Optimizing Complex Structures

Alberto TONDA

UMR 518 MIA-PS, INRAE, AgroParisTech, Université Paris-Saclay
UAR 3611, Institut des Systèmes Complexes de Paris Île-de-France

Outline

- Genetic Programming
- Linear Genetic Programming
- Grammatical Evolution
- Examples

Genetic Programming

- Genetic Programming
  - John Koza, 1992
  - Extend EAs to anything
  - Focus on computer programs

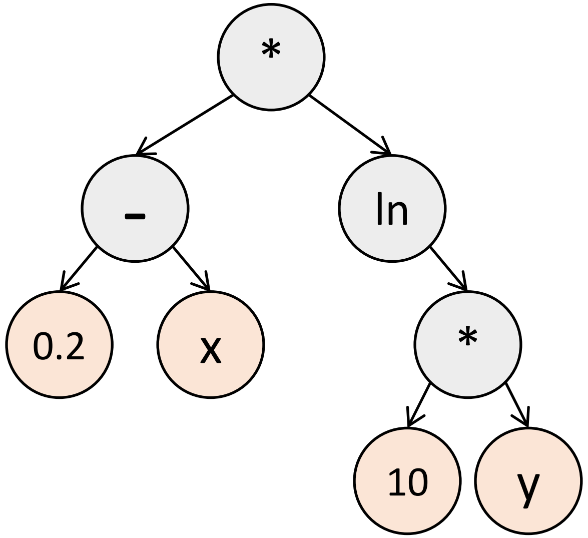
- Internal representation
  - Binary trees
  - Specialized mutations and crossovers
  - Search space difficult to characterize (or even visualize!)

Genetic Programming

- General idea: OPTIMIZE <u>ALL</u> THE THINGS with EAs
  - If you can *describe* a candidate solution to a problem…
  - …and *variators* (e.g. mutations, crossovers)…
  - …and you can *define a fitness/objective function*…
  - …EAs can explore the *space of all possible solutions*!

- It worked for *life on Earth*!
  - DNA is pretty complicated
  - Genome doesn't need to be *just numbers*

Genetic Programming

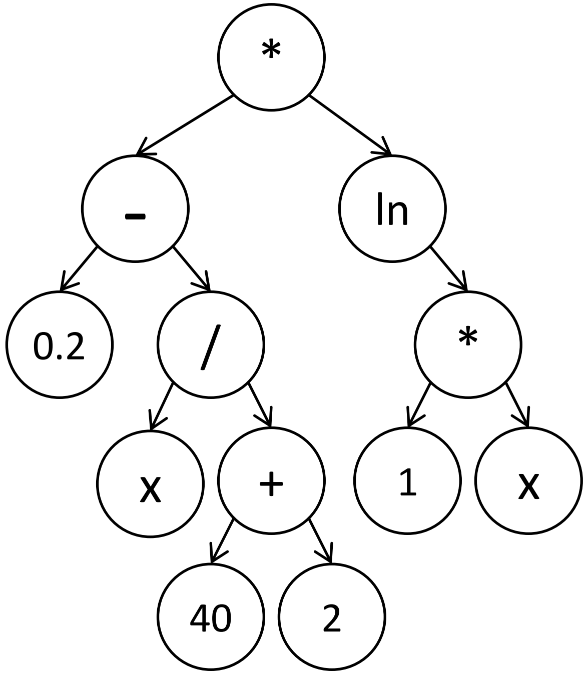


Operators: +, -, *, /, ln...

Terminals: reals, ints, vars, ...

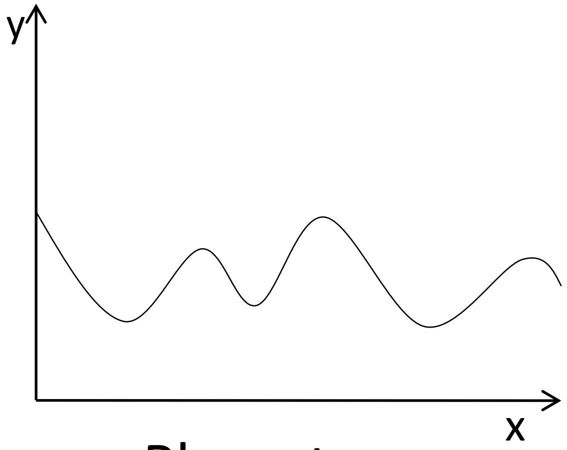
$$f(x,y) = (0.2 - x) * \ln(10 * y)$$

Symbolic Regression



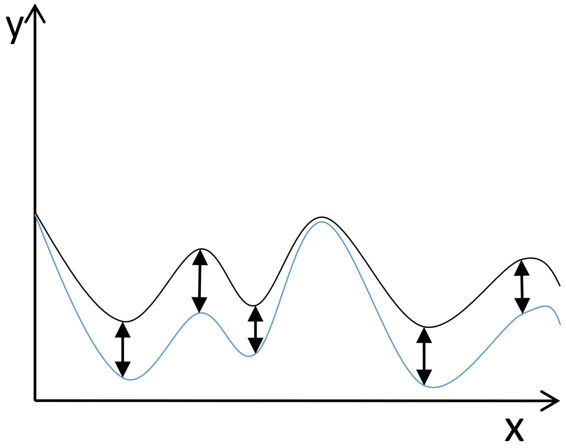
Genotype/Internal representation

f(x) = [0.2 – (x/42)] * ln(x)



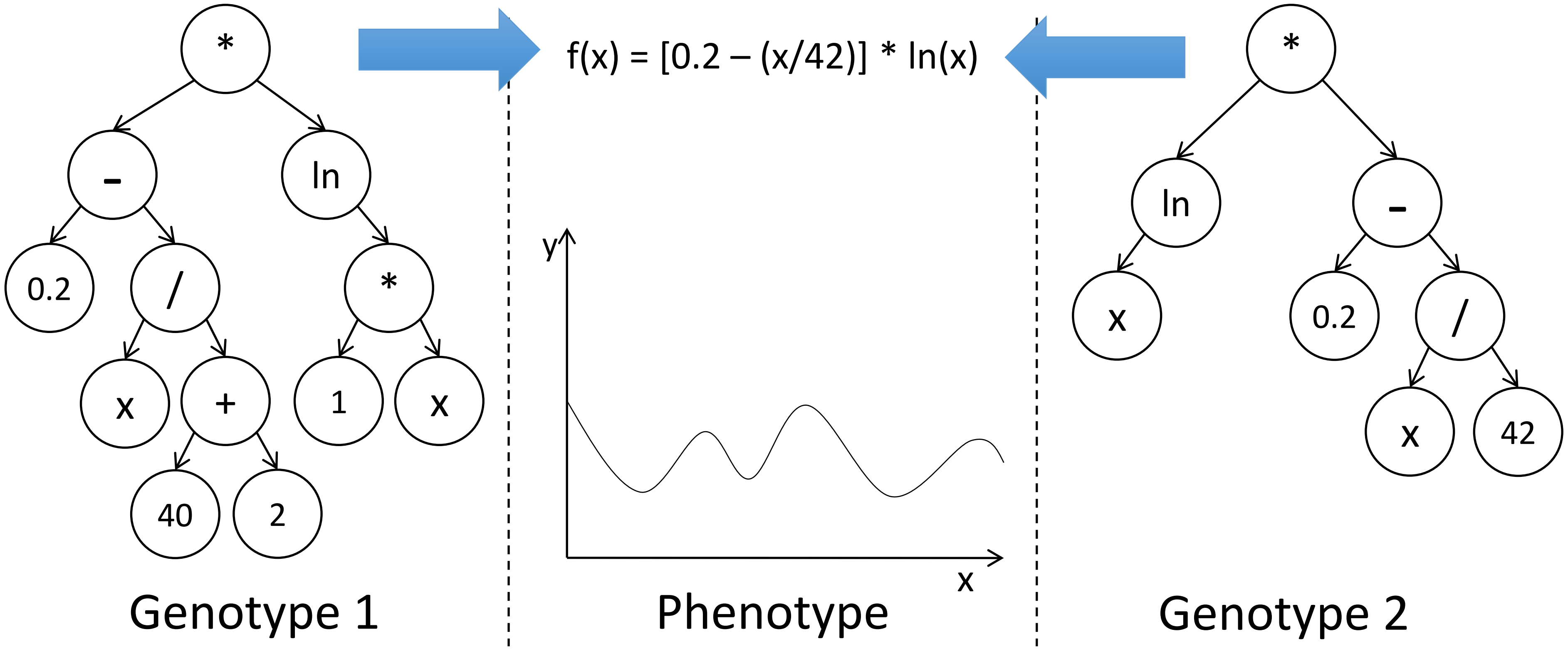
Phenotype

Fitness = $\sum_{i=0}^{N}$ abs(f(xi) −g(xi))

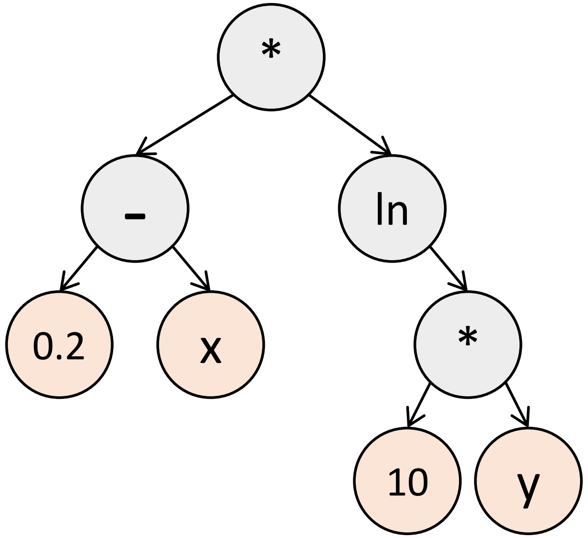


Fitness/Objective function

Symbolic Regression



Genetic Programming

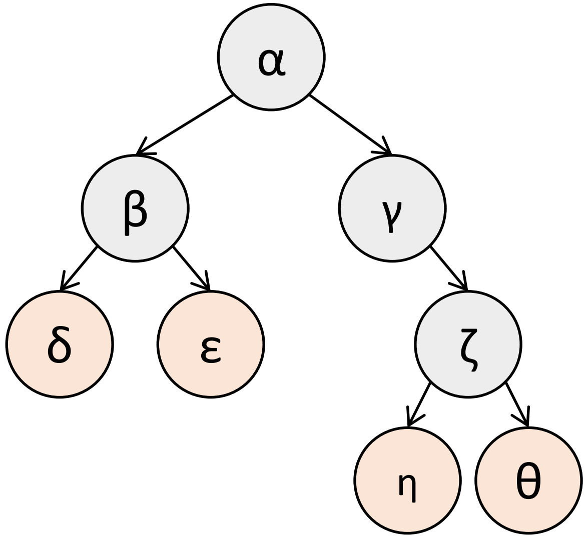


Operators: +, -, *, /, ln…

Terminals: reals, ints, vars, …

$$f(x, y) = (0.2 - x) * \ln(10 \ * y)$$

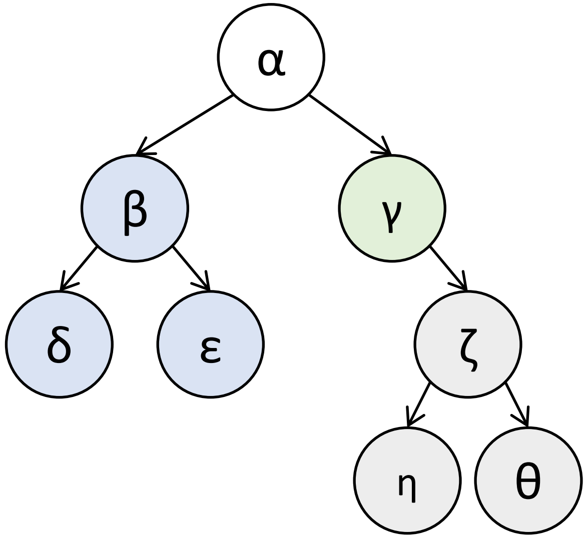
Genetic Programming



Operators: α, β, γ, ζ, λ, π, ς, …

Terminals: δ, ε, η, θ, ρ, σ, τ, …

Genetic Programming



Genotype

$$I = \alpha(\beta(\delta, \varepsilon), \gamma(\zeta(\eta, \theta)))$$

Phenotype

$$f(I)$$

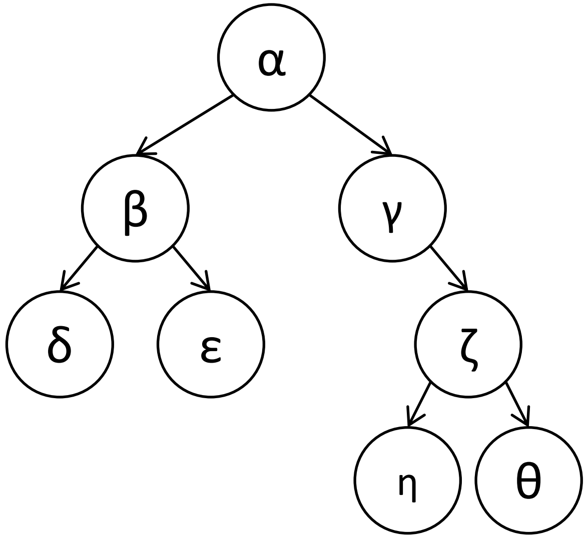
Fitness

Genetic Programming

- How to explore this complex search space?

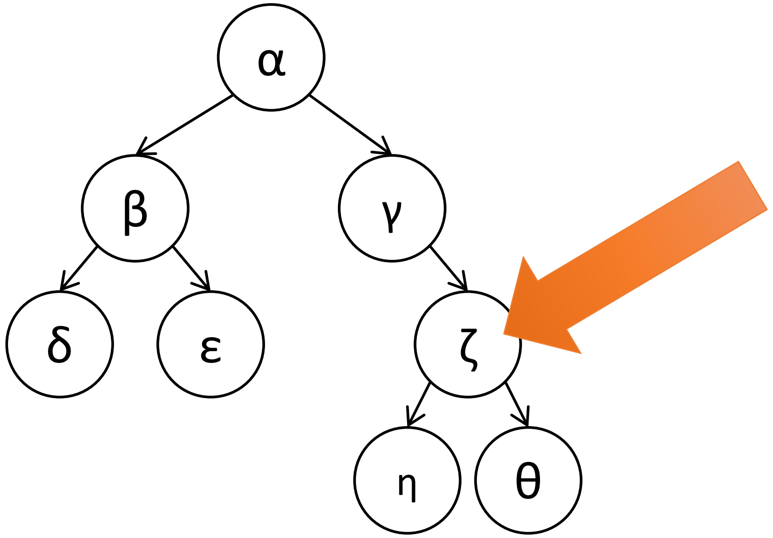
Genetic Programming

- Mutation(s)



Genetic Programming

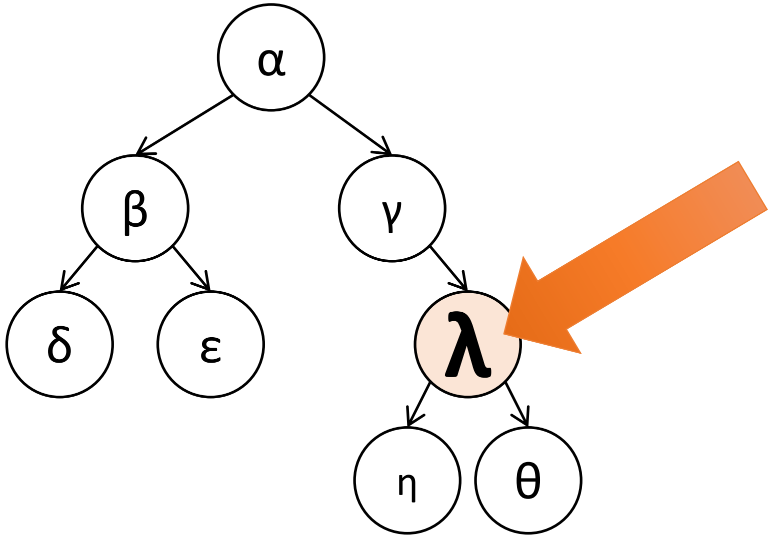
- Mutation(s)



POINT MUTATION

Genetic Programming

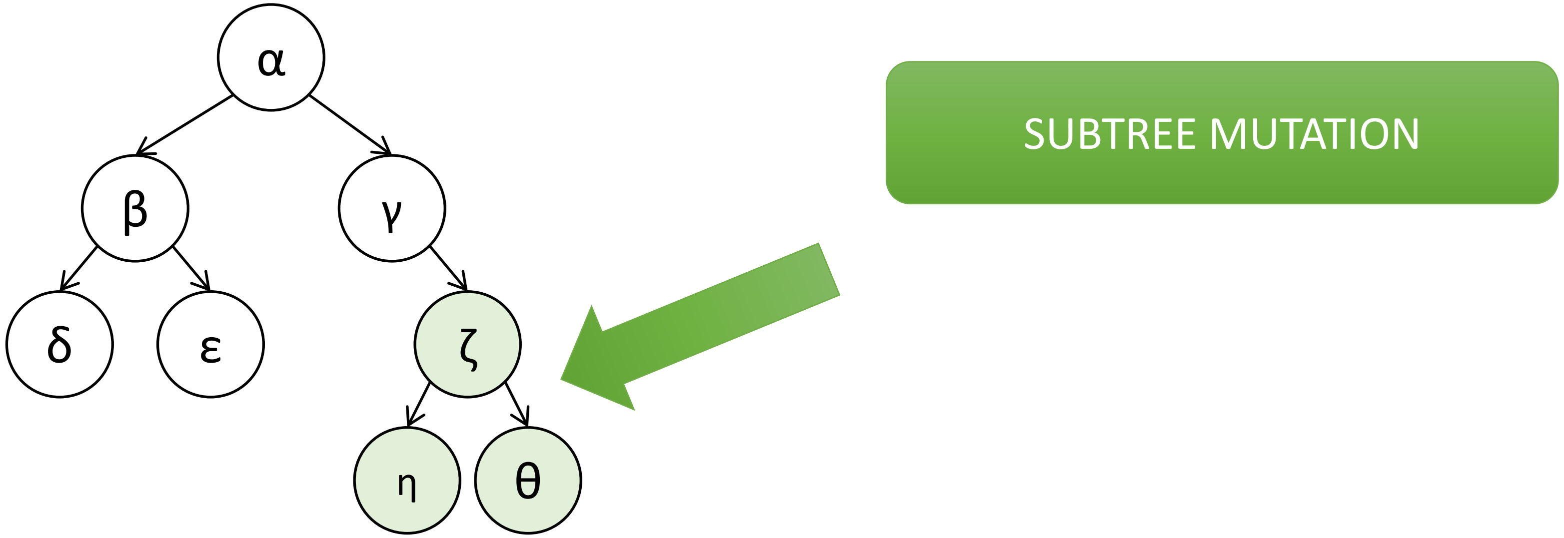
- Mutation(s)



POINT MUTATION

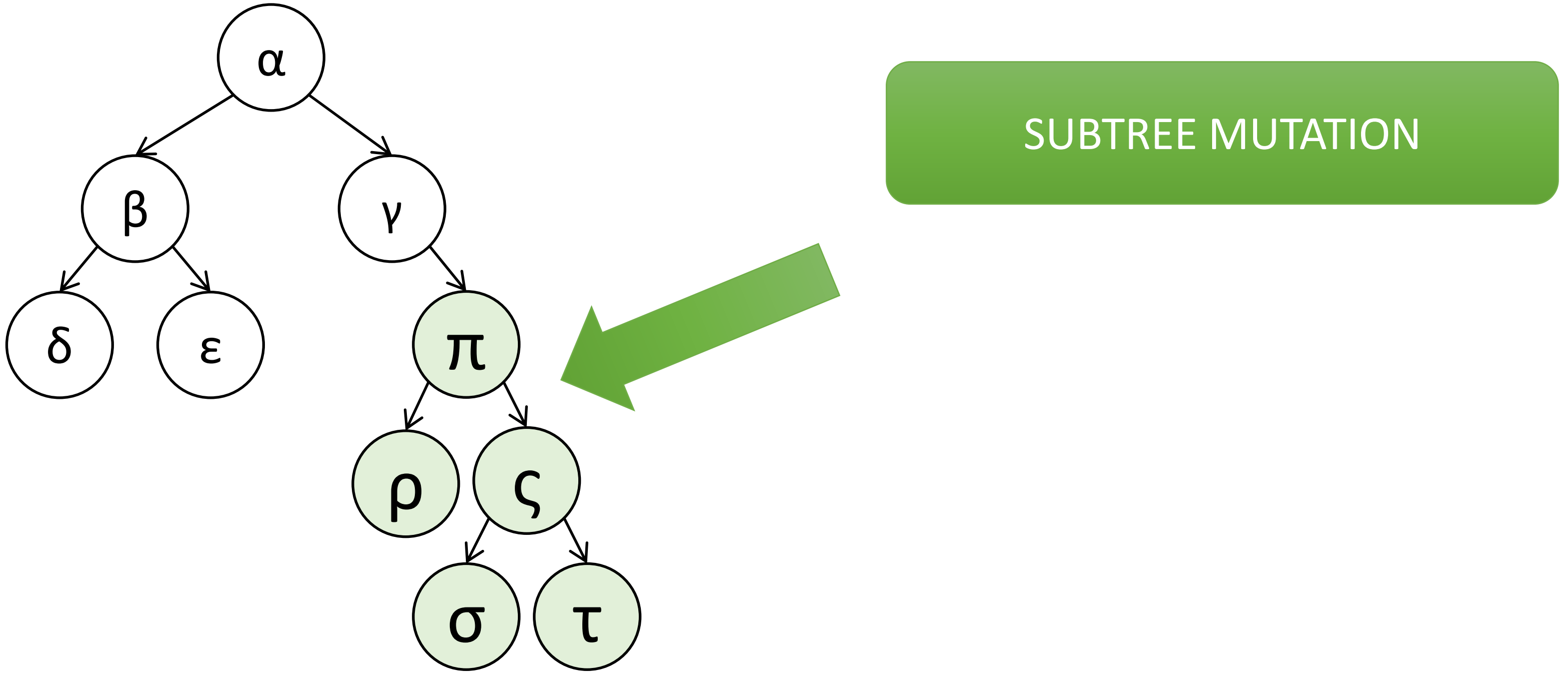
Genetic Programming

- Mutation(s)



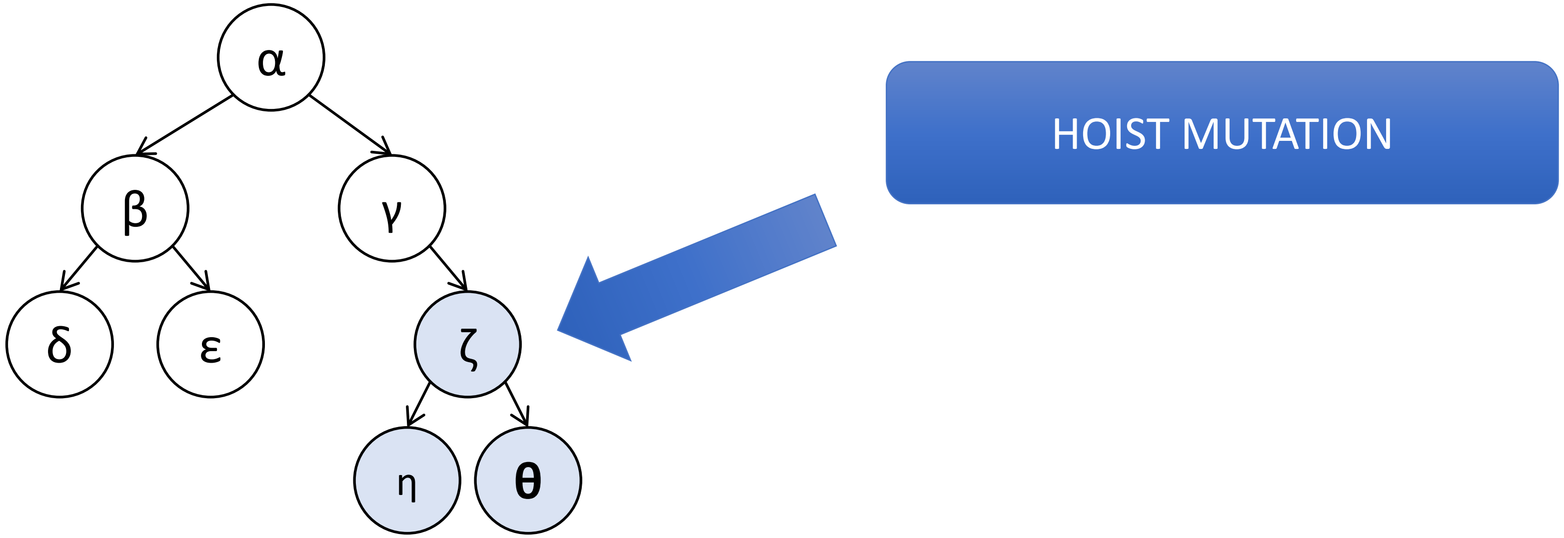
Genetic Programming

- Mutation(s)



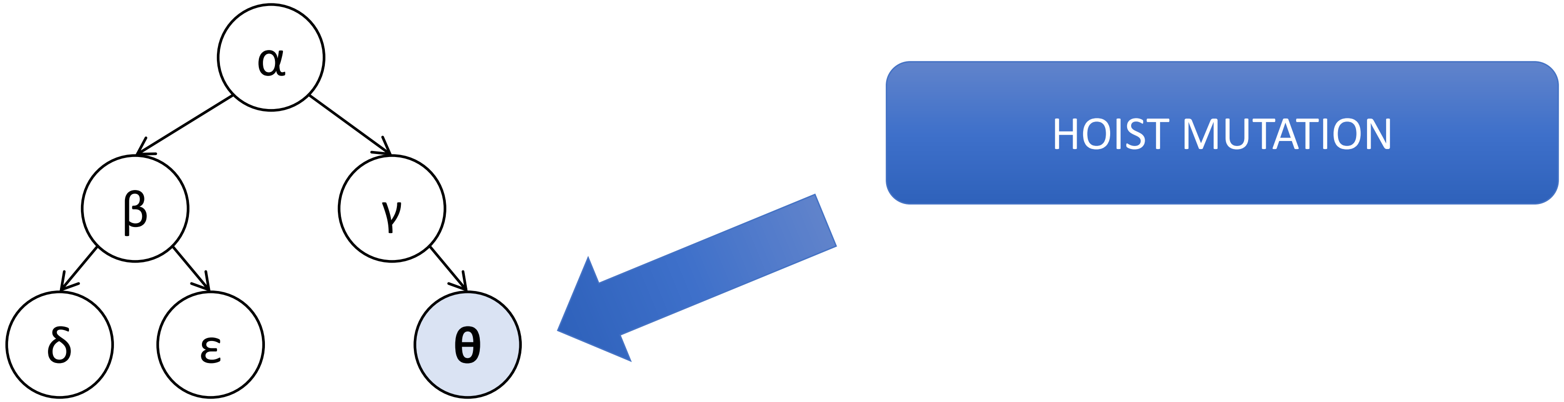
Genetic Programming

- Mutation(s)



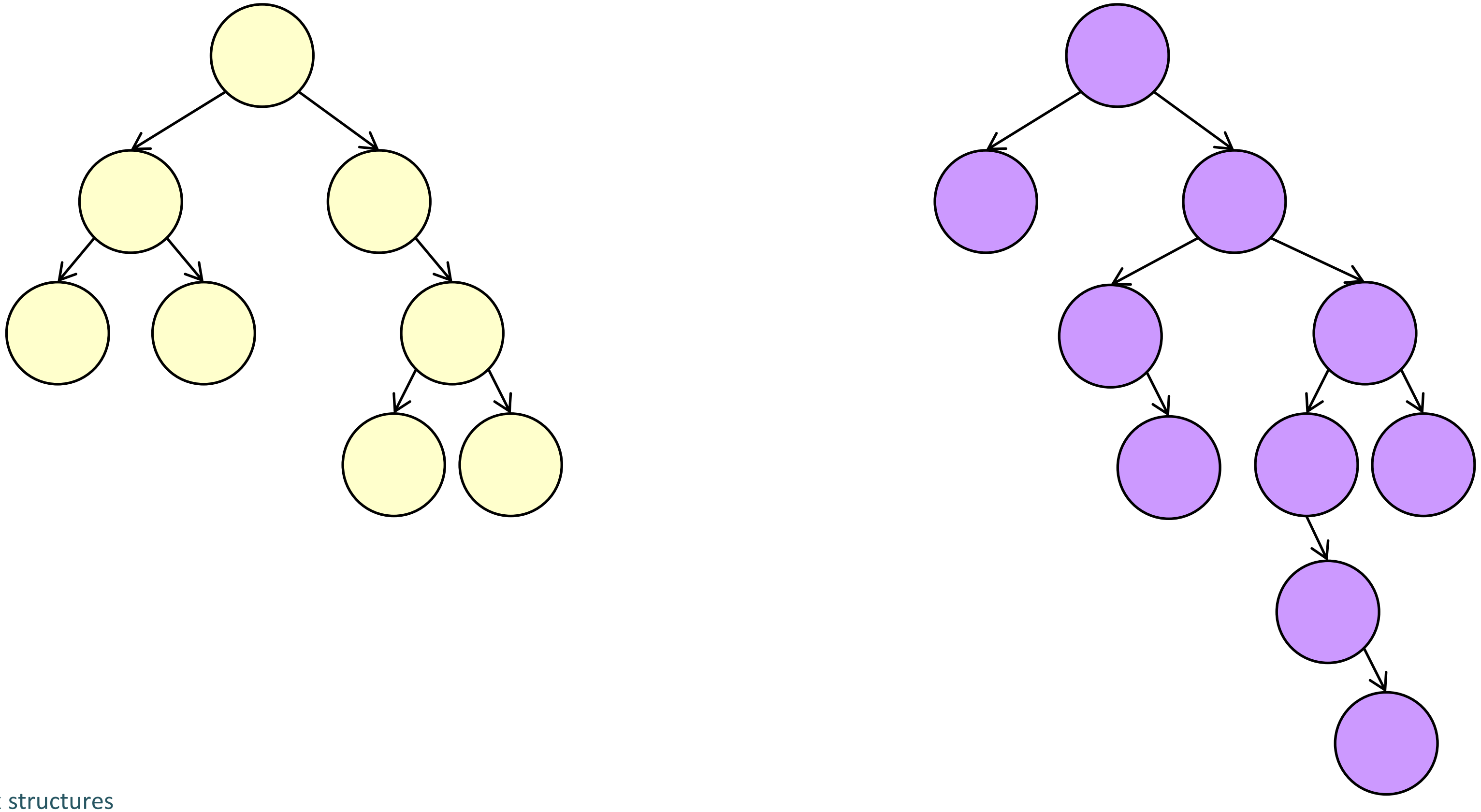
Genetic Programming

- Mutation(s)



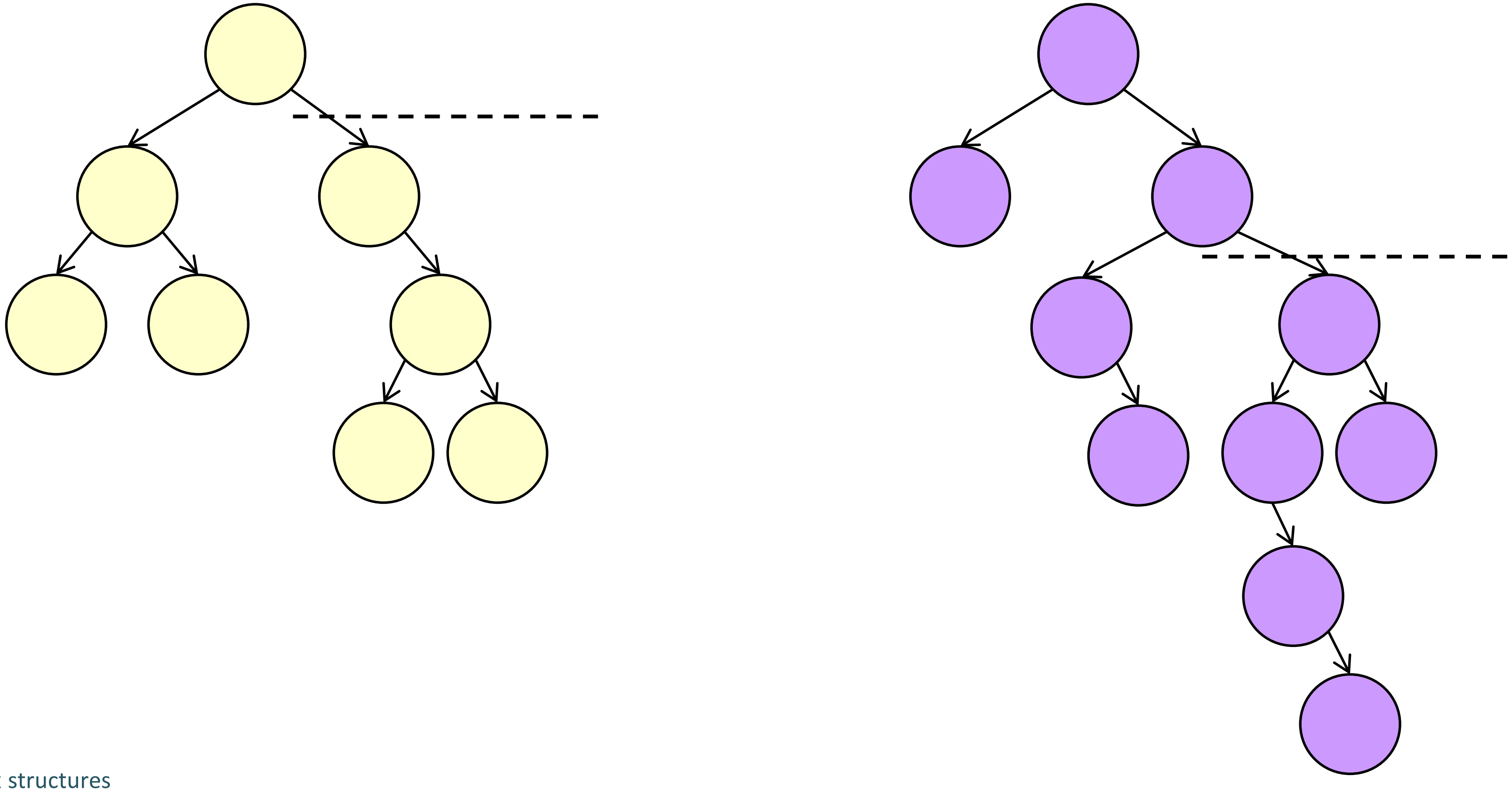
Genetic Programming

- Crossover(s)



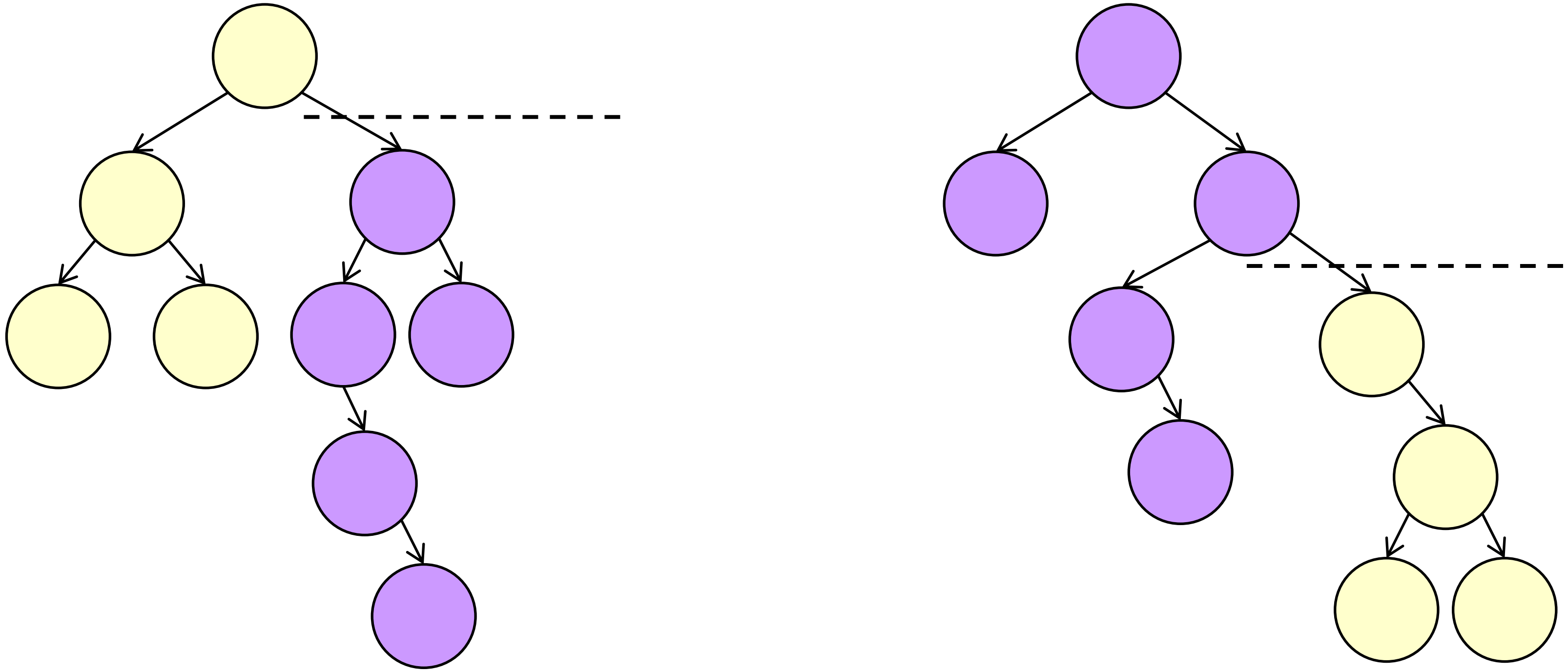
Genetic Programming

- Crossover(s)



Genetic Programming

- Crossover(s)



Genetic programming: issues

- "Bloating"
  - As the optimization proceeds, trees tend to become bigger
  - With no positive impact on objective function value
  - **Solution**: add term to objective function to penalize size
  - **Solution**: use a multi-objective approach (minimize size)

- Analyzing this in terms of ML, it could be *overfitting*
  - Bigger model, more parameters, easier to fit a function
  - For bigger models also easier to "memorize" training set

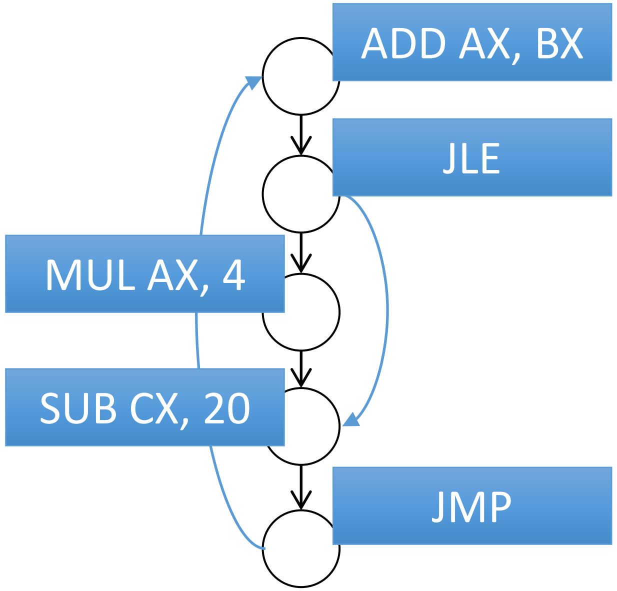
Genetic Programming

- What are we doing?
- Blending optimization and machine learning?
  - If your candidate solution is a *model*…
  - …then you are (arguably) doing *machine learning*!

- Terminology is still *in development*
- Researchers in GP getting closer to the ML community

Linear Genetic Programming

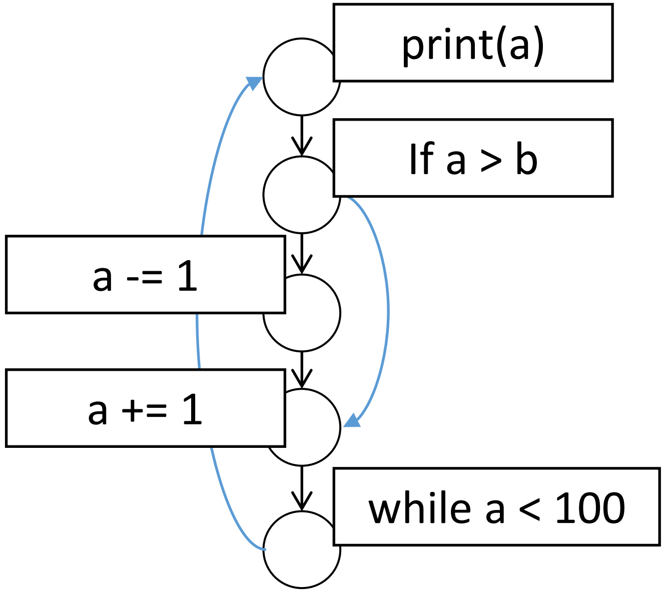
- Evolving linear graphs
- Used for evolving computer programs
- Backward/forward arcs interpreted as jumps




```
label1: ADD AX, BX
JLE label2
MUL AX, 4
label2: SUB CX, 20
JMP label1
```

Linear Genetic Programming

- Evolving linear graphs
- Used for evolving computer programs
- Backward/forward arcs interpreted as jumps




```
while a < 100 :
        print(a)
        if a > b :
                a += 1
        else :
                a -= 1
```

Grammatical Evolution

- "Grammar" in computer science
  - Defines a way to generate/validate a sequence of symbols
  - Used to check syntax coherence of programming languages
  - Evolutionary algorithm generate candidate solutions w/ grammar

$$\{\boldsymbol{start}\} \rightarrow a|b$$
$$\boldsymbol{a} \rightarrow a|b$$
$$\boldsymbol{b} \rightarrow b|\{end\}$$

ab ✓

aaaaaabbbbbb ✓

aba ✗

Example: Evolving AIs

- Real-time strategy (RTS)
  - Planet Wars (Google)
  - StarCraft
  - Student StarCraft AI Tournament

- Trade-off
  - ANNs are better
  - You can read GP trees

Example: Genetic Improvement

- Automatic correction of software bugs
- Individual: series of code modifications

- Fitness/Objective function
  - A series of test cases
  - They still have to work

Comment line 52
Swap lines 3 and 22
Change variables lines 42 and 11
...

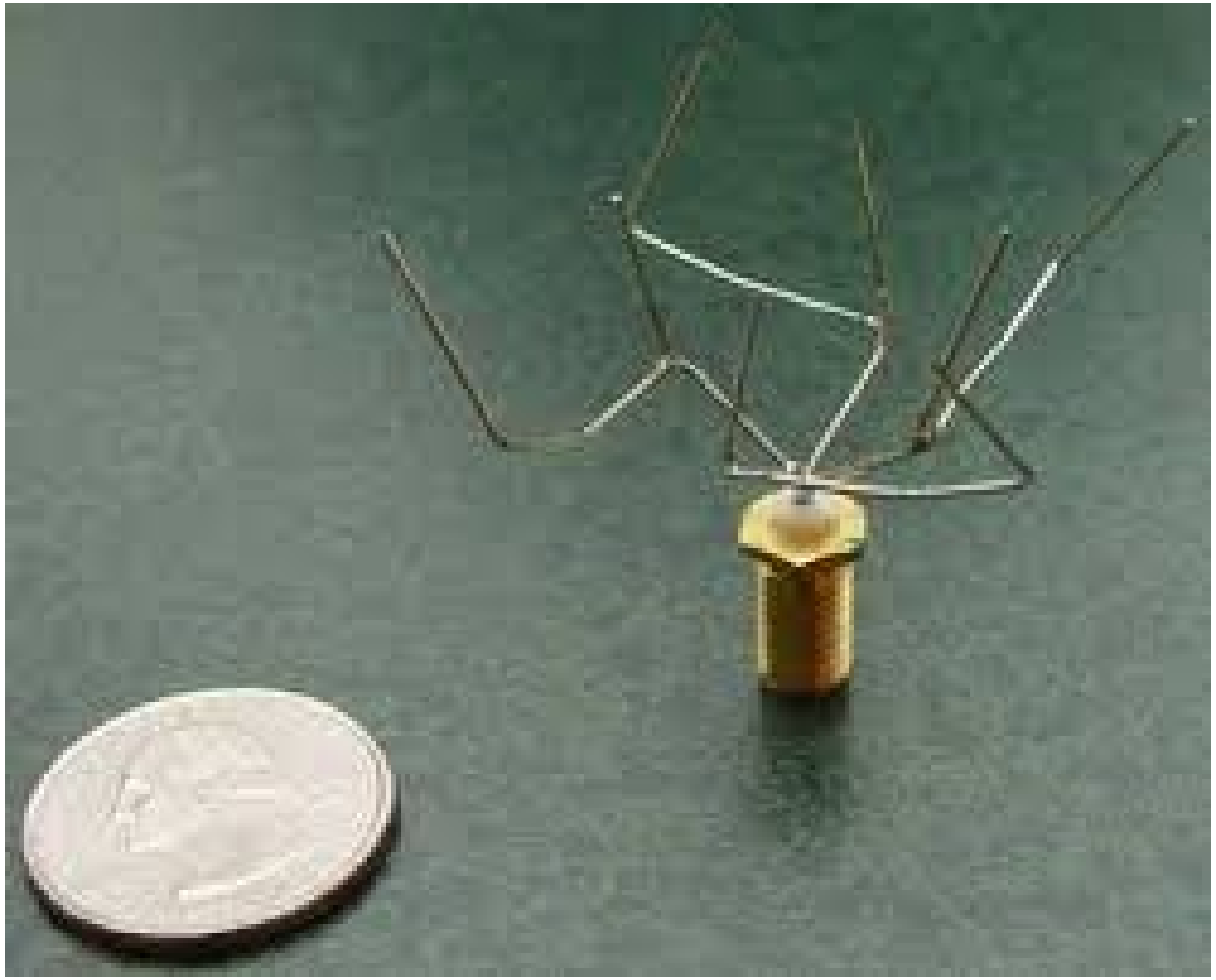
Example: Genetic Improvement

- But does it *really* work?
  - Aren't EAs introducing other bugs?
  - Aren't HUMANS introducing other bugs?
  - In the end, you just need to be *as good as* the average programmer, and you save time
  - Still experimental

Langdon, William B. **Genetically Improved Software**, 2015

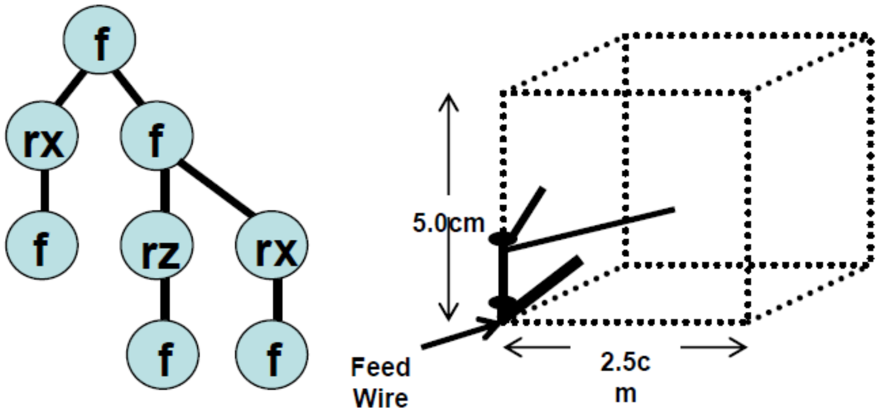
Justyna Petke and Saemundur O. Haraldsson and Mark Harman and William B. Langdon and David R. White and John R. Woodward. **Genetic Improvement of Software: a Comprehensive Survey**, 2017

Example: ?

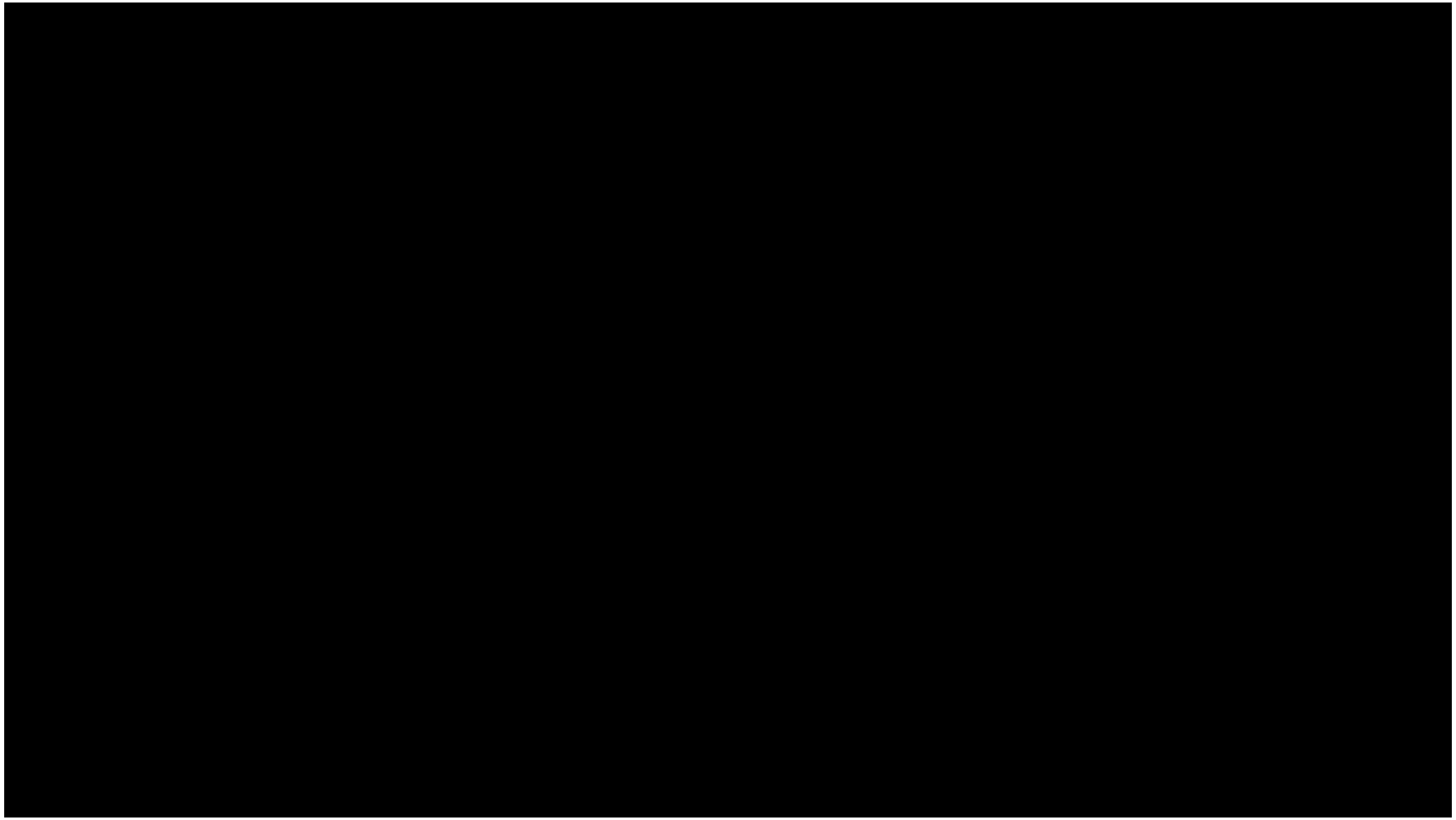


Example: Antennas

- Design of antennas for satellite ST5 (2006)
- Lots of constraints: weight, size, efficiency…
- Genome/representation is a tree
  - Forward (length, radius)
  - Rotate_x (angle)
  - Rotate_y (angle)
  - Rotate_z (angle)
- It worked!

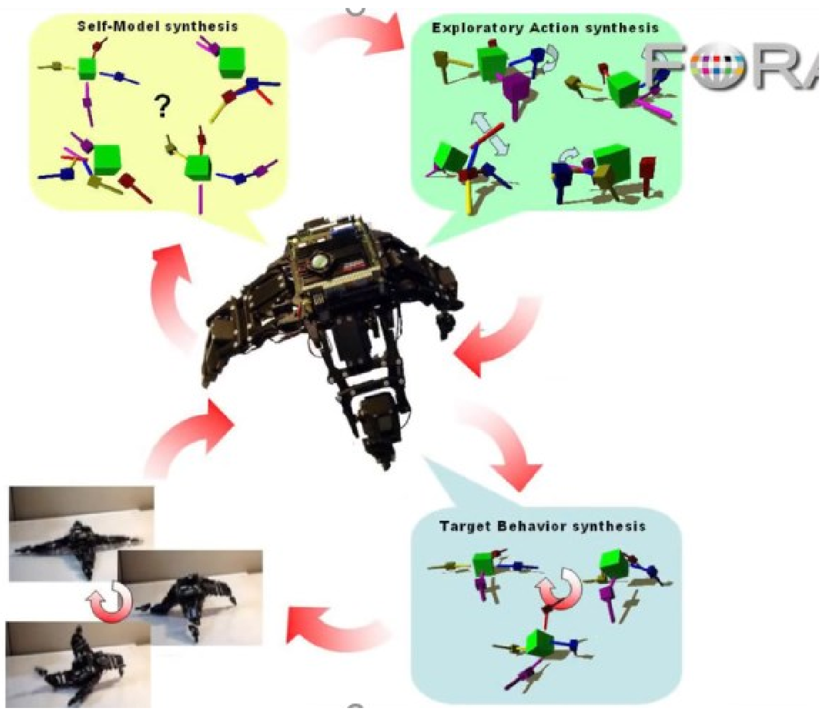


Example: Robot Movement



Hod Lipson mentioned something…

- While speaking about the candidate models for the robot
- He said “next, we are going to compare the models on the **most discriminating movement**”
- Candidate models fit the data they have already seen
- It make sense to test them on new data where they have **different predictions**
- **But how do we know what is the movement for which the models have the most different predictions?**



He did not say! But we know…

- It’s **optimization**!
  - You have a vast search space of possible movements
  - And an archive of candidate models
  - Search for the candidate movement that **maximizes** the difference in prediction between candidate models
- Competitive Co-evolutionary Algorithms
  - Lipson and Schmidt, “Coevolving Fitness Models for Accelerating Evolution and Reducing Evaluations”, 2007
  - Lipson and Schmidt, “Coevolution of Fitness Predictors”, 2008

Example: Soft Robot Movement

Evolved Electrophysiological Soft Robots

Nick Cheney[1]
Jeff Clune[2]
Hod Lipson[1]

[1] Creative Machines Lab, Cornell University
[2] Evolving AI Lab, University of Wyoming

Genetic Programming vs Generative NNs?

- Generative neural networks
  - Learn from existing training samples (e.g. images or text)
  - Able to create high-quality results, fast inference (slow training)
  - Generalize poorly “out of distribution”
- Genetic programming
  - Can create unique structures, never seen before
  - Needs an ad-hoc objective function to sample
  - Extremely slow (one whole run for one creation)
  - Human-interpretable (up to a certain limit)

Questions?

Bibliography

- Koza, *Genetic Programming*, 1992
- Garcia-Sanchez et al., *Towards Automatic StarCraft Strategy Generation Using Genetic Programming*, 2015
- Lipson & Pollack, *Automatic design and manufacture of robotic lifeforms*, 2000