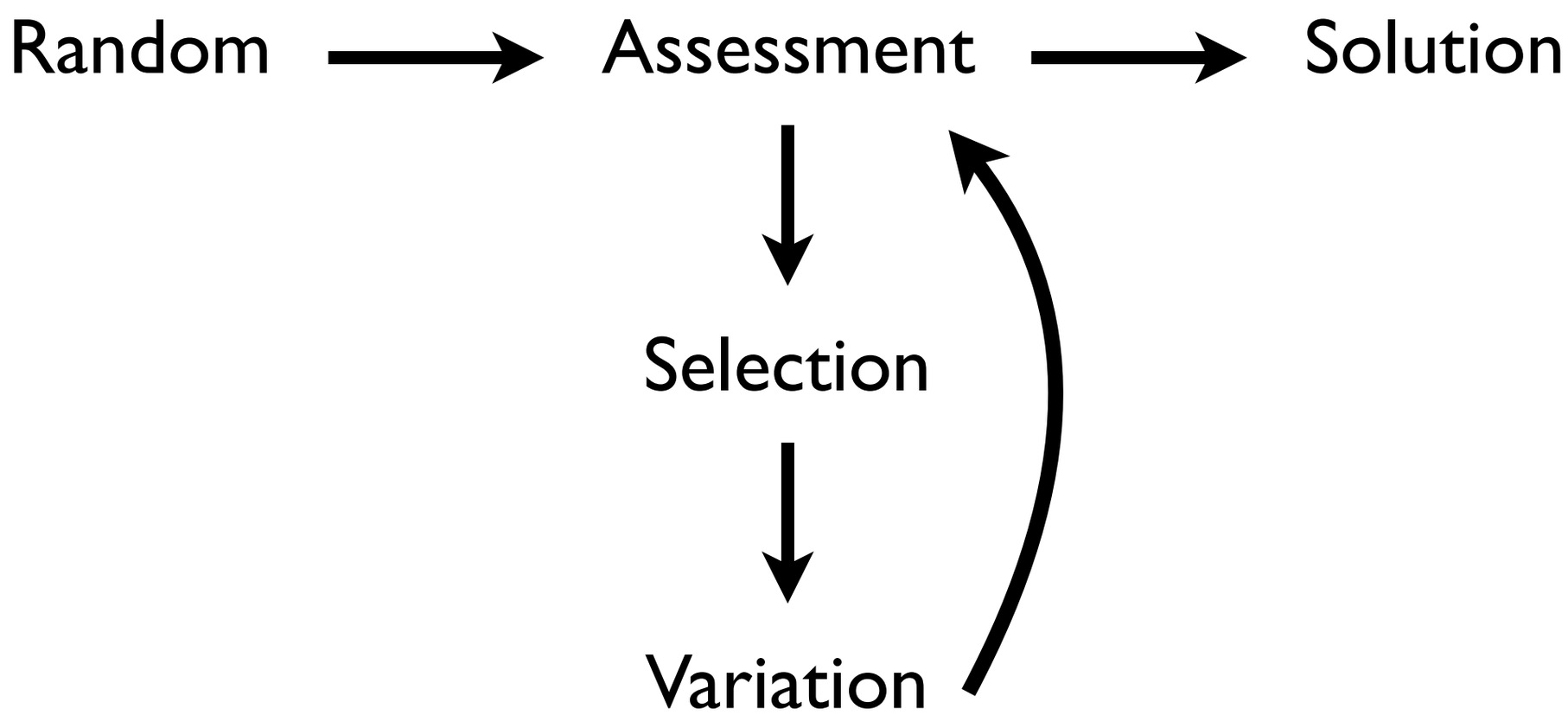
Evolving evolution,
in computers,
to solve some of the world's hardest problems

Lee Spector
Cognitive Science, Hampshire College
Information and Computer Sciences, UMass Amherst
http://hampshire.edu/lspector

Outline

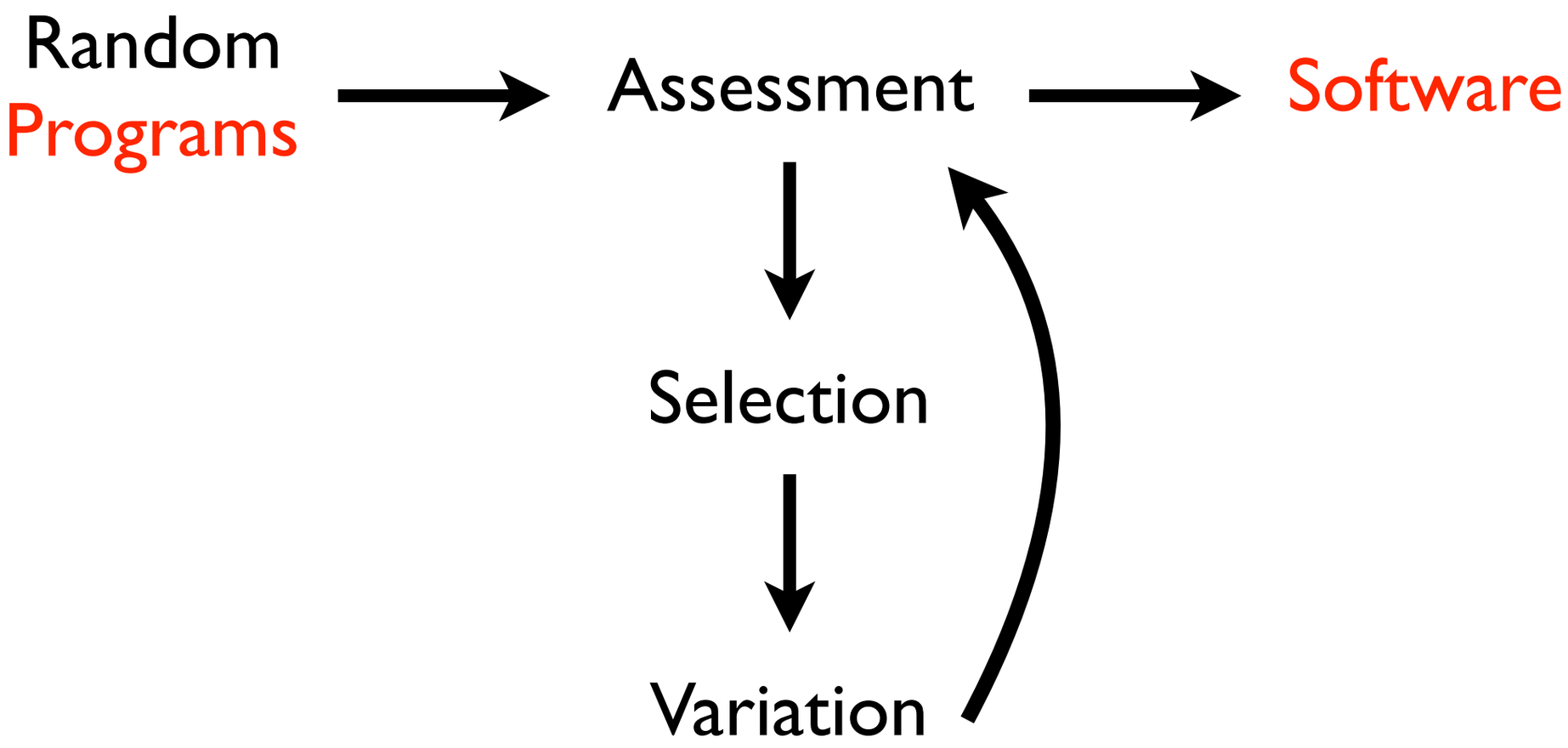
- Evolutionary computation
- Evolved solutions
- Evolving evolution

Evolutionary Computation



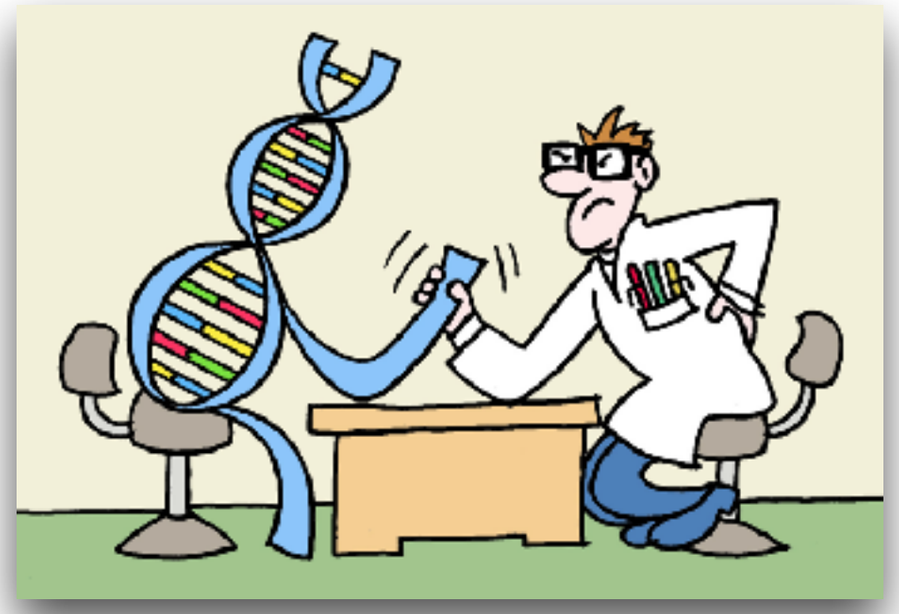
Evolving Lego Bridges

Genetic Programming



Analyzing a Decade of Human-Competitive ("HUMIE") Winners: What Can We Learn?

Karthik Kannappan, Lee Spector, Moshe Sipper, Thomas Helmuth, William Lacava, Jake Wisdom, Omri Bernstein



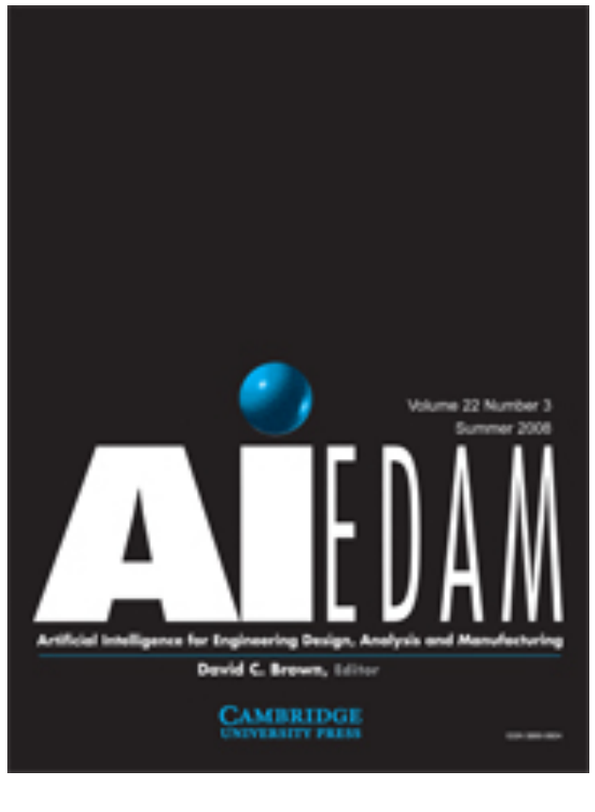
- Competition for evolutionary computation results
- Up to $10,000 in cash prizes
- Held annually since 2004

Humies Criteria

- The result was ***patented as an invention*** in the past is an improvement over a patented invention or would qualify today as a patentable new invention.
- The result is equal to or better than a result that was accepted as a ***new scientific result*** at the time when it was published in a peer-reviewed scientific journal.
- The result is equal to or better than a result that was placed into a database or archive of results maintained by an ***internationally recognized panel of scientific experts***.
- The result is ***publishable in its own right*** as a new scientific result independent of the fact that the result was mechanically created.
- The result is equal to or better than the ***most recent human-created*** solution to a long-standing problem for which there has been a succession of increasingly better human-created solutions.
- The result is equal to or better than a result that was considered an ***achievement in its field*** at the time it was first discovered.
- The result solves a problem of ***indisputable difficulty*** in its field.
- The result holds its own or wins a regulated ***competition involving human contestants*** (in the form of either live human players or human-written computer programs).

Humies Winners

- Application areas: antennas, biology, chemistry, computer vision, electrical engineering, electronics, games, image processing, mathematics, mechanical engineering, medicine, operations research, optics, optimization, photonics, physics, planning, polymers, quantum computing, security, software engineering
- Winningest technique: genetic programming
- Winningest problem type: design



AIEDAM
Artificial Intelligence for Engineering Design, Analysis and Manufacturing

VOLUME 22 SUMMER 2008 NUMBER 3

SPECIAL ISSUE

Genetic Programming for Human-Competitive Designs

Guest Editor

LEE SPECTOR

Evolution, the Designer

WHAT WOULD DARWIN SAY? | LEE SPECTOR

The Boston Globe

And now, digital evolution

By Lee Spector | August 29, 2005

RECENT developments in computer science provide new perspective on "intelligent design," the view that life's complexity could only have arisen through the hand of an intelligent designer. These developments show that complex and useful designs can indeed emerge from random Darwinian processes.

"Darwinian evolution is itself a designer worthy of significant respect, if not religious devotion."

Yavalath

Yavalath is an abstract board game for two or three players, invented by a computer program called Ludi. It has an easy rule set that any player can pick up immediately, but which produces surprisingly tricky emergent play.

Yavalath is available from nestorgames, making it the first — and still only — computer-generated game to be commercially published, together with its sister game Pentalath.

In October 2011, Yavalath was ranked in the top #100 abstract board games ever invented on the BoardGameGeek database. This helped it win the GECCO "Humies" gold medal for human-competitive results in evolutionary computation for 2012.

Here is a Yavalath article in the November 2013 issue of Bitcoin magazine.

Rules

The board starts empty.

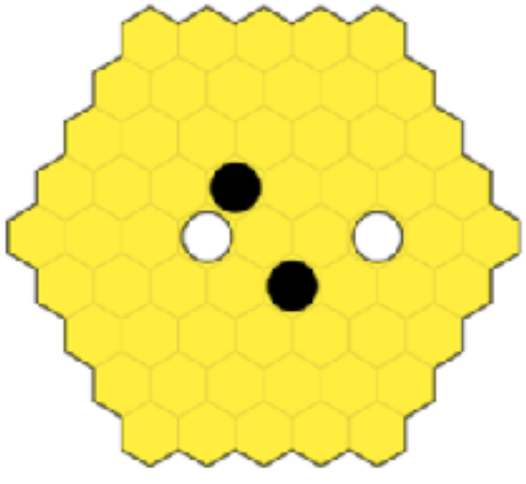
Two players take turns adding a piece of their colour to an empty cell.

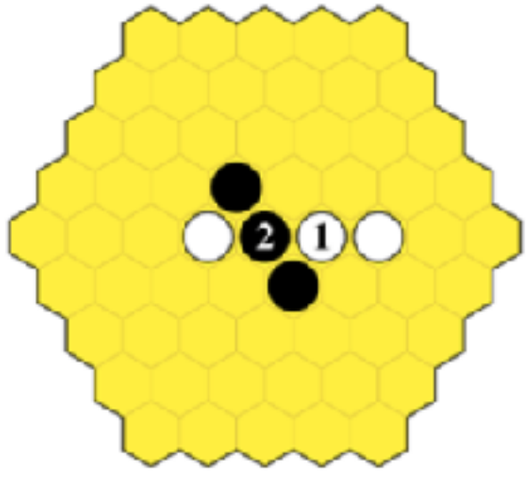
Win by making a line-of-4 (or more) pieces of your colour.
Lose by making a line-of-3 pieces of your colour beforehand.
Draw if the board otherwise fills up.

No, players are not allowed to pass.

Tactics and Strategy

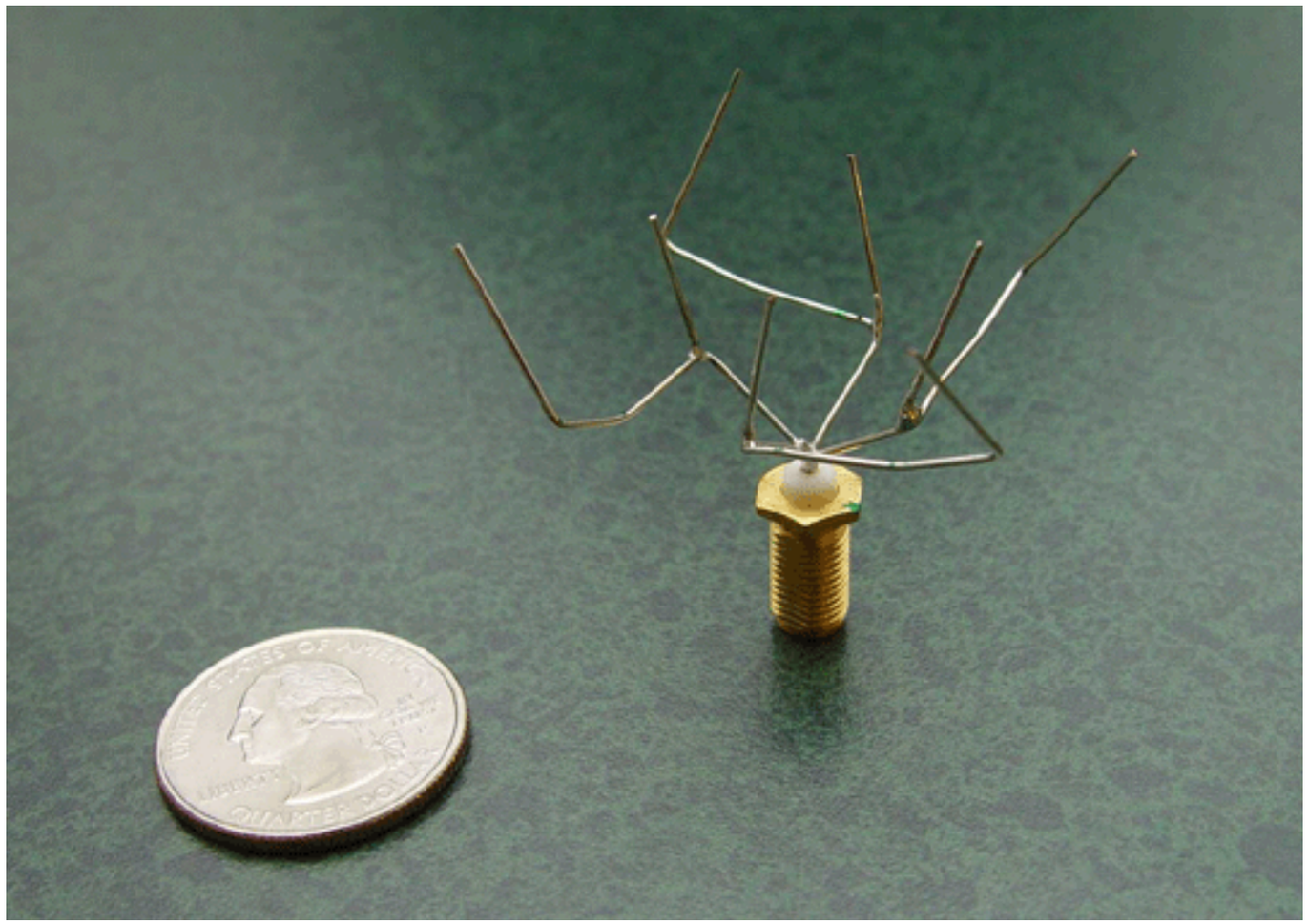
The key tactical play in Yavalath is the *forcing move*, as shown below. White move 1 forces Black to lose with the blocking move 2.





Humies Gold Medal, 2012

Evolved Antenna

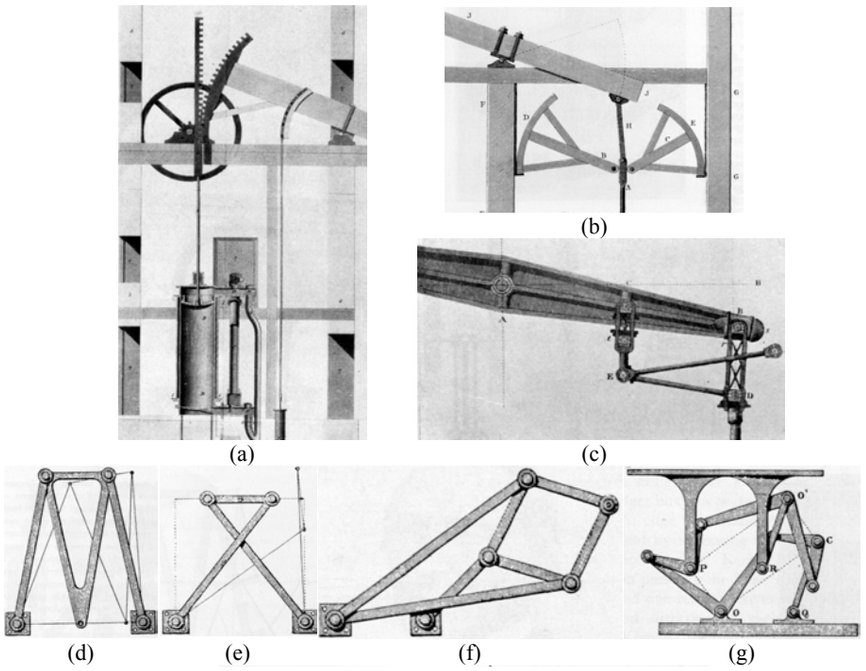


NASA Space Technology 5 Mission, Lohn, Hornby, and Linden

Humies Gold Medal, 2004

Line-Drawing Mechanism

- Without reference to an existing straight line
- Human-competitive results; challenged world's greatest inventors for a century (spanning 18th and 19th)



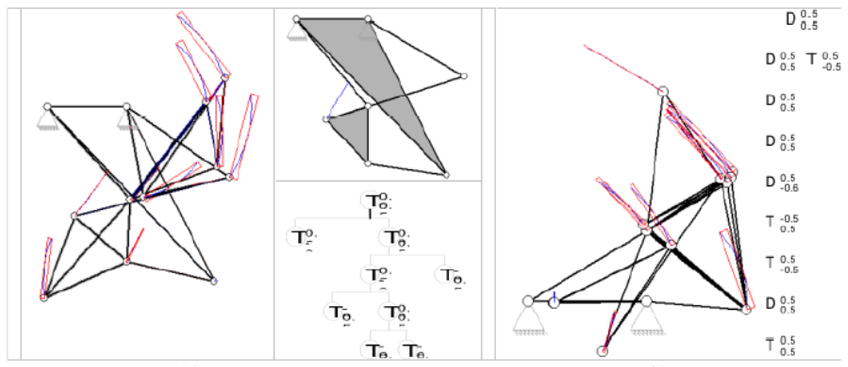
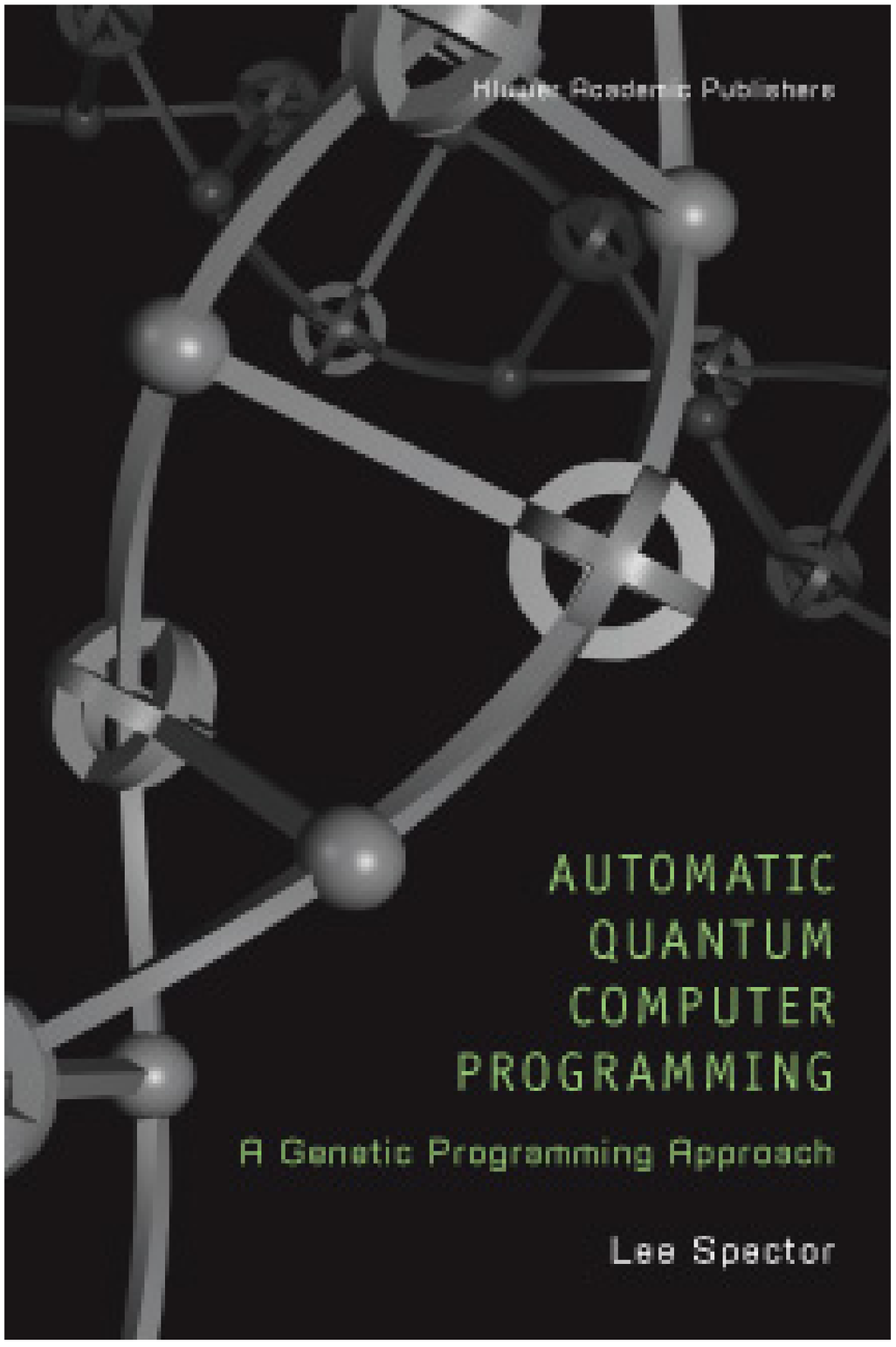


Fig. 10. Two Evolved mechanisms and their tree representations (a) Linearity 1:12819; The simplified equivalent shown top right, and (b) Linearity 1:4979.

Lipson, H. 2004. How to Draw a Straight Line Using a GP: Benchmarking Evolutionary Design Against 19th Century Kinematic Synthesis. GECCO-2004.

Humies Silver Medal, 2012



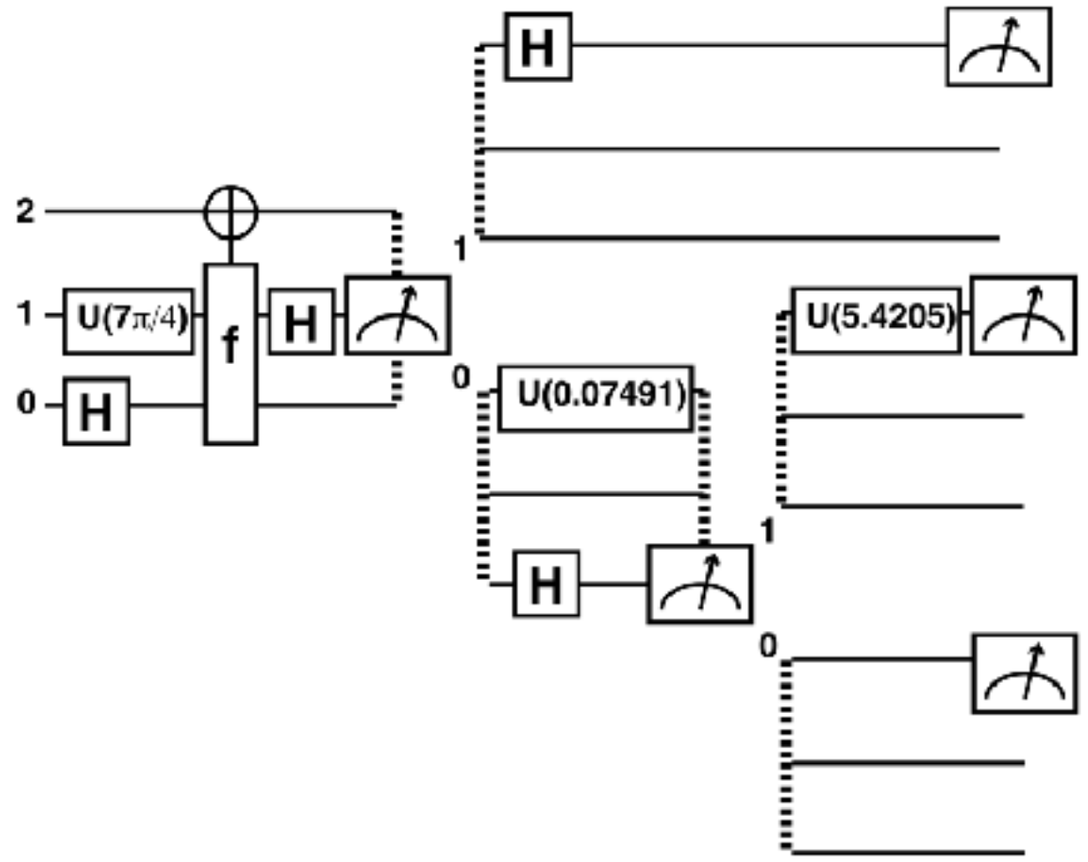


Figure 8.11. A gate array diagram for an evolved solution to the AND/OR oracle problem. The gate marked "f" is the oracle. The sub-diagrams on the right represent the possible execution paths following the intermediate measurements.

Humies Gold Medal, 2004

Genetic Programming for Finite Algebras

Lee Spector
Cognitive Science
Hampshire College
Amherst, MA 01002
lspector@hampshire.edu

David M. Clark
Mathematics
SUNY New Paltz
New Paltz, NY 12561
clarkd@newpaltz.edu

Ian Lindsay
Hampshire College
Amherst, MA 01002
iml04@hampshire.edu

Bradford Barr
Hampshire College
Amherst, MA 01002
bradford.barr@gmail.com

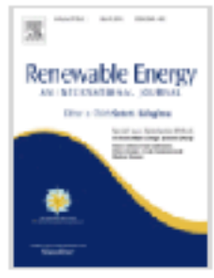
Jon Klein
Hampshire College
Amherst, MA 01002
jk@artificial.com


```
((((((((((x*(y*x))*x)*z)*(z*x))*((x*(z*(x*(z*y))))*z))*
z)*z)*(z*((((x*(((z*z)*x)*(z*x)))*x)*y)*(((y*(z*(z*
y)))*(((y*y)*x)*z))*(x*(((z*z)*x)*(z*(x*(z*y))))))))))
```

Humies Gold Medal, 2008

Renewable Energy

Volume 87, Part 2, March 2016, Pages 892–902



Automatic identification of wind turbine models using evolutionary multiobjective optimization

William La Cava[a], Kourosh Danai[a], Lee Spector[b], Paul Fleming[c], Alan Wright[c], Matthew Lackner[a]

Show more

https://doi.org/10.1016/j.renene.2015.09.068

Highlights

- Accurate, succinct models of wind turbine dynamics are identified from normal operating data.
- A novel evolutionary multi-objective optimization system is described.
- The proposed method produces physically meaningful models without prior knowledge of the system.
- The method is bench-marked against other modeling techniques.

Genetic programming representations for multi-dimensional feature learning in biomedical classification

William La Cava[1], Sara Silva[2,3], Leonardo Vanneschi[4], Lee Spector[5], and Jason Moore[1]

[1] Institute for Biomedical Informatics, University of Pennsylvania, Philadelphia PA, USA
lacava@upenn.edu
[2] BioISI - Biosystems & Integrative Sciences Institute, Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa, 1749-016 Lisboa, Portugal
[3] CISUC, Department of Informatics Engineering, University of Coimbra, Portugal
[4] NOVA IMS, Universidade Nova de Lisboa, 1070-312 Lisbon, Portugal
[5] School of Cognitive Science, Hampshire College, Amherst MA, USA

Abstract. We present a new classification method that uses genetic programming (GP) to evolve feature transformations for a deterministic, distanced-based classifier. This method, called M4GP, differs from common approaches to classifier representation in GP in that it does not enforce arbitrary decision boundaries and it allows individuals to produce multiple outputs via a stack-based GP system. In comparison to typical methods of classification, M4GP can be advantageous in its ability to produce readable models. We conduct a comprehensive study of M4GP, first in comparison to other GP classifiers, and then in comparison to six common machine learning classifiers. We conduct full hyper-parameter optimization for all of the methods on a suite of 16 biomedical data sets, ranging in size and difficulty. The results indicate that M4GP outperforms other GP methods for classification. M4GP performs competitively with other machine learning methods in terms of the accuracy of the produced models for most problems. M4GP also exhibits the ability to detect epistatic interactions better than the other methods.

Keywords: genetic programming, feature learning, classification

Intro Programming

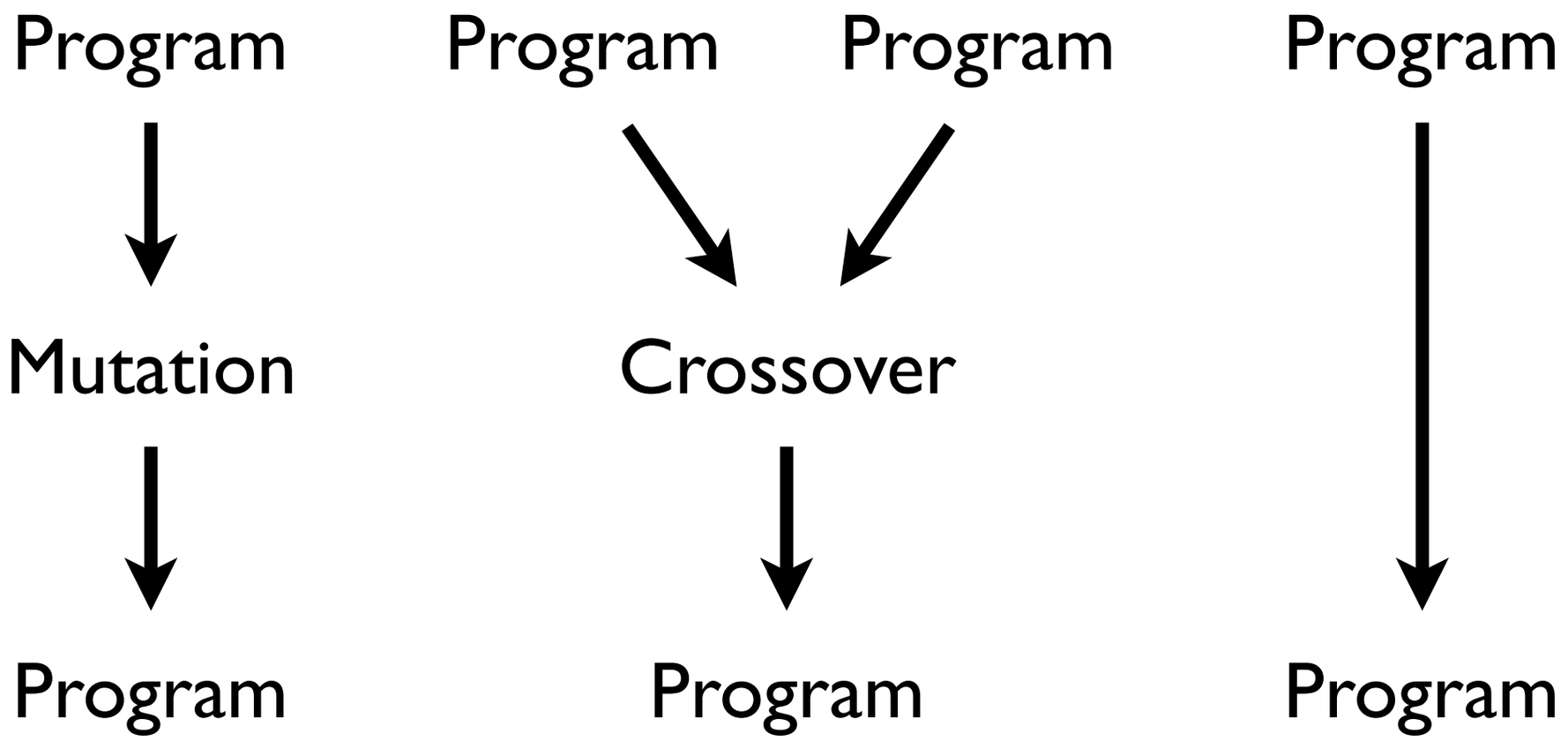
Number IO, Small or Large, For Loop Index, Compare String Lengths, Double Letters, Collatz Numbers, Replace Space with Newline, String Differences, Even Squares, Wallis Pi, String Lengths Backwards, Last Index of Zero, Vector Average, Count Odds, Mirror Image, Super Anagrams, Sum of Squares, Vectors Summed, X-Word Lines, Pig Latin, Negative to Zero, Scrabble Score, Word Stats, Checksum, Digits, Grade, Median, Smallest, Syllables

7. **Replace Space with Newline (P 4.3)** Given a string input, print the string, replacing spaces with newlines. Also, return the integer count of the non-whitespace characters. The input string will not have tabs or newlines.

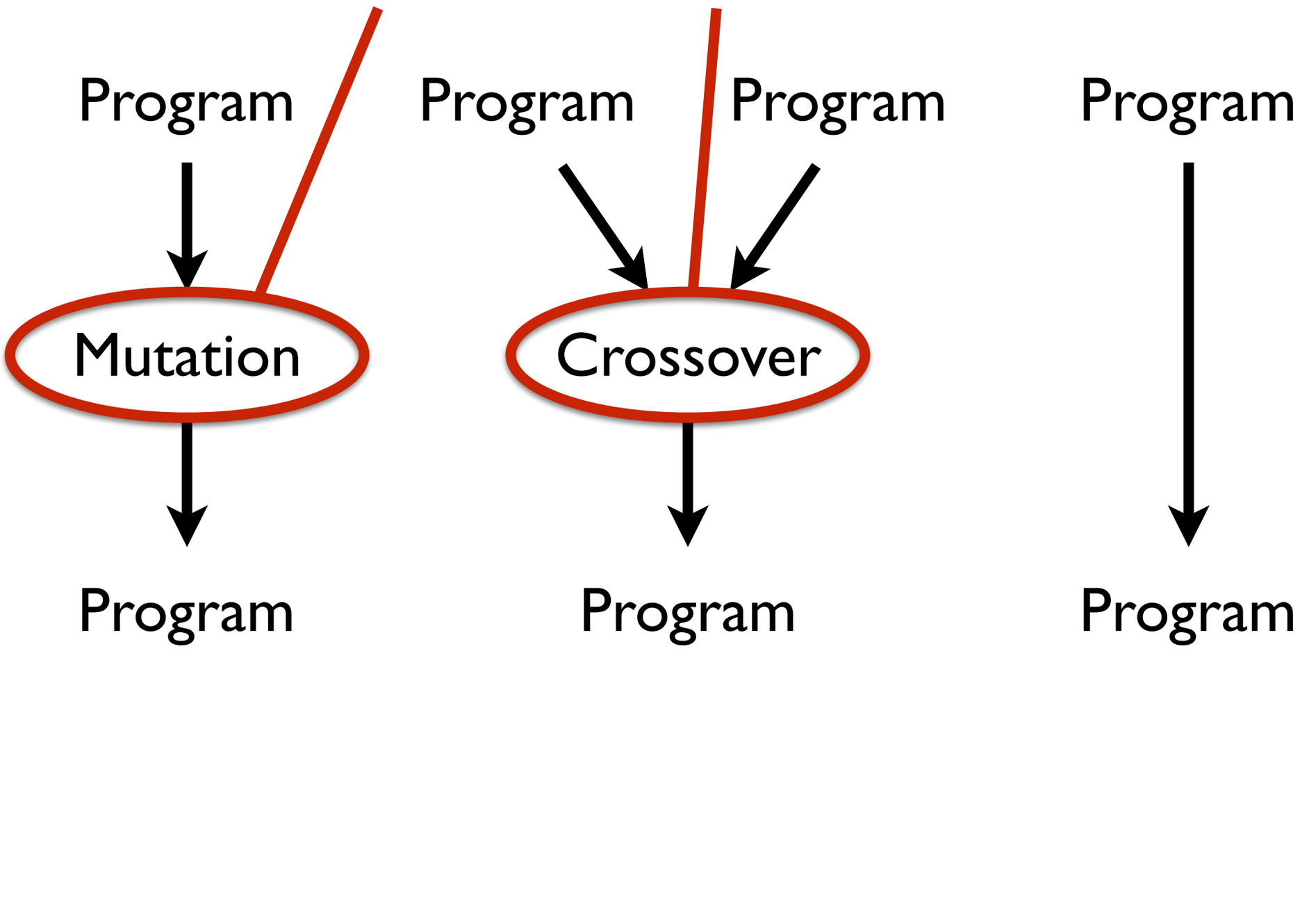
- Multiple types, looping, multiple tasks
- Simplified solution:

```
(\space char_dup exec_dup in1 \newline string_replacechar
print_string string_removechar string_length)
```

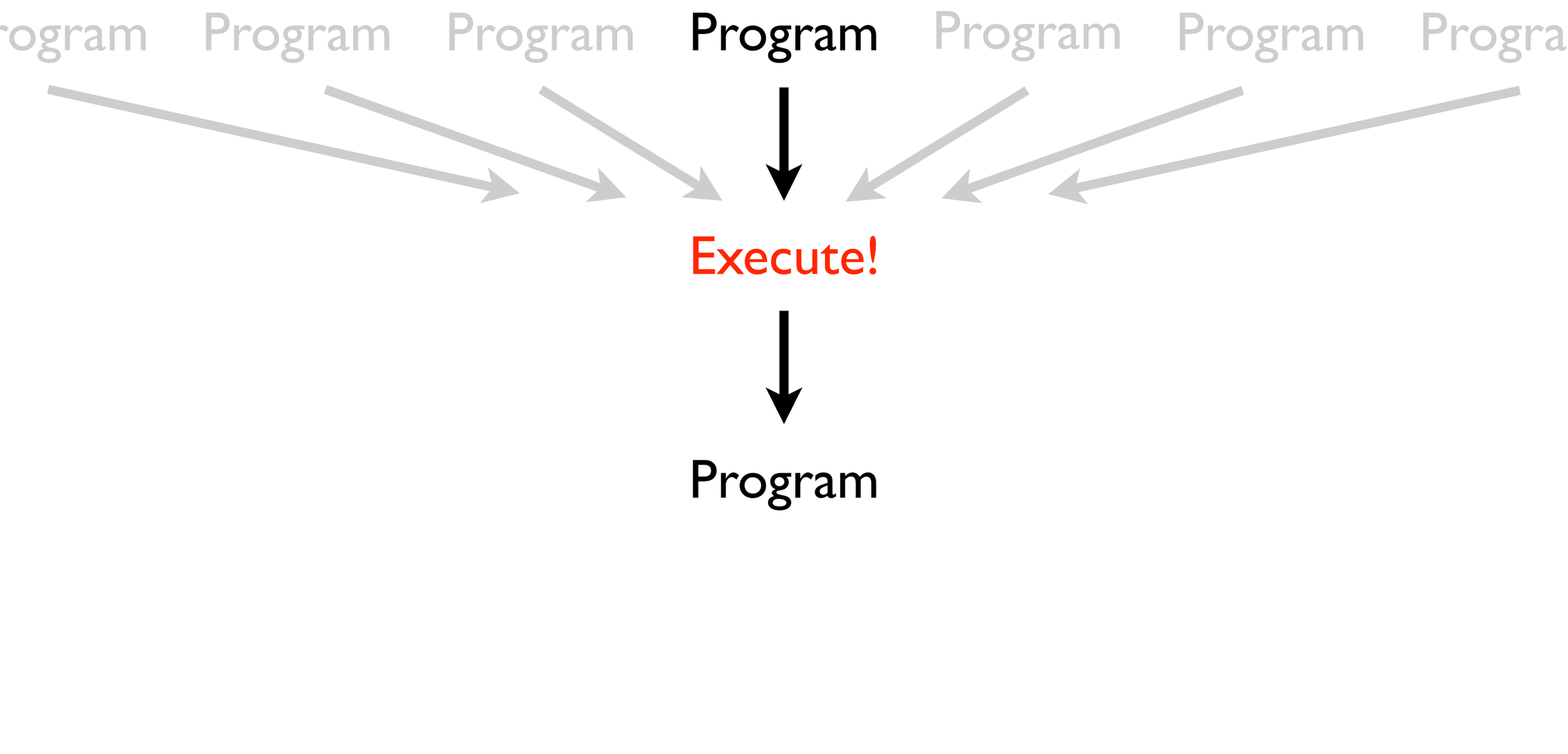
Variation in GP



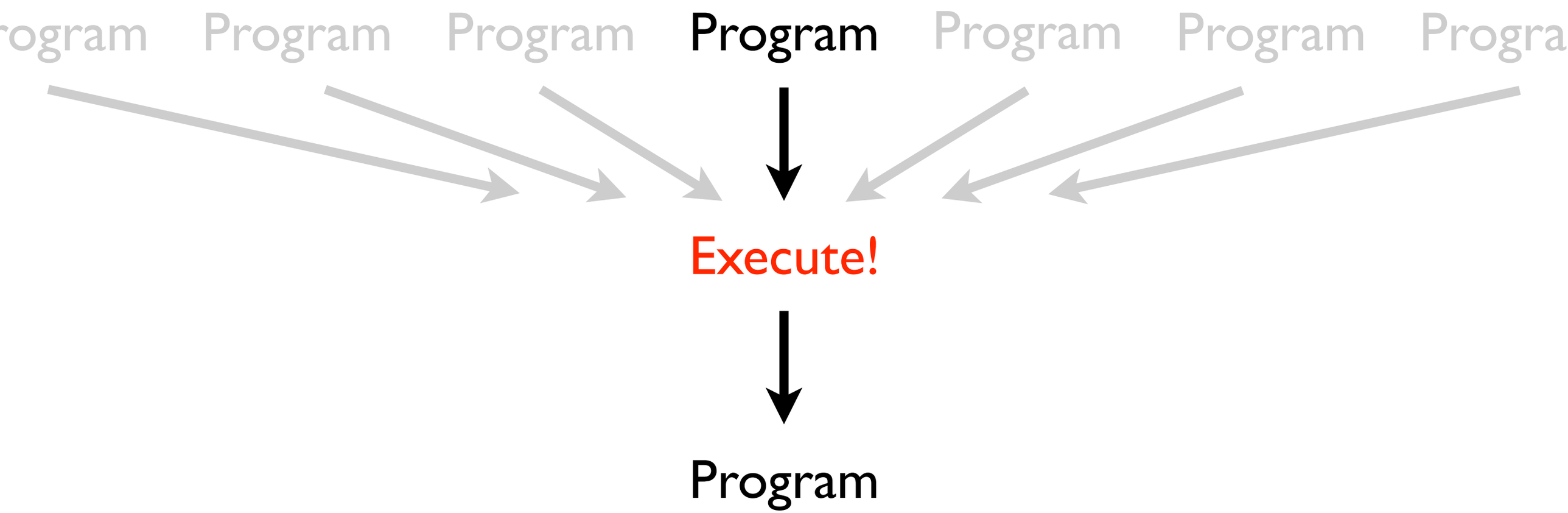
Variation in GP



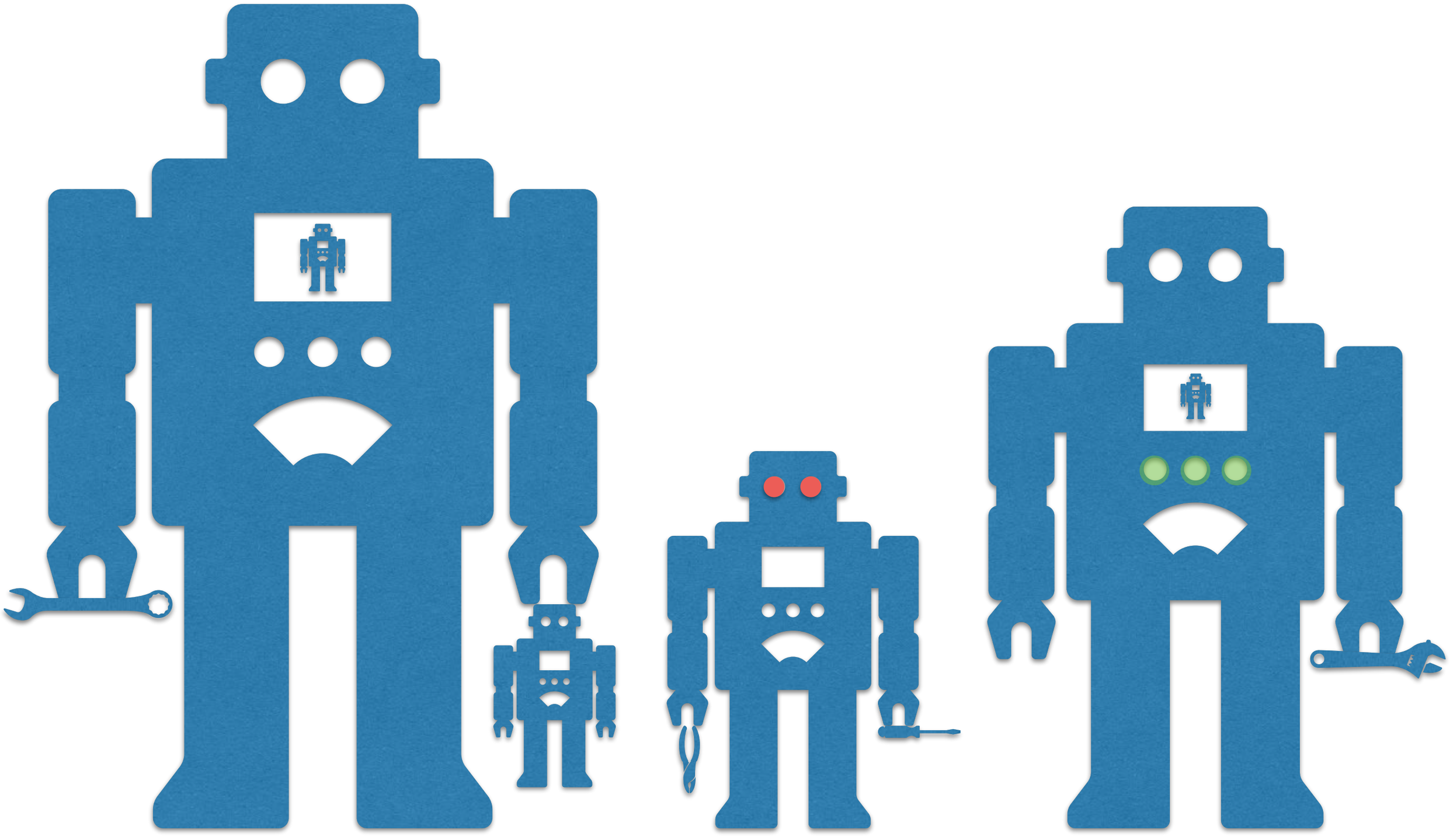
Autoconstruction



Autoconstruction



A bit more complicated when genomes distinguished from programs



Autoconstructive Evolution

- Evolve evolution while evolving solutions
- How? Individuals produce and vary their own children, with methods that are subject to variation
- Requires understanding the evolution of variation
- Hope: May produce EC systems more powerful than we can write by hand

Needed for Evolution^2

- Diversity: Individuals vary
- Diversification: Individuals produce descendants that vary, in various ways
- Recursive Variance: Individuals produce descendants that vary in the ways that they vary their offspring

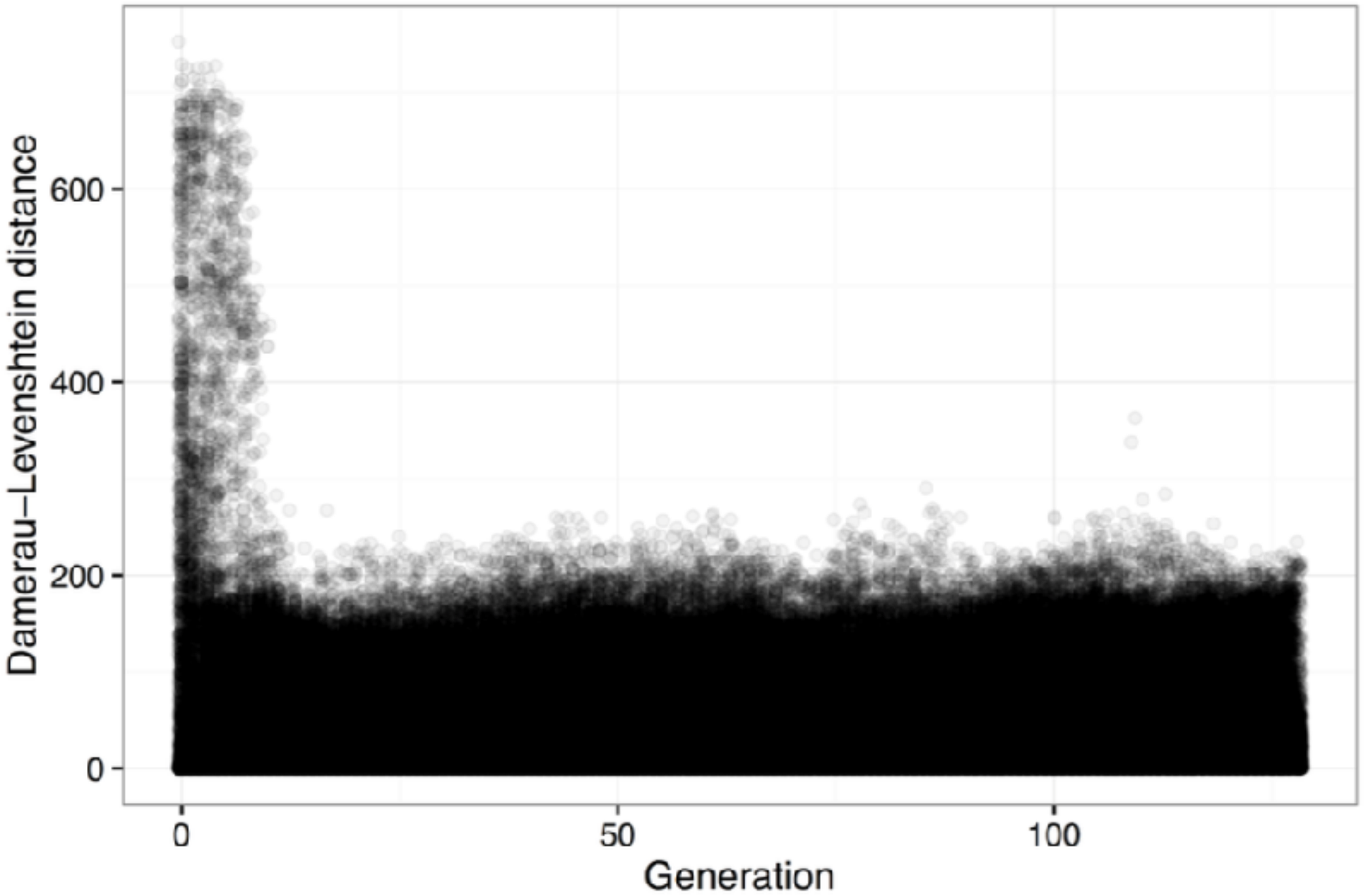


Figure 1: DL-distances between parent and child during a single non-autoconstructive run of GP on the Replace Space With Newline problem

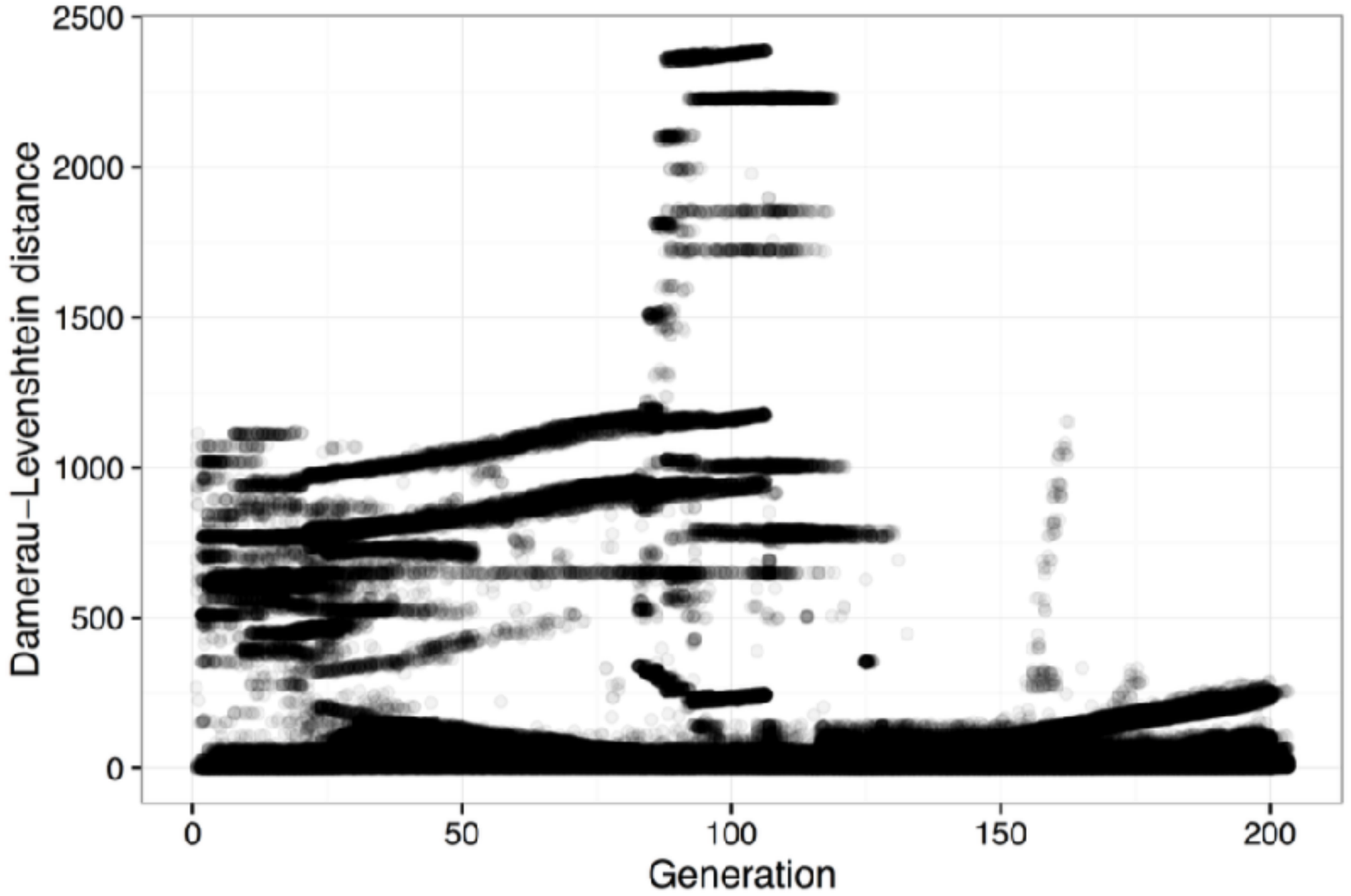


Figure 3: DL-distances between parent and child during a single autoconstructive run of GP on the Replace Space With Newline problem

Rivaling Ordinary GP

- Autoconstructive evolution can succeed as much and as fast as non-autoconstructive evolution
- In 20 runs in one configuration, 75% success within 300 generations on Replace Space With Newline (100% by generation 628)
- Surprising!

8. **String Differences (P 4.4)** Given 2 strings (without whitespace) as input, find the indices at which the strings have different characters, stopping at the end of the shorter one. For each such index, print a line containing the index as well as the character in each string. For example, if the strings are "dealer" and "dollars", the program should print:

```
1 e o
2 a l
4 e a
```

Extending GP's Reach

- Without autoconstruction, string difference not yet solved by GP, despite many efforts/configurations
- 3 autoconstructive solutions so far

Prospects

- Evolutionary computation is already solving important, hard problems
- If it can be applied to itself, to evolve as it runs, then it will be able to solve harder problems
- We are beginning to see how this might work
- Help would be appreciated!

Clojush

Lee Spector (lspector@hampshire.edu), started 20100227 See version history. Older version history is in `old-version-history.txt`.

This is the README file accompanying Clojush, an implementation of the Push programming language and the PushGP genetic programming system in the Clojure programming language. Among other features this implementation takes advantage of Clojure's facilities for multi-core concurrency.

Availability

https://github.com/lspector/Clojush/

Requirements

To use this code you must have a Clojure programming environment; see http://clojure.org/. The current version of Clojush requires Clojure 1.7.0.

Clojure is available for most OS platforms. A good starting point for obtaining and using Clojure.

Quickstart

Using Leiningen you can run an example from the OS command line (in the Clojush directory) with a call like:

Pysh

Push Genetic Programming in Python. For the most complete documentation, refer to the ReadTheDocs.

http://pysh2.readthedocs.io/en/latest/

Push Genetic Programming

Push is programming language that plays nice with evolutionay computing / genetic programming. It is a stack-based language that features 1 stack per data type, including code. Programs are represented by lists of instructions, which modify the values on the stacks. Instuctions are executed in order.

More information about PushGP can be found on the Push Redux and the Push Homepage.

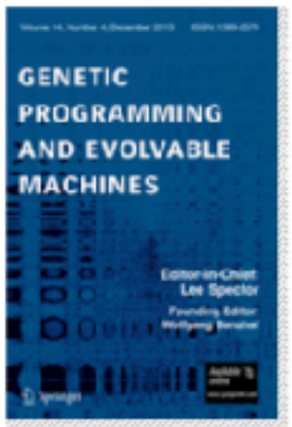
For the most cutting edge PushGP framework, see the Clojure implementaion called Clojush.

Installing Pysh

> Pysh is compatale with python `2.7.x` and `3.5.x`

Install from pip

Coming whith first beta release of `pyshgp`. Check the Roadmap to get a sense of how far off this is.



Genetic Programming and Evolvable Machines

Editor-in-Chief: Lee Spector

ISSN: 1389-2576 (print version)
ISSN: 1573-7632 (electronic version)

Journal no. 10710

Read Online

$99.00 Personal Rate e-only for the Americas

Get Subscription

- Online subscription, valid from January through December of current calendar year
- Immediate access to this years issues via SpringerLink
- 1 Volume(-s) with 4 issue(-s) per annual subscription
- Automatic annual renewal
- More Information: >> FAQs // >> Policy

ABOUT THIS JOURNAL | EDITORIAL BOARD | ETHICS & DISCLOSURES

Speed

54 No. of days from submission of the manuscript to first decision – 2016

26 No. of days from acceptance at publisher to published online – 2016

Usage

26,330 No. of downloads – 2016

149 Usage Factor 2015/2016

4 No. of articles Discussed via Social Media platforms – 2016

Impact

1.514 IF 2016

1.619 5 YR IF 2016

1.620 SNIP – 2016 Source Normalized Impact per Paper

0.454 SJR – 2016 SCImago Journal Rank

16 h5 Index – 2016

READ THIS JOURNAL ON SPRINGERLINK

Online First Articles

All Volumes & Issues

FOR AUTHORS AND EDITORS

2016 Impact Factor 1.514

Aims and Scope

Submit Online

Open Choice - Your Way to Open Access

Instructions for Authors

GPEM blog

SERVICES FOR THE JOURNAL

Contacts

Download Product Flyer

Shipping Dates

Order Back Issues

Bulk Orders

Article Reprints

ALERTS FOR THIS JOURNAL

Get the table of contents of every new issue published in Genetic Programming and Evolvable Machines.

Your E-Mail Address

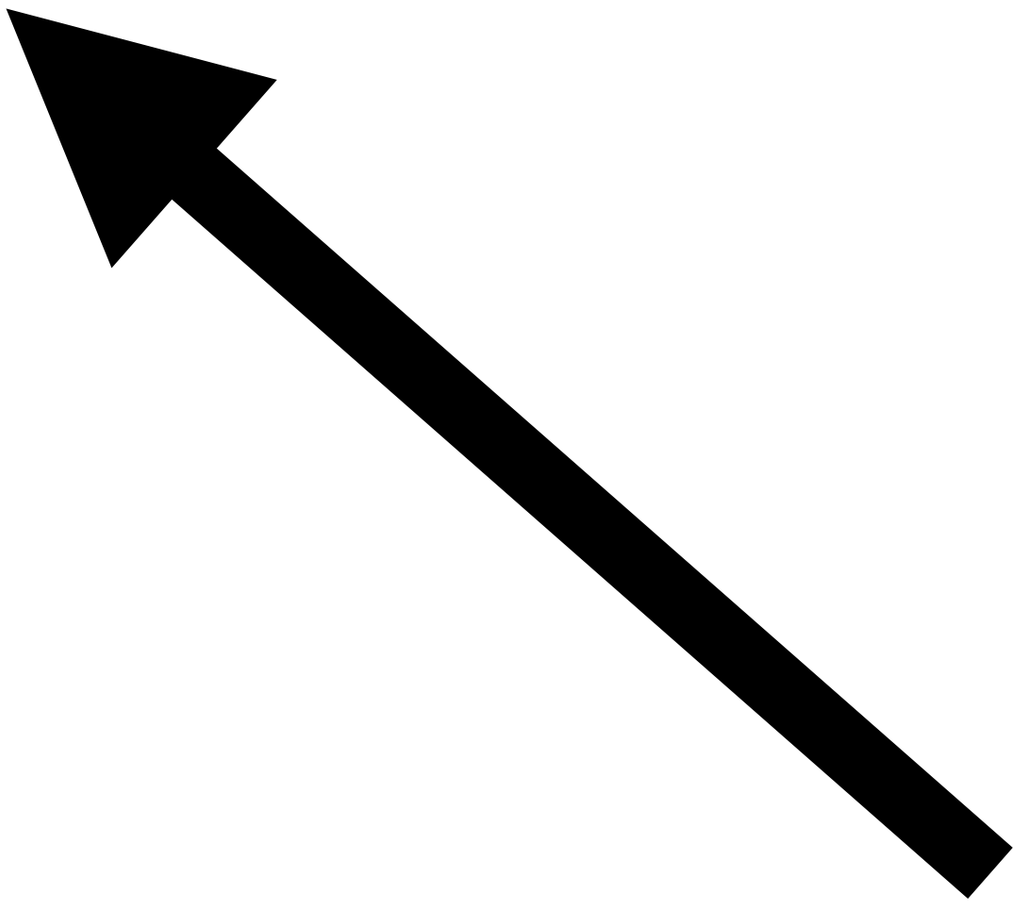
Risks

- Technology that we don't understand

Risks

- Technology that we don't understand
- Like children and other life forms

More



- pushlanguage.org
- http://hampshire.edu/lspector
- My office

Thanks

- Members of the Hampshire College Computational Intelligence Lab
- This material is based upon work supported by the National Science Foundation under Grants No. 1017817, 1129139, and 1331283. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.