) \in S:\\
&\qquad ps1 = probSelecting(s^i_{T_1}, T_1)\\
&\qquad ps2 = probCreating(s^i_{T_2}, P)\\
&\qquad res = res * (1 - ps1 * ps2)\\
&\quad return(res)\\
&\}
\end{aligned}
$$

the complexity is "reasonable"!

The operator *diff(T1, T2)* returns the set

$$S = \{(s^1_{T_1}, s^1_{T_2}), (s^2_{T_1}, s^2_{T_2}), \ldots, (s^n_{T_1}, s^n_{T_2})\}$$

such that:

if we replace $s^i_{T_1}$ with $s^i_{T_2}$ ( $i$ [1, n]) in $T_2$ we obtain $T_1$

# Summary of Crossover-Distance results

SCD appropriate for:

- Measuring the FDC *dynamically* (during evolution)
- Fitness Sharing

Our hypothesis: SCD appropriately models subtree crossover

- SCD diversity behave differently than ED diversity (slightly increasing and larger than zero for successful runs, approximately zero for unsuccessful runs)

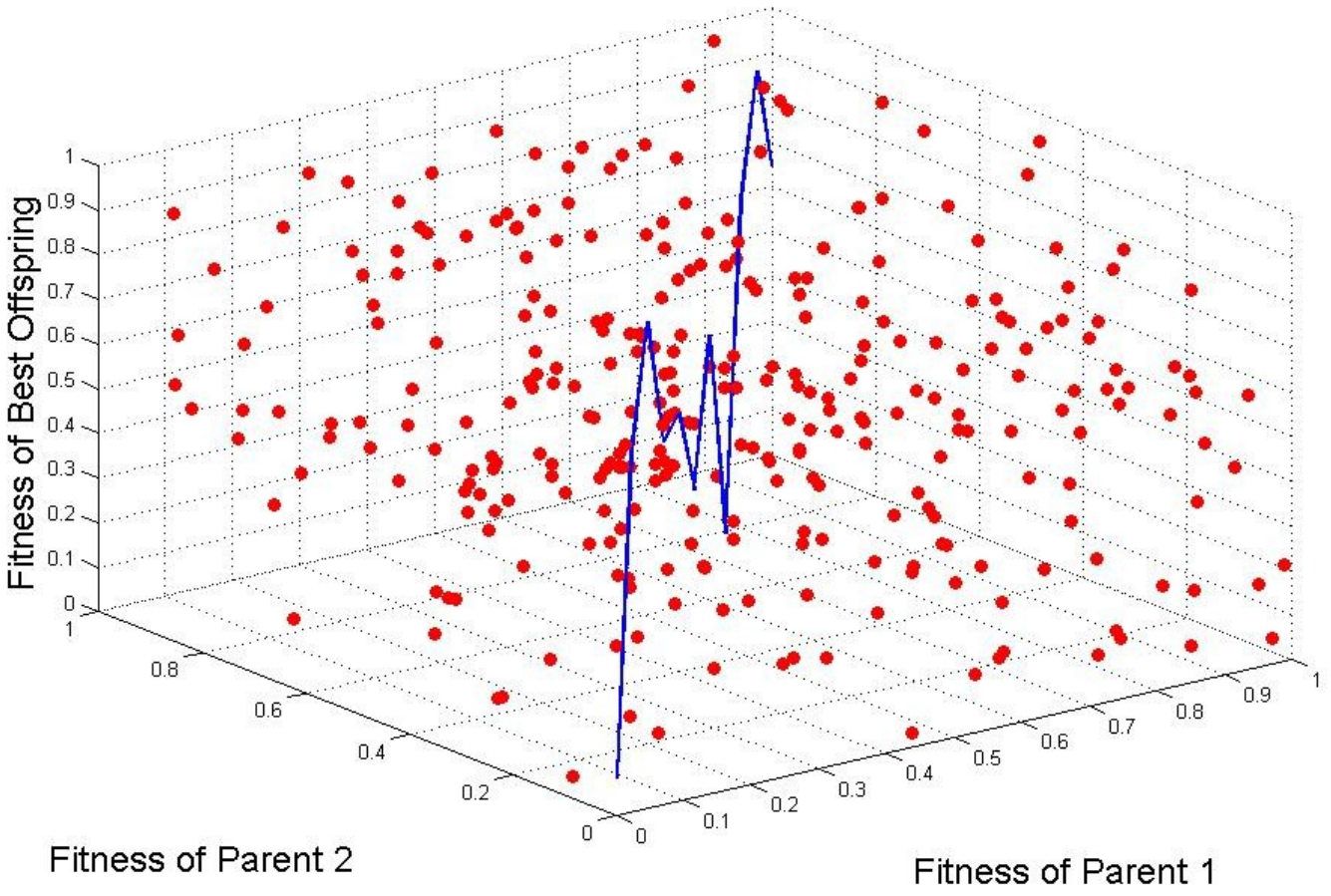
Can it be used to predict the behavior of GP runs?

## Last Discussion:

## Can we define an NSC "with" crossover?

# A possible idea

- Generate two samples of individuals $S_1$ and $S_2$
- Repeat
    - Take one individual $i_1$ from $S_1$, one individual $i_2$ from $S_2$
    - Perform the crossover between $i_1$ and $i_2$, let $j_1$ and $j_2$ be the offspring
    - Let j = best ($j_1$, $j_2$)
    - Plot the triple ($i_1$, $i_2$, j) on a 3D plane
    - Eliminate $i_1$ from $S_1$ and $i_2$ from $S_2$
- Until $S_1$ and $S_2$ are empty


Fitness of Best Offspring
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
0.8
0.6
0.4
0.2
0
Fitness of Parent 2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Fitness of Parent 1

# Bibliography


- J. M. Daida, R. Bertram, S. Stanhope, J. Khoo, S. Chaudhary, O. Chaudhary
  What makes a problem GP-hard? Analysis of a tunably difficult problem in genetic programming.
  *Genetic Programming and Evolvable Machines*, 2:165–191, 2001.

- J. M. Daida, H. Li, R. Tang, A. M. Hilss
  What makes a problem GP-hard? Validating a hypothesis of structural causes.
  In R. Poli et al. editors *Genetic and Evolutionary Computation – GECCO-2003*, volume 2724 of *LNCS*, pages 1665–1677. Springer-Verlag, Berlin.

- K. E. Kinnear
  Fitness landscapes and difficulty in genetic programming.
  In *Proceedings of the First IEEE Conference on Evolutionary Computing*, pages 142–147. IEEE Press, Piscataway, NY.

- J. R. Koza
  *Genetic Programming*
  The MIT Press, Cambridge, Massachusetts, 1992

- W. B. Langdon, R. Poli
  *Foundations of Genetic Programming*
  Springer, 2002

- N. I. Nikolaev,V. Slavov.
  Concepts of inductive genetic programming.
  In W. B. Langdon et al. editors,
  *Genetic Programming, Proceedings of EuroGP'1998*,
  volume 1391 of *LNCS*, pages 49–59. Springer-Verlag, 1998

- M. Tomassini, L. Vanneschi, P. Collard, M. Clergue
  A study of fitness-distance correlation as a difficulty measure in genetic programming
  *Evolutionary Computation*, 13(2): 213-239, 2005

- L. Vanneschi, M. Tomassini, P. Collard, S. Verel
  Negative Slope Coefficient. A measure to characterize genetic programming fitness landscapes
  In P. Collet et al. editors, *Genetic Programming, 9th European Conference, EuroGP 2006*, pages 178-189, Lecture Notes in Computer Science. 2006

- L. Vanneschi, S. Gustafson, G. Mauri
  Using subtree crossover distance to investigate genetic programming dynamics
  In P. Collet et al. editors, *Genetic Programming, 9th European Conference, EuroGP 2006*, pages 238-249, Lecture Notes in Computer Science. 2006

- T. Yu, J. Miller
  Neutrality and the evolvability of boolean function landscape.
  In J. Miller et al., editor, *Proceedings of the Fourth European Conference on Genetic Programming* (*EuroGP-2001*), volume 2038 of LNCS, pages 204–217, Lake Como, Italy, 2001. Springer, Berlin, Heidelberg, New York. Lecture notes in Computer Science vol. 2038.

- W. B. Langdon and R. Poli,
  Why Ants are Hard.
  In *Genetic Programming* 1998: Proceedings of the Third Annual Conference Morgan Kaufmann, J. R. Koza et al. editors, pages = 193-201, 1998

- S. Gustafson, A. Ekárt, E. K. Burke, G. Kendall
  Problem Difficulty and Code Growth in Genetic Programming
  *Genetic Programming and Evolvable Machines,* 2004, Volume: 5, Issue: 3, p. 271-290