

Evolutionary Machine Learning - The Next Frontier in Artificial Intelligence?

Wolfgang Banzhaf
John R. Koza Chair in Genetic Programming
Computer Science & Engineering and BEACON Center
Michigan State University, East Lansing, USA





“The Bitter Lesson”

The Bitter Lesson

Rich Sutton

March 13, 2019

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. The ultimate reason for this is Moore's law, or rather its generalization of continued exponentially falling cost per unit of computation. Most AI research has been conducted as if the computation available to the agent were constant (in which case leveraging human knowledge would be one of the only ways to improve performance) but, over a slightly longer time than a typical research project, massively more computation inevitably becomes available. Seeking an improvement that makes a difference in the shorter term, researchers seek to leverage their human knowledge of the domain, but the only thing that matters in the long run is the leveraging of computation. These two need not run counter to each other, but in practice they tend to. Time spent on one is time not spent on the other. There are psychological commitments to investment in one approach or the other. And the human-knowledge approach tends to complicate methods in ways that make them less suited to taking advantage of general methods leveraging computation. There were many examples of AI researchers' belated learning of this bitter lesson, and it is instructive to review some of the most prominent.

Richard Sutton, 2019



“The Bitter Lesson”

- The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin.

Richard Sutton, 2019



“The Bitter Lesson”

- The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin.
- We want AI agents that can discover like we can, not which contain what we have discovered. Building in our discoveries only makes it harder to see how the discovering process can be done.

Richard Sutton, 2019



The Birth of Machine Learning





The Birth of Machine Learning

- Machine Learning is the branch of A.I. that concerns itself with building models of systems



The Birth of Machine Learning

- Machine Learning is the branch of A.I. that concerns itself with building models of systems
- Arthur Samuel coined the term in connection with his Checkers playing program and reports in 1959:



The Birth of Machine Learning

- Machine Learning is the branch of A.I. that concerns itself with building models of systems
- Arthur Samuel coined the term in connection with his Checkers playing program and reports in 1959:

“Two machine-learning procedures have been investigated in some detail using the game of checkers. Enough work has been done to verify the fact that a computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program. Furthermore, it can learn to do this in a remarkably short period of time (8 or 10 hours of machine-playing time) when given only the rules of the game, a sense of direction, and a redundant and incomplete list of parameters which are thought to have something to do with the game, but whose correct signs and relative weights are unknown and unspecified. The principles of machine learning verified by these experiments are, of course, applicable to many other situations.”



Major Reasons for the Success of ML





Major Reasons for the Success of ML

- The explosive development of hardware



Major Reasons for the Success of ML

- The explosive development of hardware
- The availability of massive amounts of data



Major Reasons for the Success of ML

- The explosive development of hardware
- The availability of massive amounts of data
- The grounding of ML in mathematical/statistical thought



Major Reasons for the Success of ML

- The explosive development of hardware
- The availability of massive amounts of data
- The grounding of ML in mathematical/statistical thought
- The ability to find solvable ML problems



The Core of Intelligence





The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?



The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?
- What IS intelligence?



The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?
- What IS intelligence?
- The core of intelligence is the *ability of a system to model other systems* (their behavior, possibly their inner workings)



The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?
- What IS intelligence?
- The core of intelligence is the *ability of a system to model other systems* (their behavior, possibly their inner workings)
- Modeling other systems entails



The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?
- What IS intelligence?
- The core of intelligence is the *ability of a system to model other systems* (their behavior, possibly their inner workings)
- Modeling other systems entails
 - the ability to *remember and understand or explain* the behavior of other systems



The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?
- What IS intelligence?
- The core of intelligence is the *ability of a system to model other systems* (their behavior, possibly their inner workings)
- Modeling other systems entails
 - the ability to *remember and understand or explain* the behavior of other systems
 - the ability to *predict* the behavior of other systems



The Core of Intelligence

- What do we want to achieve when creating or constructing “Artificial Intelligence”?
- What IS intelligence?
- The core of intelligence is the *ability of a system to model other systems* (their behavior, possibly their inner workings)
- Modeling other systems entails
 - the ability to *remember and understand or explain* the behavior of other systems
 - the ability to *predict* the behavior of other systems
- Modeling requires a way to *represent other systems and to manipulate such representations*



How about Evolution?





How about Evolution?

- What has this to do with evolution?



How about Evolution?

- What has this to do with evolution?
- Evolution is the *process* by which Nature has created systems of nearly arbitrary *complexity* (including intelligent beings)



How about Evolution?

- What has this to do with evolution?
- Evolution is the *process* by which Nature has created systems of nearly arbitrary *complexity* (including intelligent beings)
- How does this process work?



How about Evolution?

- What has this to do with evolution?
- Evolution is the *process* by which Nature has created systems of nearly arbitrary *complexity* (including intelligent beings)
- How does this process work?
 - It acts among living organisms that multiply and diversify via *inheritance* and *variation*



How about Evolution?

- What has this to do with evolution?
- Evolution is the *process* by which Nature has created systems of nearly arbitrary *complexity* (including intelligent beings)
- How does this process work?
 - It acts among living organisms that multiply and diversify via *inheritance* and *variation*
 - It sets up a *competition* among organisms (for food, shelter, reproduction, etc.) - the *struggle for survival*



How about Evolution?

- What has this to do with evolution?
- Evolution is the *process* by which Nature has created systems of nearly arbitrary *complexity* (including intelligent beings)
- How does this process work?
 - It acts among living organisms that multiply and diversify via *inheritance* and *variation*
 - It sets up a *competition* among organisms (for food, shelter, reproduction, etc.) - the *struggle for survival*
- To be able to survive requires to *model* the behavior of the environment, including other organisms



How about Evolution?

- What has this to do with evolution?
- Evolution is the *process* by which Nature has created systems of nearly arbitrary *complexity* (including intelligent beings)
- How does this process work?
 - It acts among living organisms that multiply and diversify via *inheritance* and *variation*
 - It sets up a *competition* among organisms (for food, shelter, reproduction, etc.) - the *struggle for survival*
- To be able to survive requires to *model* the behavior of the environment, including other organisms
- Modeling requires a way to *represent* other systems and to *manipulate* such representations



How about Evolution?



How about Evolution?

- Nature has created intelligence by using an evolutionary process



How about Evolution?

- Nature has created intelligence by using an evolutionary process
- If that was the way to do it in Nature we should think about a similar process if we want to create “Artificial Intelligence”



How about Evolution?

- Nature has created intelligence by using an evolutionary process
- If that was the way to do it in Nature we should think about a similar process if we want to create “Artificial Intelligence”

How about Evolution?

- Nature has created intelligence by using an evolutionary process
- If that was the way to do it in Nature we should think about a similar process if we want to create “Artificial Intelligence”
- *The field of evolutionary machine learning (EML) concerns itself with the application of the evolutionary process to machine learning problems and methods and the application of ML to evolutionary computation methods.*

Living Systems have
inspired models of
computation since the
beginning of Computing

Bio-inspiration

Bio-inspiration

- Alan Turing: Intelligence, Evolution



Bio-inspiration

- Alan Turing: Intelligence, Evolution



- John von Neumann: SR automata, CAs, Artificial Life



Bio-inspiration

- Alan Turing: Intelligence, Evolution



- John von Neumann: SR automata, CAs, Artificial Life



- Frank Rosenblatt: Perceptron



Bio-inspiration

- Alan Turing: Intelligence, Evolution



- John von Neumann: SR automata, CAs, Artificial Life



- Frank Rosenblatt: Perceptron



- John Holland: Adaptive Systems, GA



Bio-inspiration

- Alan Turing: Intelligence, Evolution



- John von Neumann: SR automata, CAs, Artificial Life



- Frank Rosenblatt: Perceptron



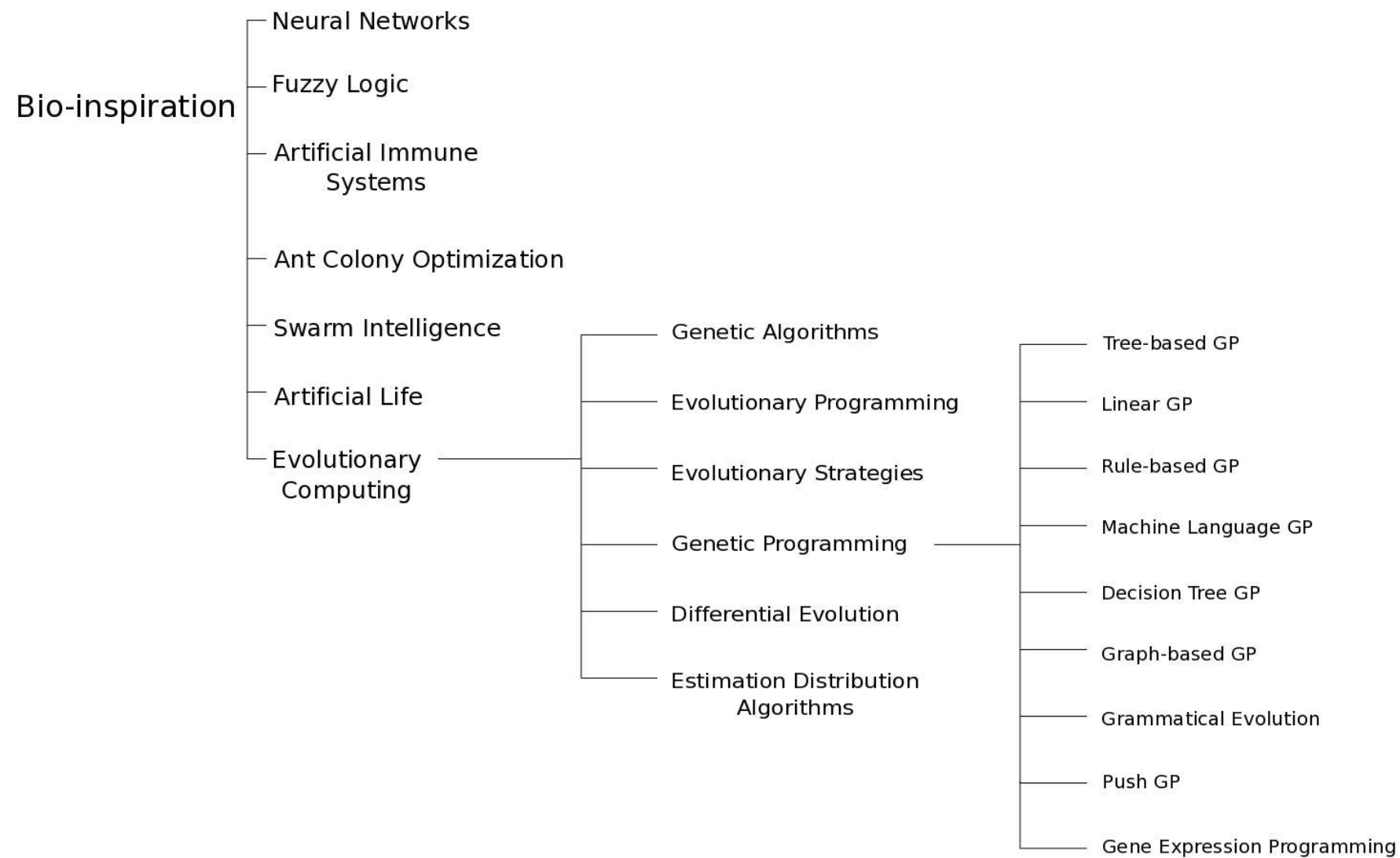
- John Holland: Adaptive Systems, GA



- Ingo Rechenberg: Evolutionary optimization of mechanical systems



Tree of Bio-inspired Computing



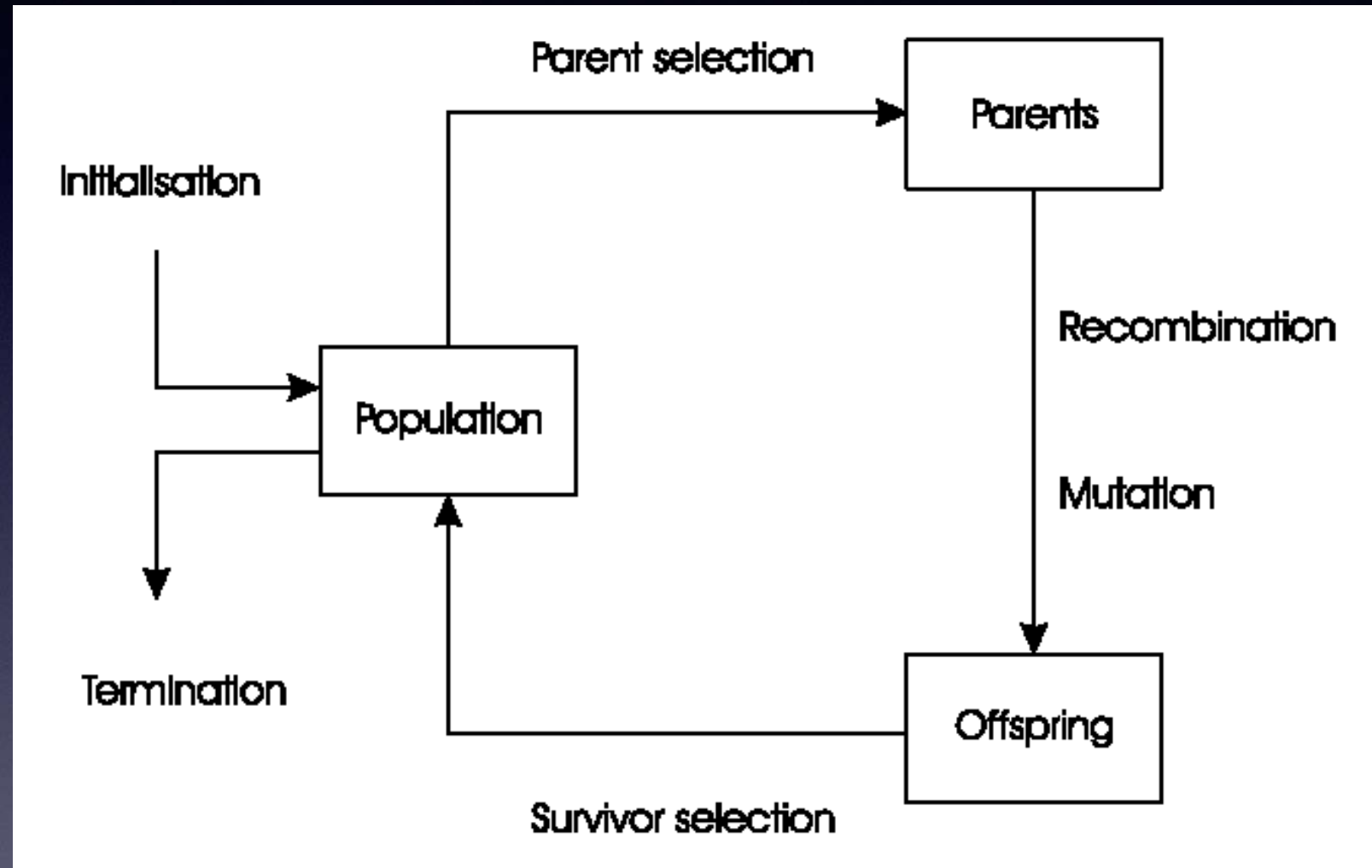
1950-1990

1960-1995

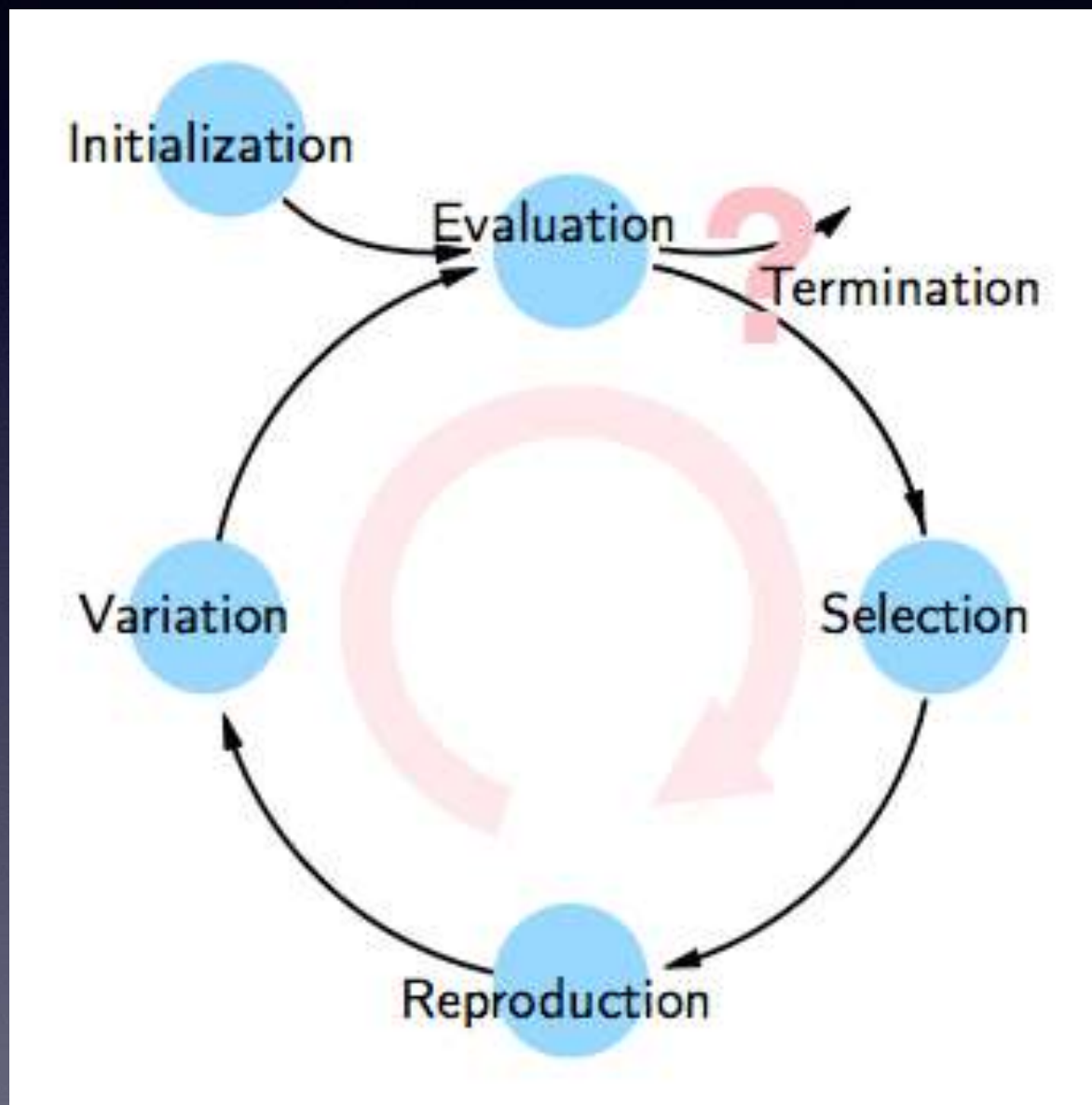
1990-2000 & later

Evolutionary Algorithms and Genetic Programming

General Scheme of EAs



The Feedback Loop of Evolutionary Computation



Loop
Feedback cycle

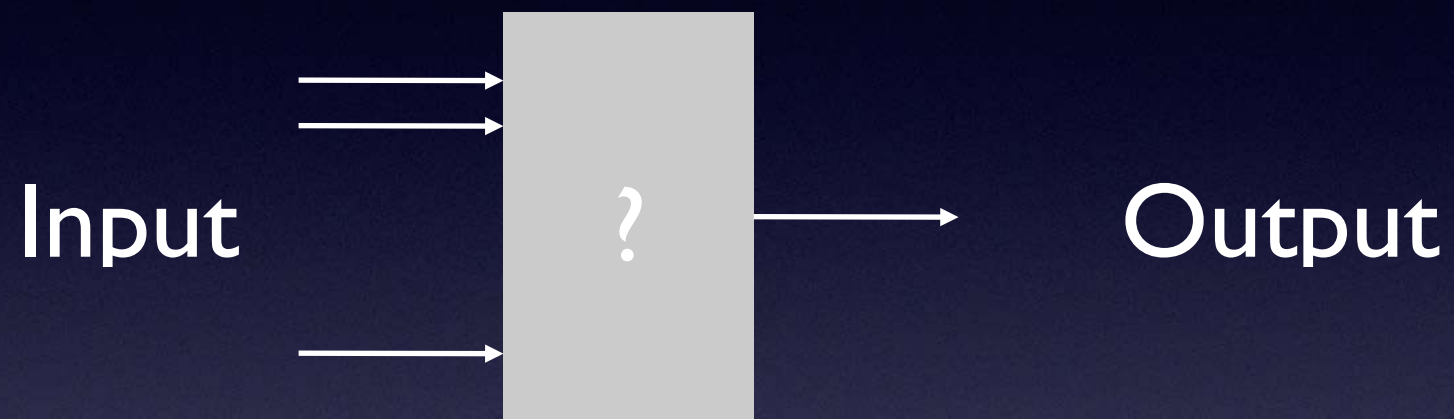
Positive = destabilizing
-> growth

Negative = stabilizing
-> disappearance

Elements of a system capable of GP

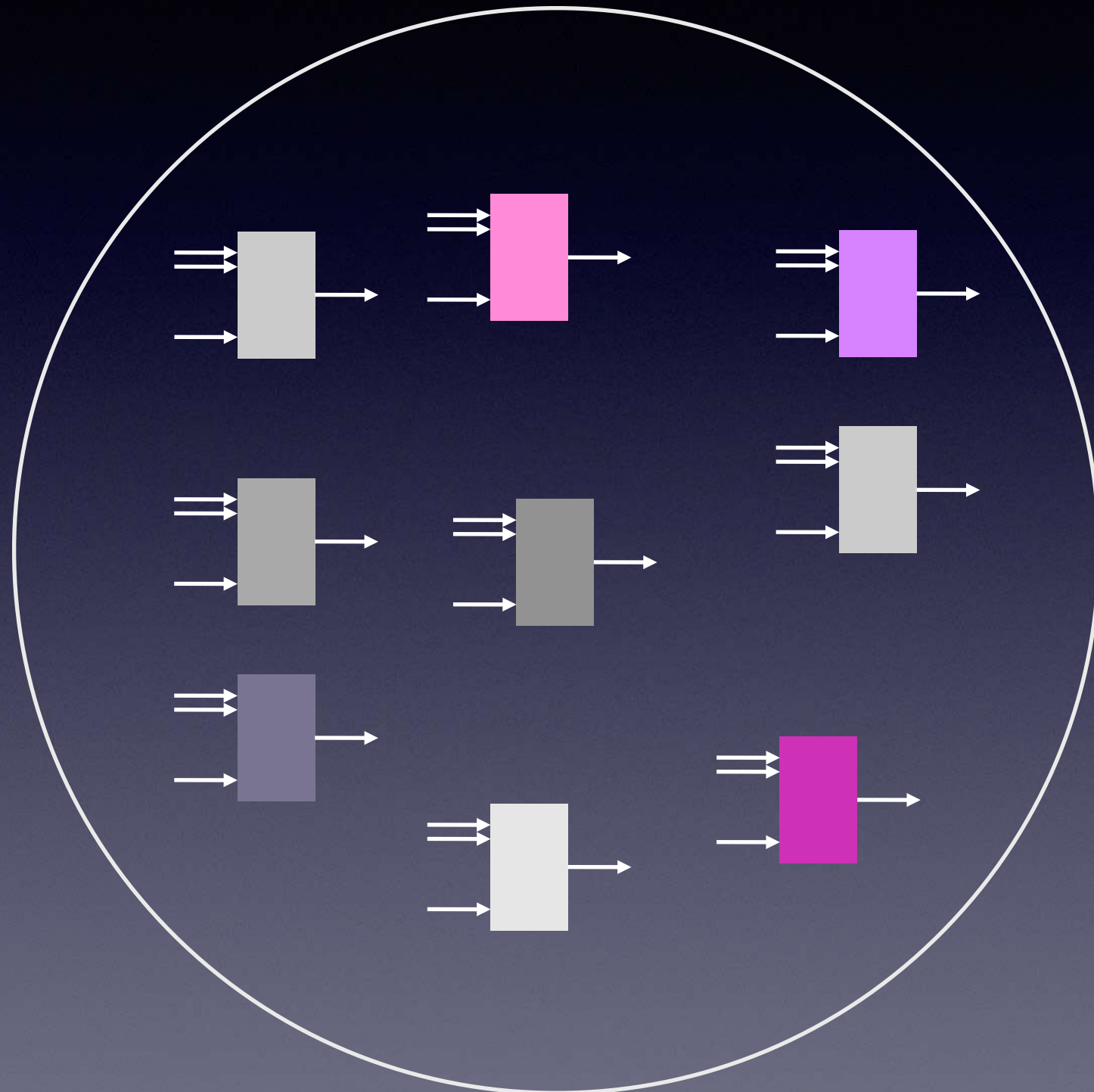
1. Population of individuals representing programs/algorithms
2. Individuals have genes specifying behavioral elements
3. Selection according to fitness cases characterizing input/output of individuals
4. Mutation and recombination for new variants

A GP Individual

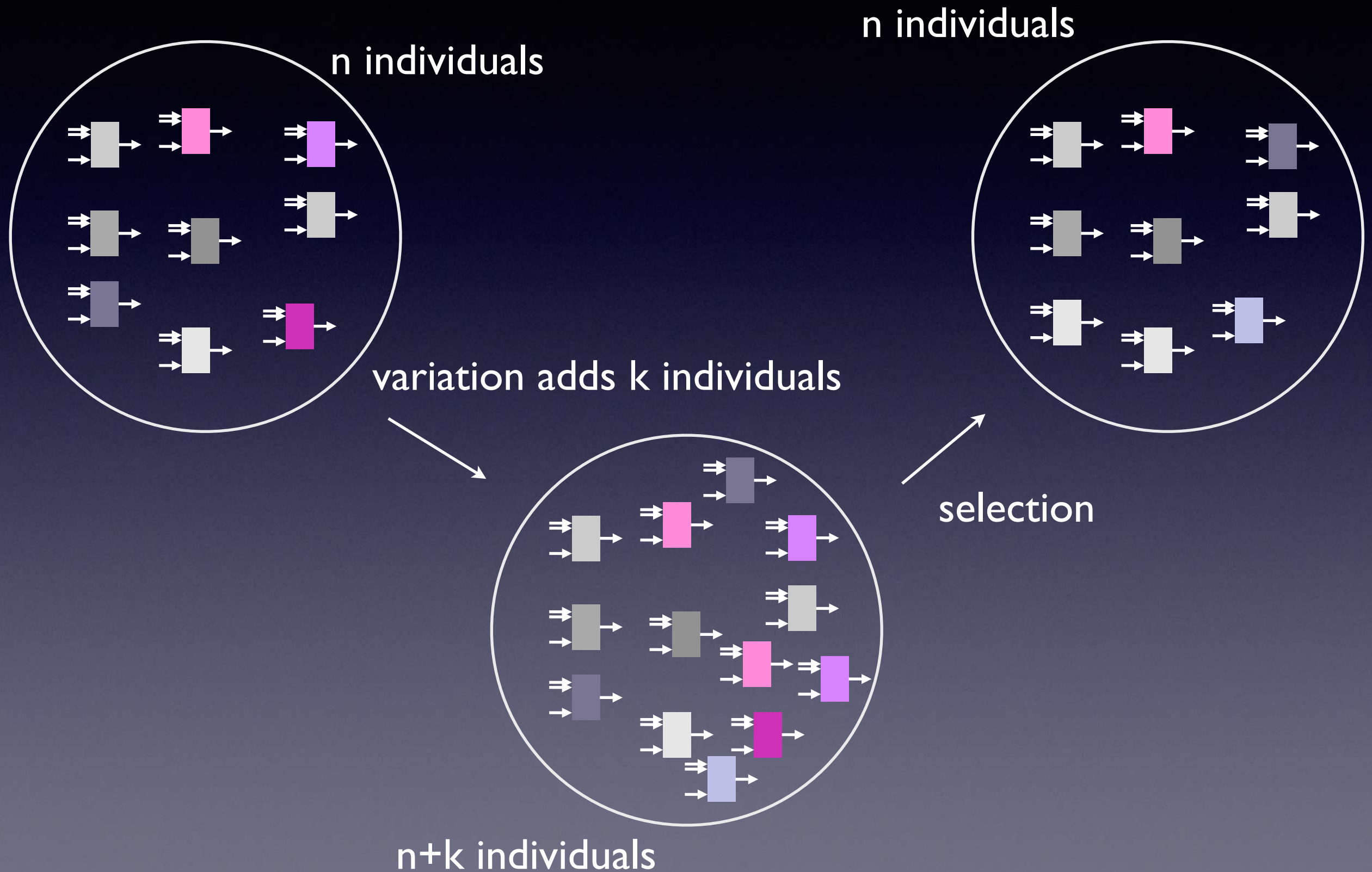


- “?” is a program (or algorithmic model)
- Define a quality of solutions (e.g. error measure like actual output vs. target output)
- Generate different programs that solve the problem more or less accurately
- Test programs on „fitness cases“, i.e. a set of input/output pairs
- Improve solutions by trying variants

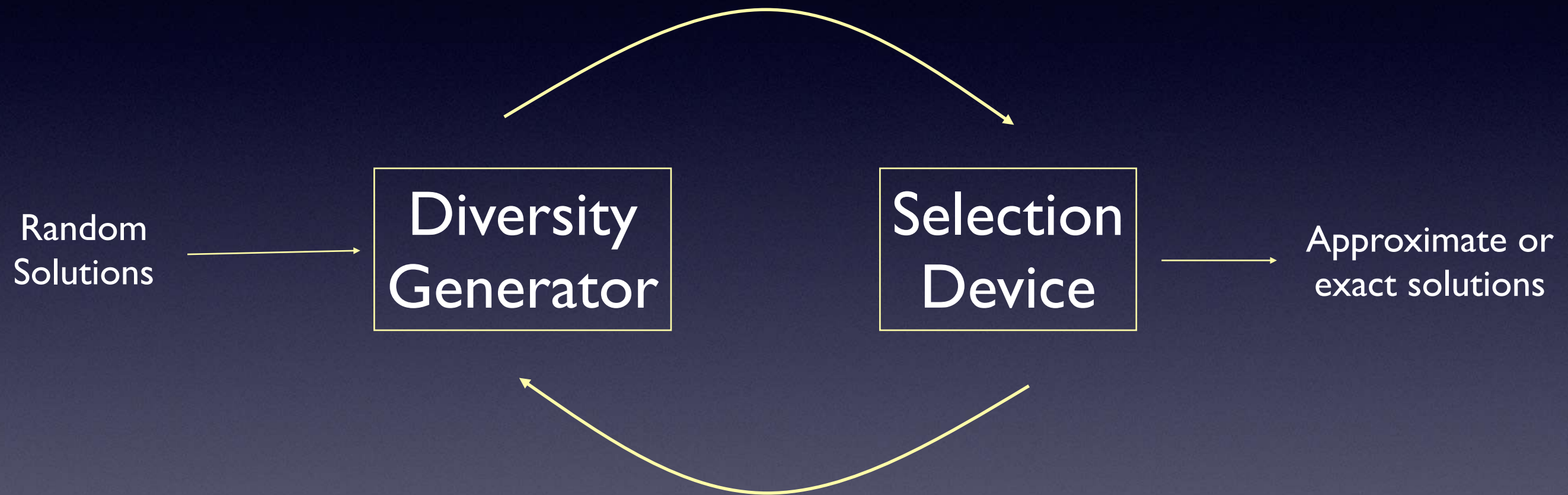
A GP Population



A GP Selection Cycle

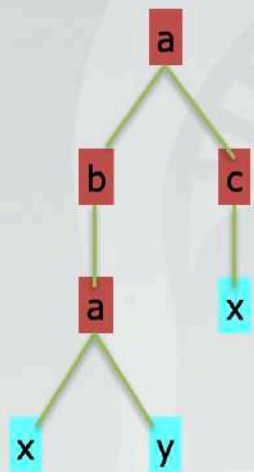


Cumulative Selection

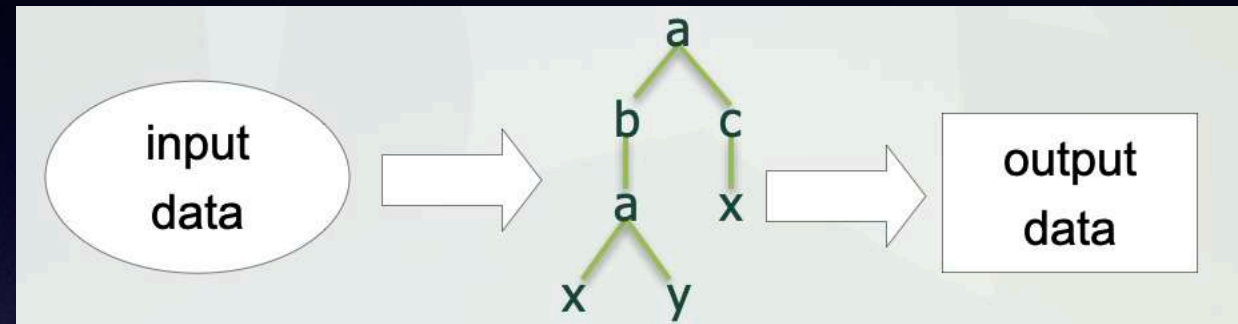


Tree GP

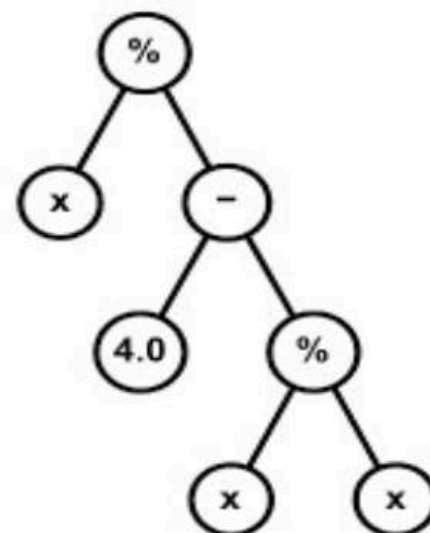
Functions and Terminals



functions: {a, b, c}
terminals: {x, y}



Expression Trees



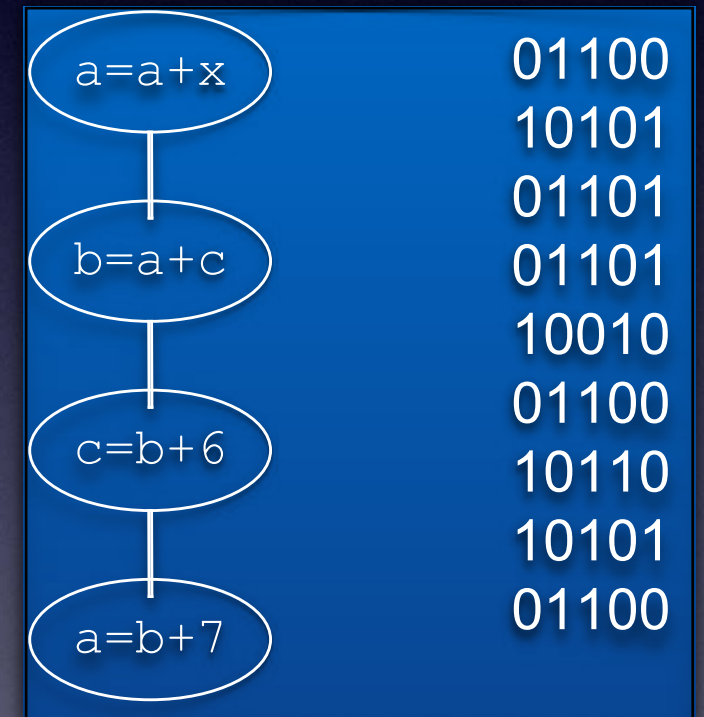
How you read it?

$$x / (4 - x / x) = x / 3$$

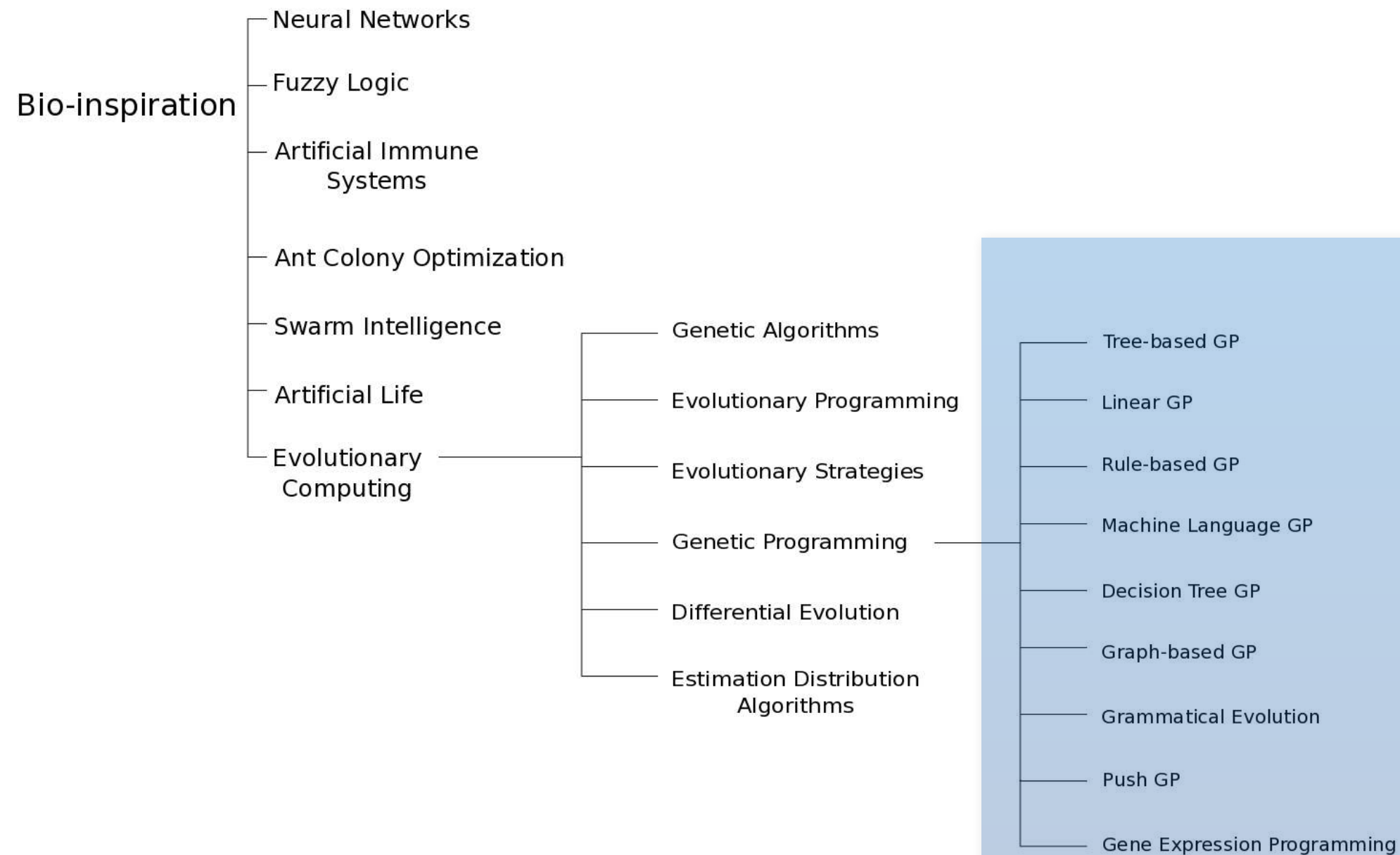
?: Protected division

Linear GP

- Follows principles of imperative languages
- Based on instruction sequences: Each instruction is a gene
- Each instruction contains the elements of operator and operand(s) and an assignment
- Bit sequences code for operators (op-code) and operands (register addresses)
- Close relationship between machine code and interpretation



Many representations in use for GP





The Advantages of Evolutionary Principles



The Advantages of Evolutionary Principles

- If intelligence is fundamentally about prediction in a world with many uncertainties - what better way to do it than with *populations of models* that either cooperate (ensembles) or compete (as in EC).



The Advantages of Evolutionary Principles

- If intelligence is fundamentally about prediction in a world with many uncertainties - what better way to do it than with *populations of models* that either cooperate (ensembles) or compete (as in EC).
- Intelligence is about problem solving in creative ways. Evolution has stochasticity as a creativity engine, which works with cumulative selection to generate new, *surprising solutions by emergence*.



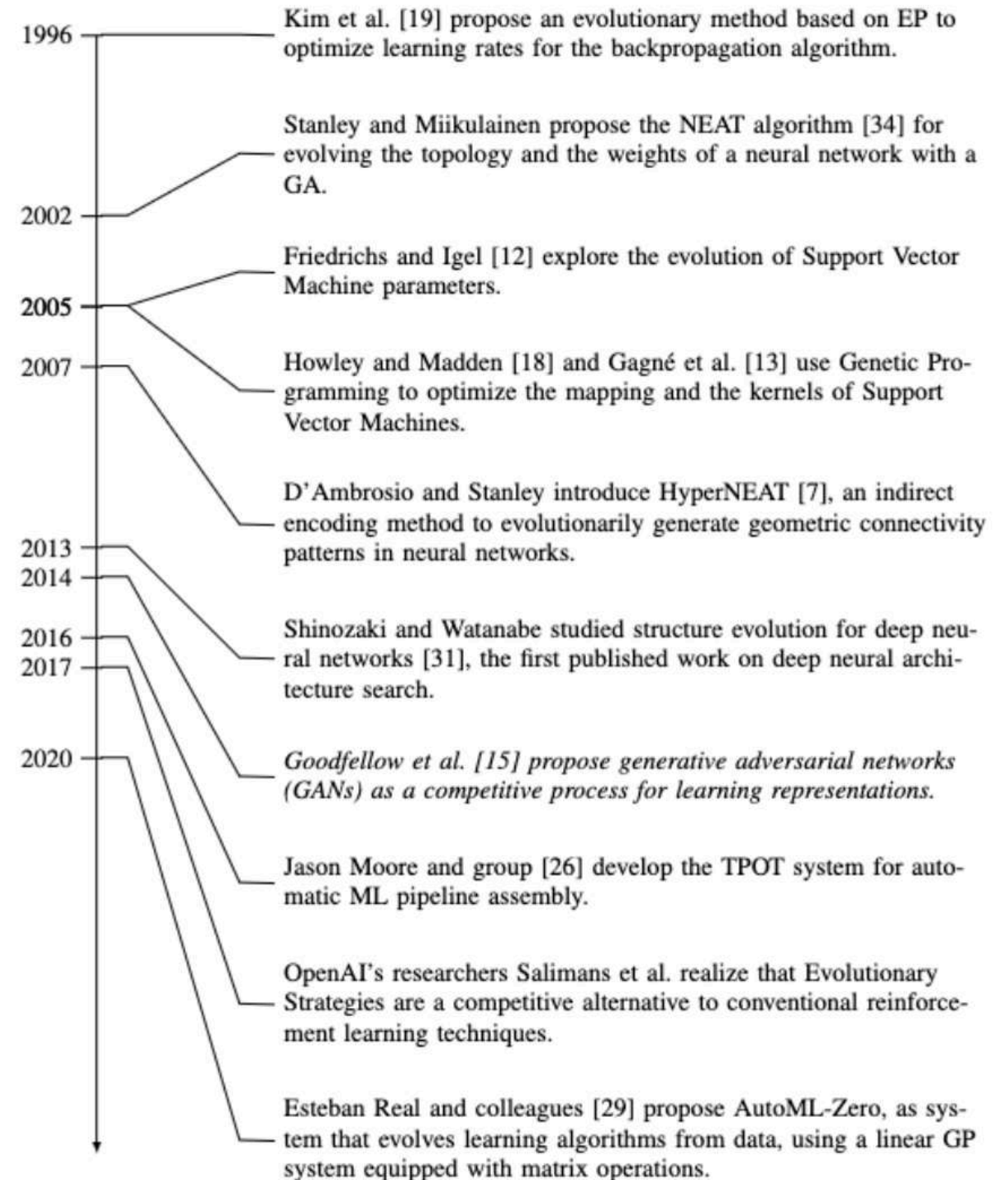
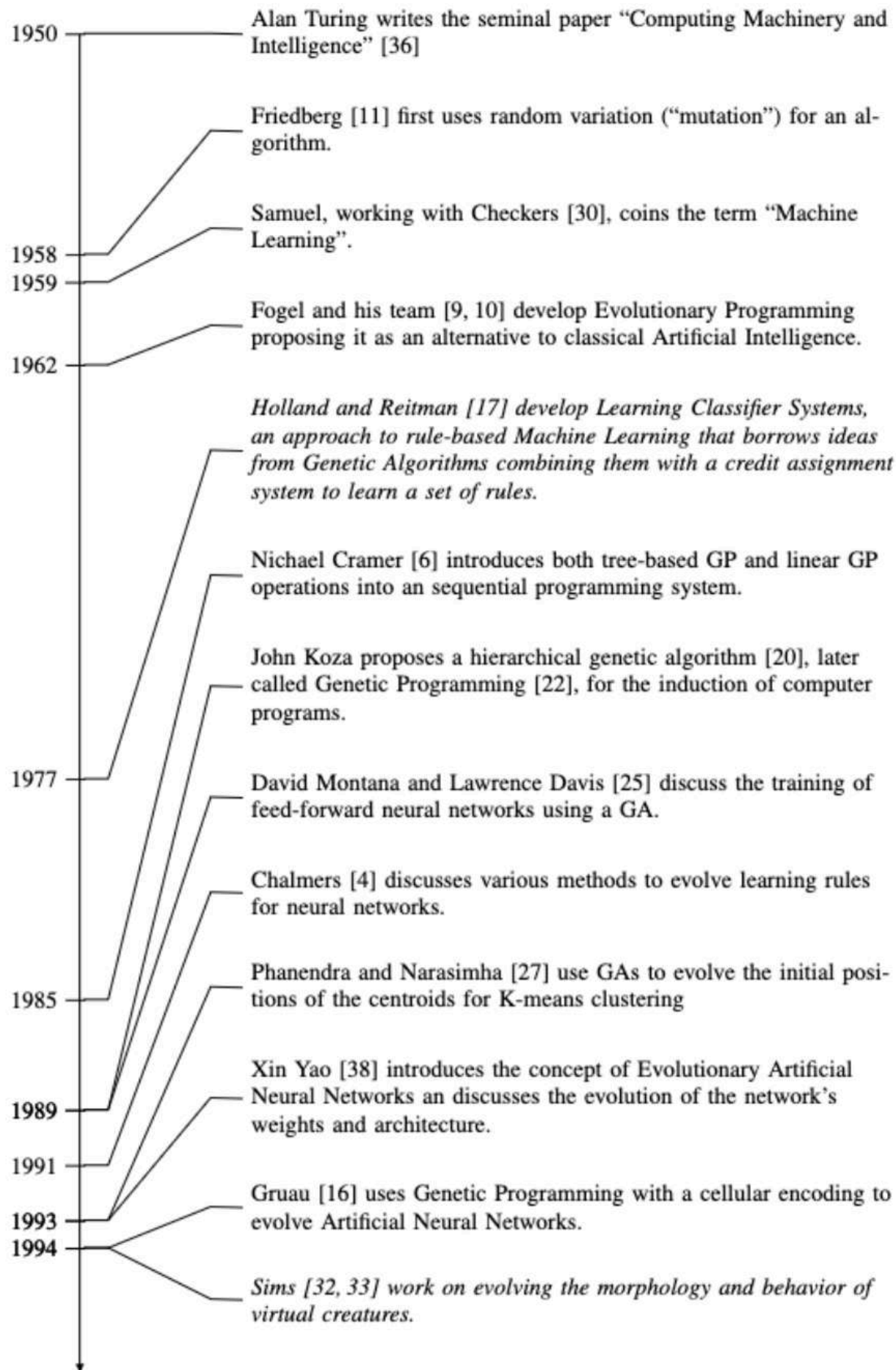
The Advantages of Evolutionary Principles

- If intelligence is fundamentally about prediction in a world with many uncertainties - what better way to do it than with *populations of models* that either cooperate (ensembles) or compete (as in EC).
- Intelligence is about problem solving in creative ways. Evolution has stochasticity as a creativity engine, which works with cumulative selection to generate new, *surprising solutions by emergence*.
- The populations of evolutionary algorithms are evolving jointly., exchanging information. There is no assumption about independence between these different models (or statistical linearity).



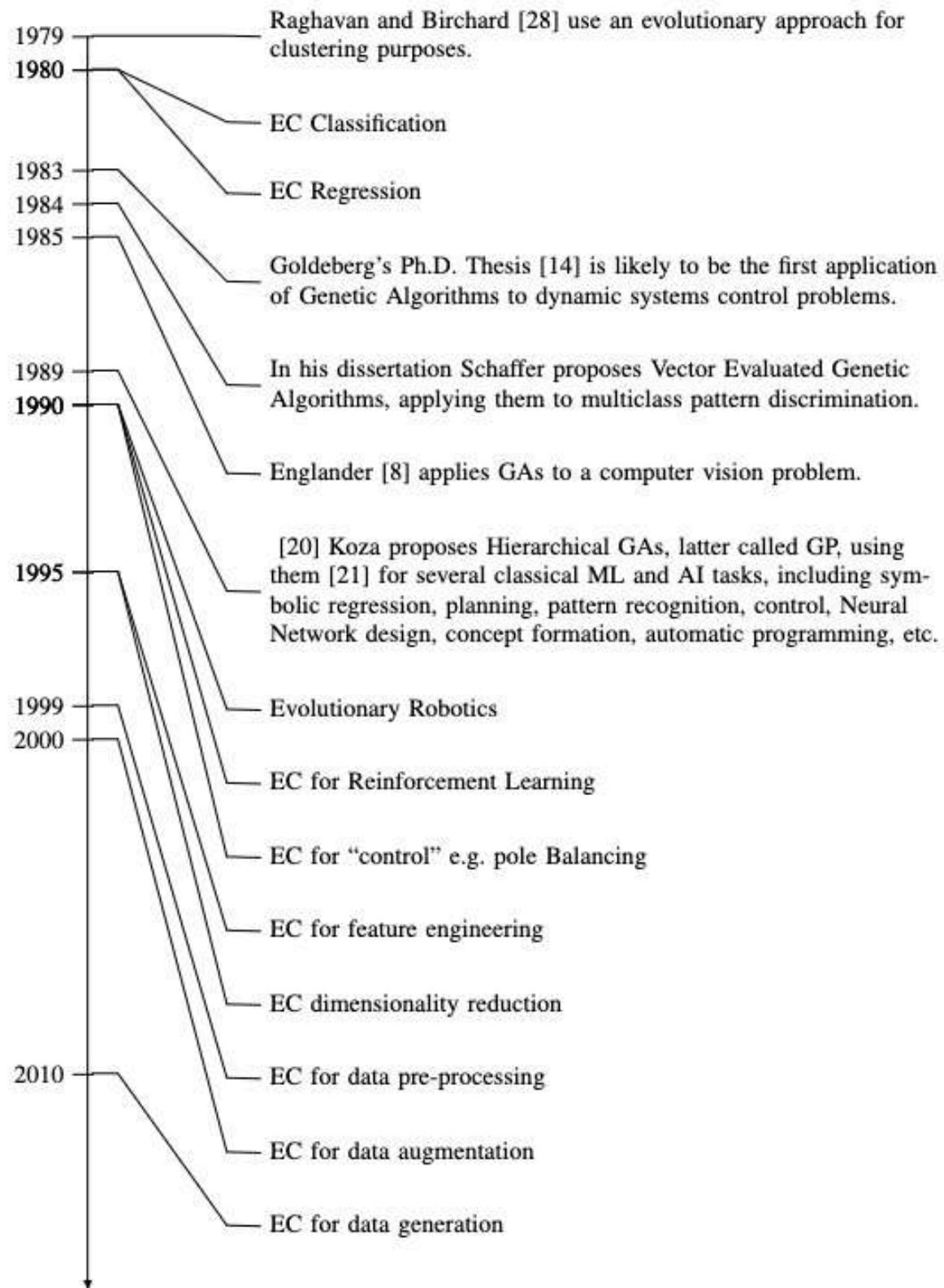
History of EML

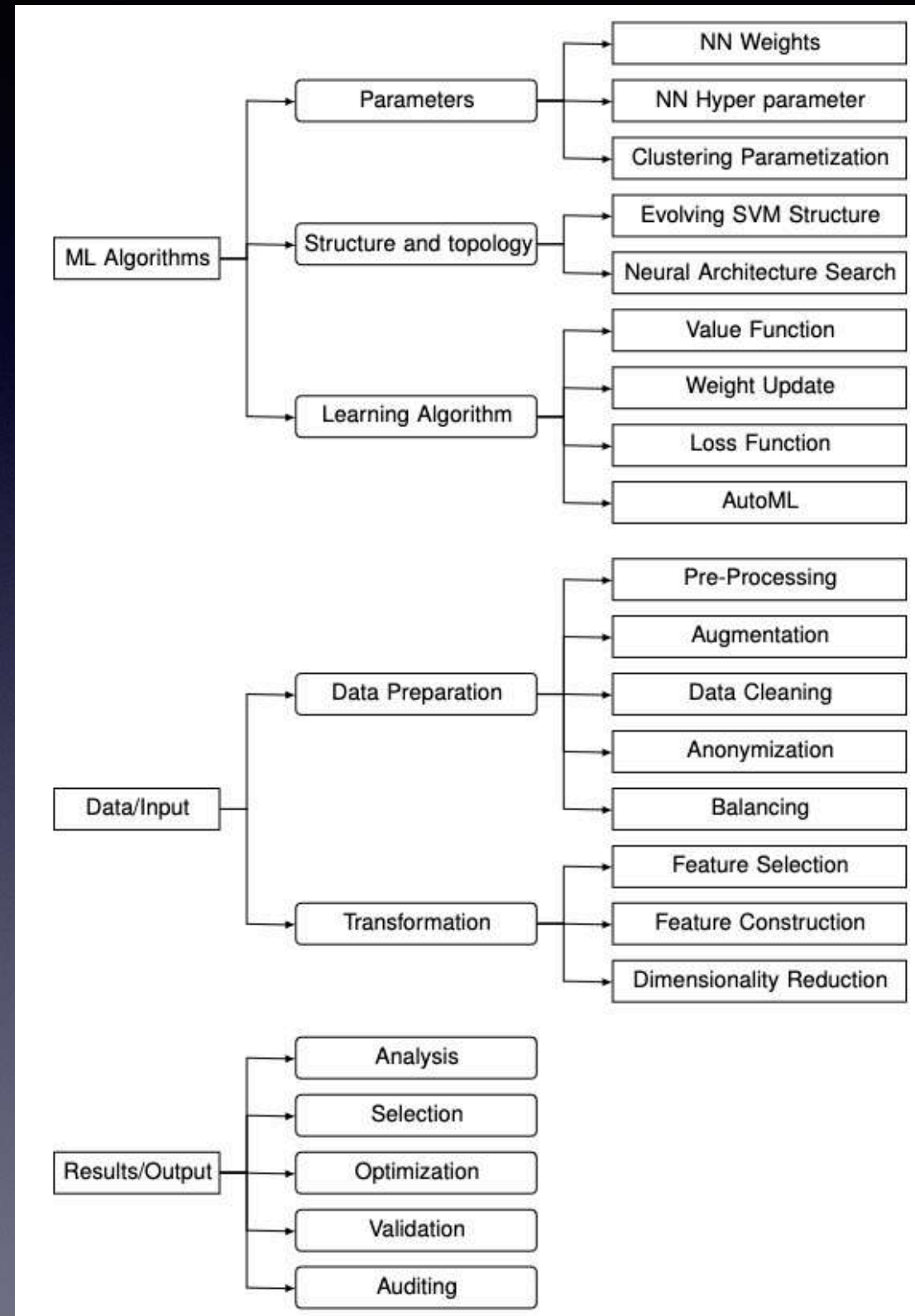
EC to ML methods



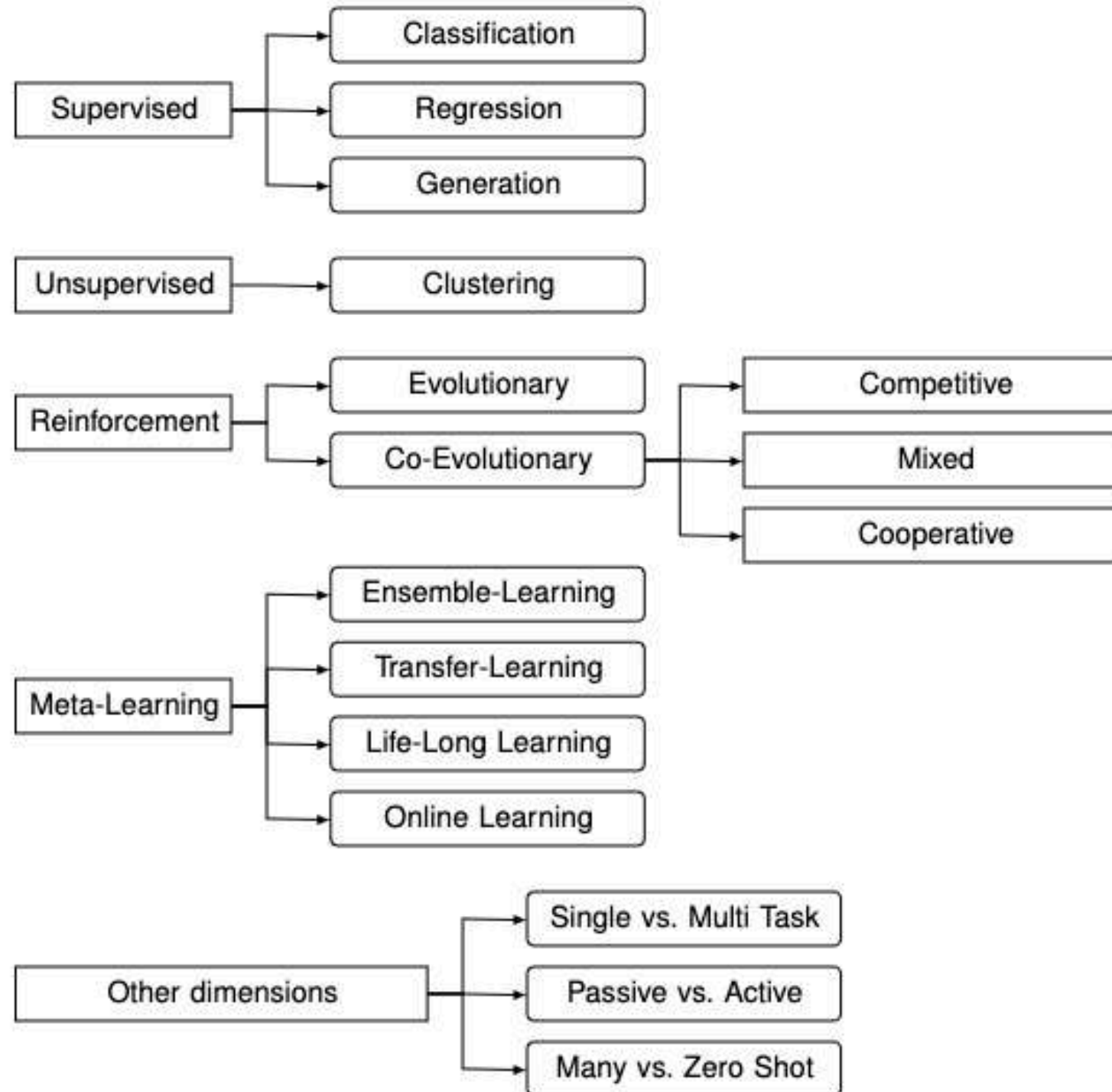
History of EML

EC to ML problems

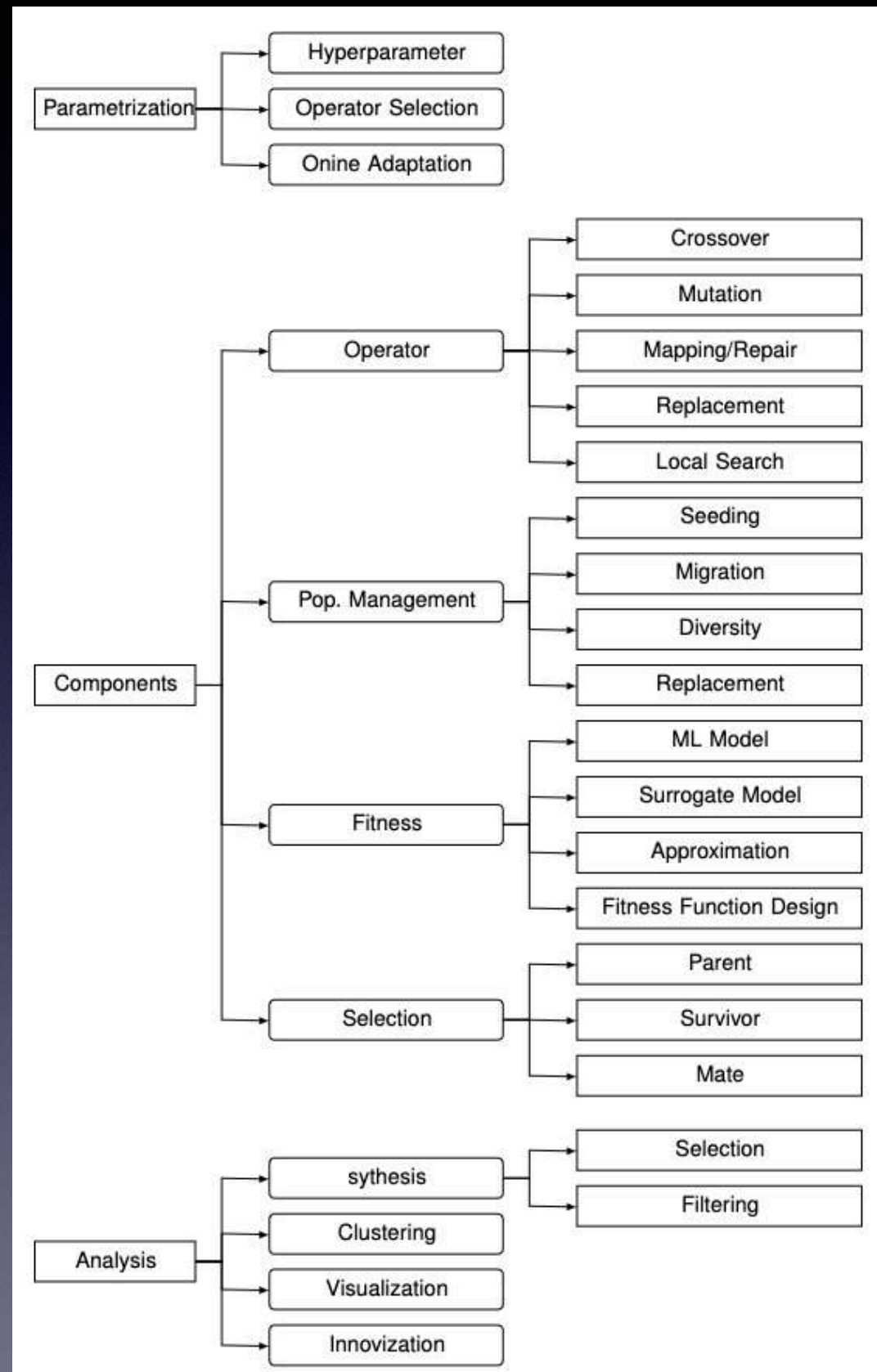




EC for ML Problems



ML for EC Methods



Preceding slides will appear in:
Handbook of Evolutionary Machine Learning
 W. Banzhaf, P. Machado and M. Zhang (Eds.)
 Springer Series in 'Genetic and Evolutionary
 Computing', Springer-Nature, 2023

Major Applications of GP

Major Applications of GP

- Modelling by symbolic regression

Major Applications of GP

- Modelling by symbolic regression
- Classification

Major Applications of GP

- Modelling by symbolic regression
- Classification
 - -> Machine Learning Applications

Major Applications of GP

- Modelling by symbolic regression
- Classification
 - -> Machine Learning Applications
- Code repair and synthesis

Major Applications of GP

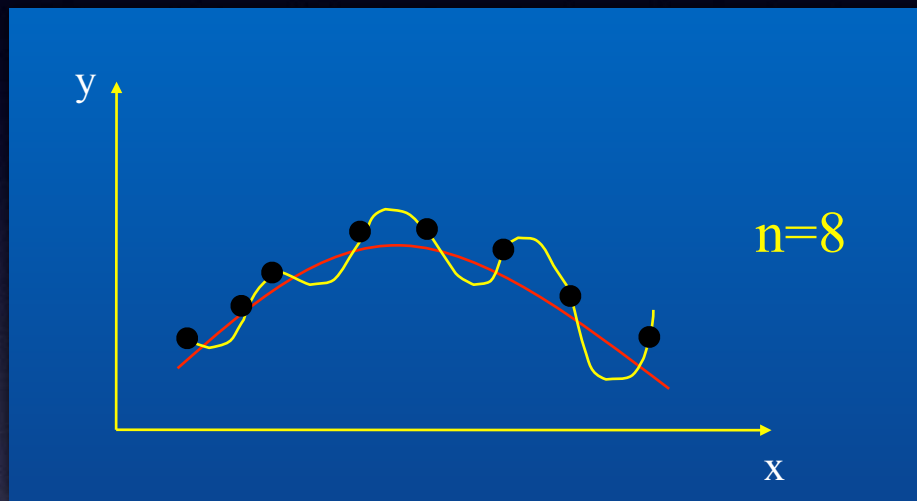
- Modelling by symbolic regression
- Classification
 - -> Machine Learning Applications
- Code repair and synthesis
 - -> Software Engineering

Major Applications of GP

- Modelling by symbolic regression
- Classification
 - -> Machine Learning Applications
- Code repair and synthesis
 - -> Software Engineering
- Meta-Learning

Symbolic Regression

Given $x_i, y_i \in \mathbb{R} \quad i \in \mathbb{N}$



Fitness cases

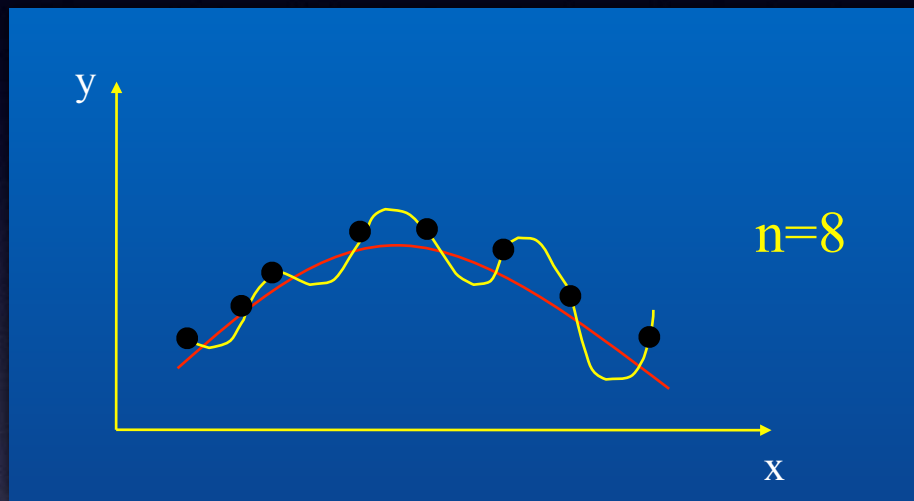
$$\text{Int: Prg} \times \mathbf{R} \rightarrow \mathbf{R}$$

$\uparrow \quad \uparrow$
 $x \quad y$

Symbolic Regression

Given $x_i, y_i \in \mathbf{R} \quad i \in \mathbf{N}$

Fitness cases



$$\text{Int: Prg} \times \mathbf{R} \rightarrow \mathbf{R}$$

$\begin{array}{c} \uparrow \quad \uparrow \\ x \quad y \end{array}$

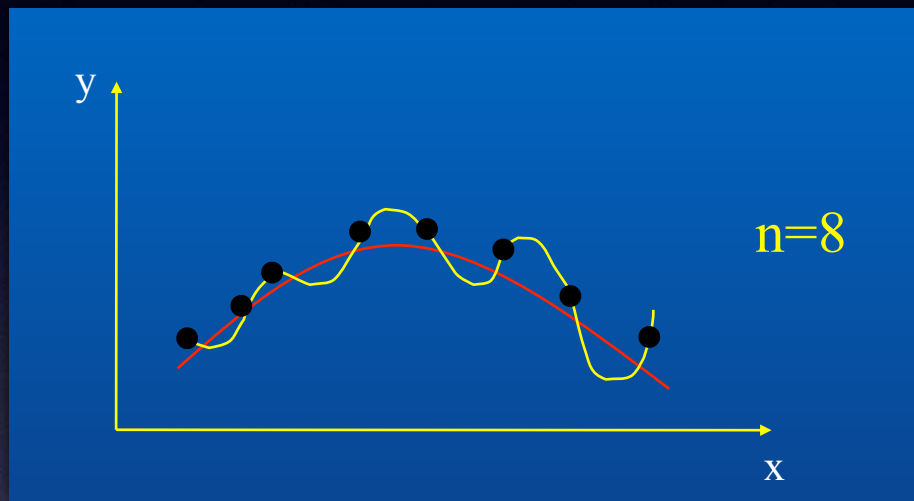
$$f(\text{Prg}) = \sum_{i=1}^n (\text{Int}(\text{Prg}, x_i) - y_i)^2$$

Quadratic Error

Symbolic Regression

Given $x_i, y_i \in \mathbf{R} \quad i \in \mathbf{N}$

Fitness cases



$$\text{Int: Prg} \times \mathbf{R} \rightarrow \mathbf{R}$$

$\begin{array}{c} \uparrow \quad \uparrow \\ x \quad y \end{array}$

$$f(\text{Prg}) = \sum_{i=1}^n \left(\text{Int}(\text{Prg}, x_i) - y_i \right)^2$$

Quadratic Error

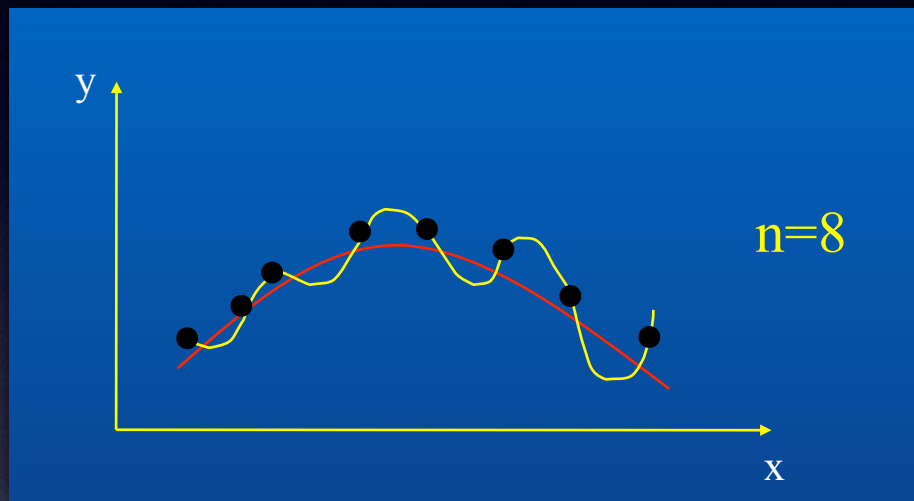
$$f(\text{Prg}) = \sum_{i=1}^n \left| \text{Int}(\text{Prg}, x_i) - y_i \right|$$

Absolute Error

Symbolic Regression

Given $x_i, y_i \in \mathbf{R} \quad i \in \mathbf{N}$

Fitness cases



$$\text{Int: Prg} \times \mathbf{R} \rightarrow \mathbf{R}$$

\uparrow
 x

\uparrow
 y

$$f(\text{Prg}) = \sum_{i=1}^n \left(\text{Int}(\text{Prg}, x_i) - y_i \right)^2$$

Quadratic Error

$$f(\text{Prg}) = \sum_{i=1}^n \left| \text{Int}(\text{Prg}, x_i) - y_i \right|$$

Absolute Error

Fitness independent of representation

GP can do Symbolic Function Regression

- Keijzer
- Korns
- Nguyen
-

Better GP Benchmarks: Community Survey Results and Proposals

19

Name	Variables	Equation	Training Set Testing Set
Keijzer-6 [25] [46]	1	$\sum_i^x \frac{1}{i}$	E[1, 50, 1] E[1, 120, 1]
Korns-12 [27]	5	$2 - 2.1 \cos(9.8x) \sin(1.3w)$	U[-50, 50, 10000] U[-50, 50, 10000]
Vladislavleva-4 [50]	5	$\frac{10}{5 + \sum_{i=1}^5 (x_i - 3)^2}$	U[0.05, 6.05, 1024] U[-0.25, 6.35, 5000]
Nguyen-7 [33]	1	$\ln(x+1) + \ln(x^2+1)$	U[0, 2, 20] None
Pagie-1 [36]	2	$\frac{1}{1+x^{-4}} + \frac{1}{1+y^{-4}}$	E[-5, 5, 0.4] None
Dow Chemical (see Section 6.1)	57	chemical process data ⁶	747 points 319 points
GP Challenge [56] (see Section 6.1)	8	protein energy data	1250–2000 per protein None

Table 5 Proposed symbolic regression benchmarks. In the training and testing sets, $U[a, b, c]$ is c uniform random samples drawn from a to b , inclusive. $E[a, b, c]$ is a grid of points evenly spaced with an interval of c , from a to b inclusive.

From: D. White et al., Gen. Progr. and Ev. Mach., 2013

Feynman AI

Table 4. Tested Feynman equations, part 1. Abbreviations in the “Methods used” column: da, dimensional analysis; bf, brute force; pf, polyfit; ev, set two variables equal; sym, symmetry; sep, separability. Suffixes denote the type of symmetry or separability (sym-, translational symmetry; sep*, multiplicative separability; etc.) or the preprocessing before brute force (e.g., bf-inverse means inverting the mystery function before bf).

Feynman Eq.	Equation	Solution Time (s)	Methods Used	Data Needed	Solved By Eureqa	Solved W/o da	Noise Tolerance
I.6.20a	$f = e^{-\theta^2/2}/\sqrt{2\pi}$	16	bf	10	No	Yes	10^{-2}
I.6.20	$f = e^{-\frac{\theta^2}{2\sigma^2}}/\sqrt{2\pi\sigma^2}$	2992	ev, bf-log	10^2	No	Yes	10^{-4}
I.6.20b	$f = e^{-\frac{(\theta-\theta_0)^2}{2\sigma^2}}/\sqrt{2\pi\sigma^2}$	4792	sym-, ev, bf-log	10^3	No	Yes	10^{-4}
I.8.14	$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$	544	da, pf-squared	10^2	No	Yes	10^{-4}
I.9.18	$F = \frac{Gm_1m_2}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$	5975	da, sym-, sym-, sep*, pf-inv	10^6	No	Yes	10^{-5}
I.10.7	$m = \frac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}}$	14	da, bf	10	No	Yes	10^{-4}
I.11.19	$A = x_1y_1 + x_2y_2 + x_3y_3$	184	da, pf	10^2	Yes	Yes	10^{-3}
I.12.1	$F = \mu N_n$	12	da, bf	10	Yes	Yes	10^{-3}
I.12.2	$F = \frac{q_1 q_2}{4\pi\epsilon r^2}$	17	da, bf	10	Yes	Yes	10^{-2}
I.12.4	$E_f = \frac{q_1}{4\pi\epsilon r^2}$	12	da	10	Yes	Yes	10^{-2}
I.12.5	$F = q_2 E_f$	8	da	10	Yes	Yes	10^{-2}
I.12.11	$F = q(E_f + Bv \sin \theta)$	19	da, bf	10	Yes	Yes	10^{-3}
I.13.4	$K = \frac{1}{2}m(v^2 + u^2 + w^2)$	22	da, bf	10	Yes	Yes	10^{-4}
I.13.12	$U = Gm_1m_2\left(\frac{1}{r_2} - \frac{1}{r_1}\right)$	20	da, bf	10	Yes	Yes	10^{-4}
I.14.3	$U = mgz$	12	da	10	Yes	Yes	10^{-2}
I.14.4	$U = \frac{k_{\text{spring}}x^2}{2}$	9	da	10	Yes	Yes	10^{-2}

Symbolic Regression Example

Symbolic Regression Example

- Schmidt & Lipson, from *Science* 2009 ...

Symbolic Regression Example

- Schmidt & Lipson, from *Science* 2009 ...
- ... Took an experimental (mechanical) system (chaotic double pendulum) ...

Symbolic Regression Example

- Schmidt & Lipson, from *Science* 2009 ...
- ... Took an experimental (mechanical) system (chaotic double pendulum) ...
- ... Measured key observables, angles and angular velocities ...

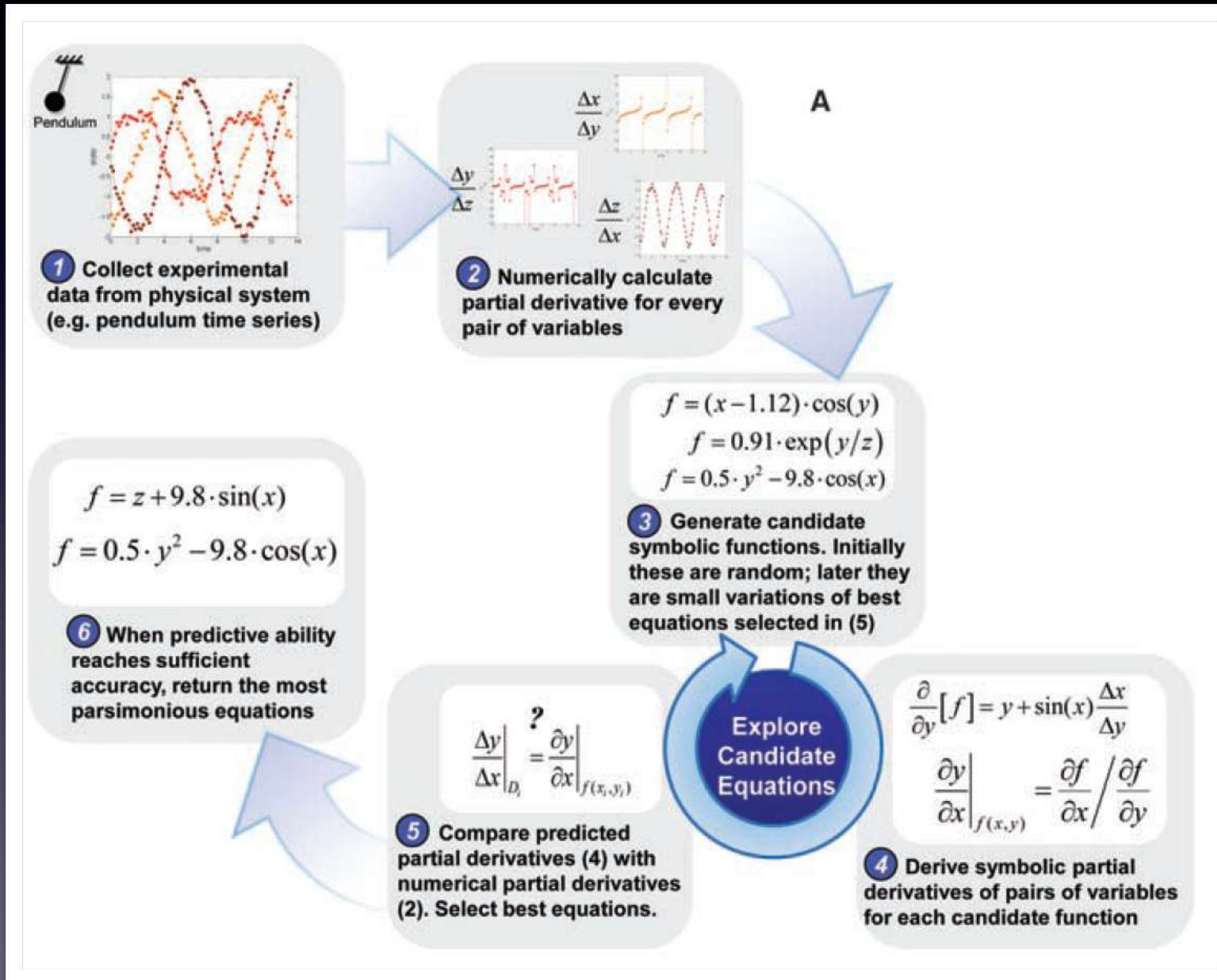
Symbolic Regression Example

- Schmidt & Lipson, from *Science* 2009 ...
- ... Took an experimental (mechanical) system (chaotic double pendulum) ...
- ... Measured key observables, angles and angular velocities ...
- ... Searched for equations describing laws relating these variables ...

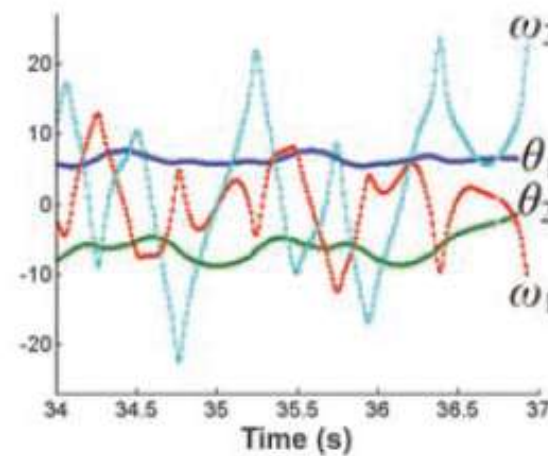
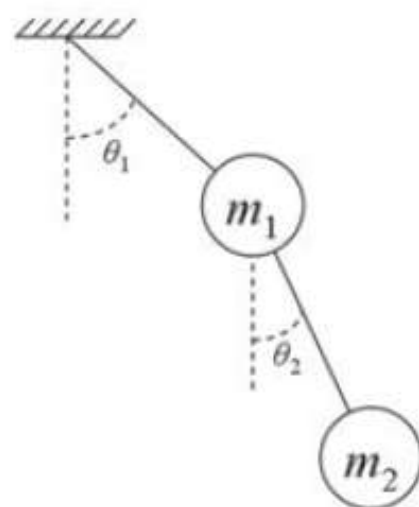
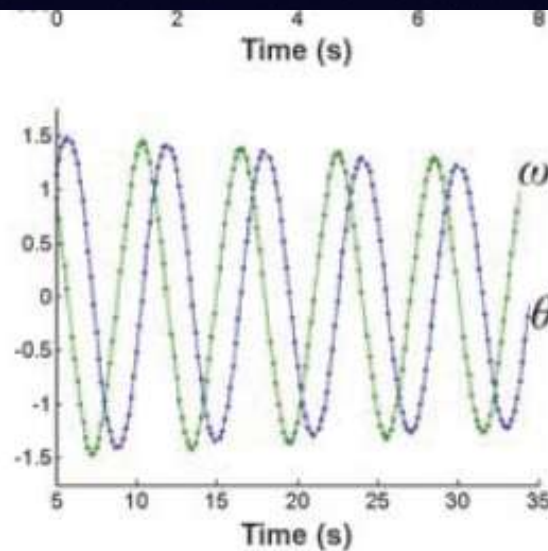
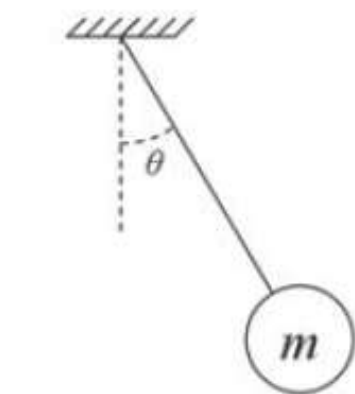
Symbolic Regression Example

- Schmidt & Lipson, from *Science* 2009 ...
- ... Took an experimental (mechanical) system (chaotic double pendulum) ...
- ... Measured key observables, angles and angular velocities ...
- ... Searched for equations describing laws relating these variables ...
- ... Found conservation law - the Hamiltonian

Processing Pipeline for GP



Resulting invariants



$$1.37 \cdot \omega^2 + 3.29 \cdot \cos(\theta)$$

Lagrangian

$$2.71\alpha + 0.054\omega - 3.54\sin(\theta)$$

Equation of motion

$$(x - 77.72)^2 + (y - 106.48)^2$$

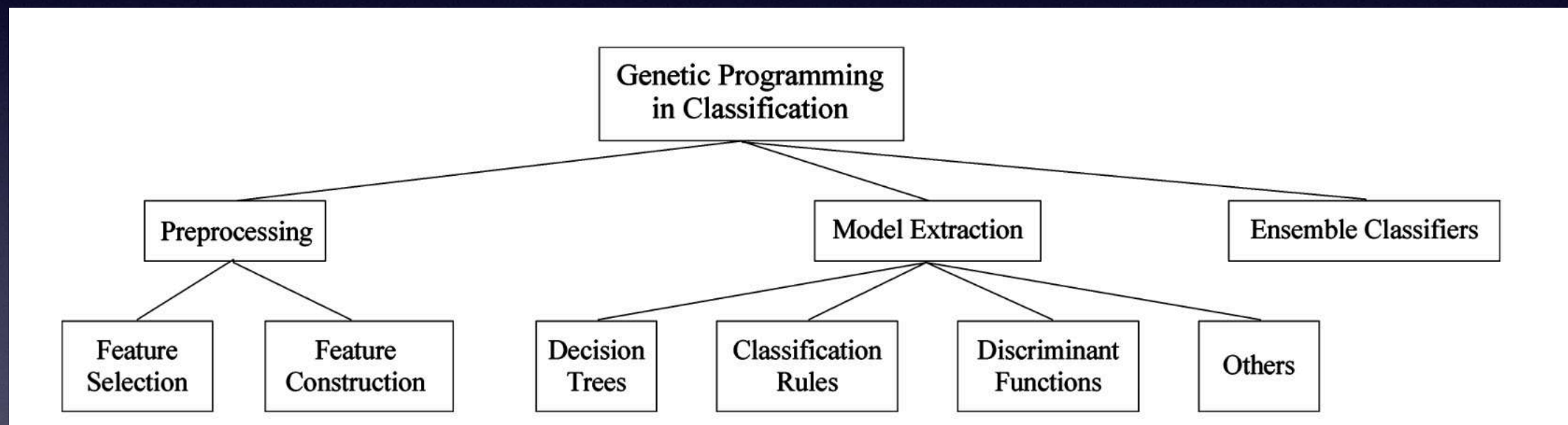
Circular manifold

$$\omega_1^2 + 0.32\omega_2^2 - 124.13\cos(\theta_1) - 46.82\cos(\theta_2) + 0.82\omega_1\omega_2\cos(\theta_1 - \theta_2)$$

Hamiltonian

Classification

GP uses in classification



From: P. Espejo et al, IEEE TA SMC-C 2010

GP as a tool for Software Engineering

- Test case generation
- Bug repair
- Improvement of code
- Co-evolution of tests and program repair
- Automatic programming (?)

Winner of the 2009 Human-Competitive Algorithms Competition

Fixing Software Bugs with GP

Forrest, Nguyen, Goues, Weimer, GECCO 2009

```
1 void zunebug_repair(int days) {
2   int year = 1980;
3   while (days > 365) {
4     if (isLeapYear(year)) {
5       if (days > 366) {
6         // days -= 366; // repair deletes
7         year += 1;
8       }
9       else {
10        }
11        days -= 366; // repair inserts
12      } else {
13        days -= 365;
14        year += 1;
15      }
16    }
17    printf("current year is %d\n", year);
18  }
```

Downloaded from <http://pastie.org/349916> (Jan. 2009).

- Infinite loop when input is last day of a leap year
- Repair is nontrivial - Microsoft recommended draining the battery
- GP discovered bug and repaired in 40 seconds

Repairing the Zune bug



AutoML Zero

Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020

AutoML-Zero: Evolving Machine Learning Algorithms From Scratch

Esteban Real^{*1} Chen Liang^{*1} David R. So¹ Quoc V. Le¹

Abstract

Machine learning research has advanced in multiple aspects, including model structures and learning methods. The effort to automate such research, known as AutoML, has also made significant progress. However, this progress has largely focused on the architecture of neural networks, where it has relied on sophisticated expert-designed layers as building blocks—or similarly restrictive search spaces. Our goal is to show that AutoML can go further: it is possible today to automatically discover complete machine learning algorithms just using basic mathematical operations as building blocks. We demonstrate this by introducing a novel framework that significantly reduces human bias through a generic search space. Despite the vastness of this space, evolutionary search can still discover two-layer neural networks trained by backpropagation. These simple neural networks can then be surpassed by evolving directly on tasks of interest, *e.g.* CIFAR-10 variants, where modern techniques emerge in the top algorithms, such as bilinear interactions, normalized gradients, and weight averaging. Moreover, evolution adapts algorithms to different task types: *e.g.*, dropout-like techniques appear when little data is available. We believe these preliminary successes in discovering machine learning algorithms from scratch indicate a promising new direction for the field.

1. Introduction

In recent years, neural networks have reached remarkable performance on key tasks and seen a fast increase in their popularity [*e.g.* 30, 75, 90]. This success was only possible

algorithms, possibly reducing the innovation potential of AutoML. Innovation is also limited by having fewer options: you cannot discover what you cannot search for [21]. Indeed, dominant aspects of performance are often left out



AutoML Zero



Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020

- AutoML Zero has
 - 3 major parts: Setup, Predict, Learn
 - 2 datasets: Dtrain, Dvalid

```
# (Setup, Predict, Learn) is the input ML algorithm.
# Dtrain / Dvalid is the training / validation set.
# sX/vX/mX: scalar/vector/matrix var at address X.
def Evaluate(Setup, Predict, Learn, Dtrain, Dvalid):
    # Zero-initialize all the variables (sX/vX/mX).
    initialize_memory()

    Setup() # Execute setup instructions.

    for (x, y) in Dtrain:
        v0 = x # x will now be accessible to Predict.
        Predict() # Execute prediction instructions.
        # s1 will now be used as the prediction.
        s1 = Normalize(s1) # Normalize the prediction.
        s0 = y # y will now be accessible to Learn.
        Learn() # Execute learning instructions.

    sum_loss = 0.0
    for (x, y) in Dvalid:
        v0 = x
        Predict() # Only execute Predict(), not Learn().
        s1 = Normalize(s1)
        sum_loss += Loss(y, s1)

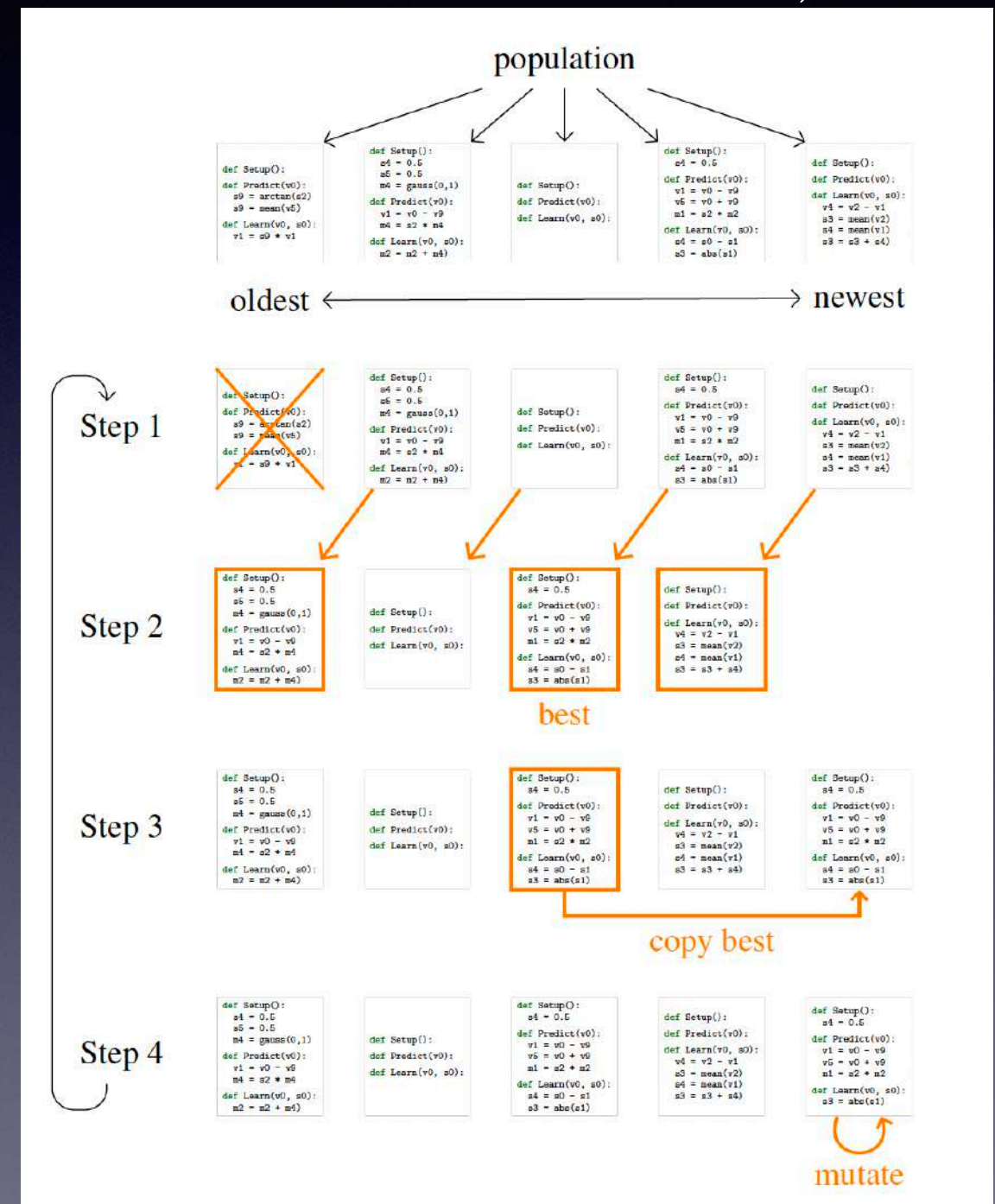
    mean_loss = sum_loss / len(Dvalid)
    # Use validation loss to evaluate the algorithm.
    return mean_loss
```


AutoML Zero

Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020

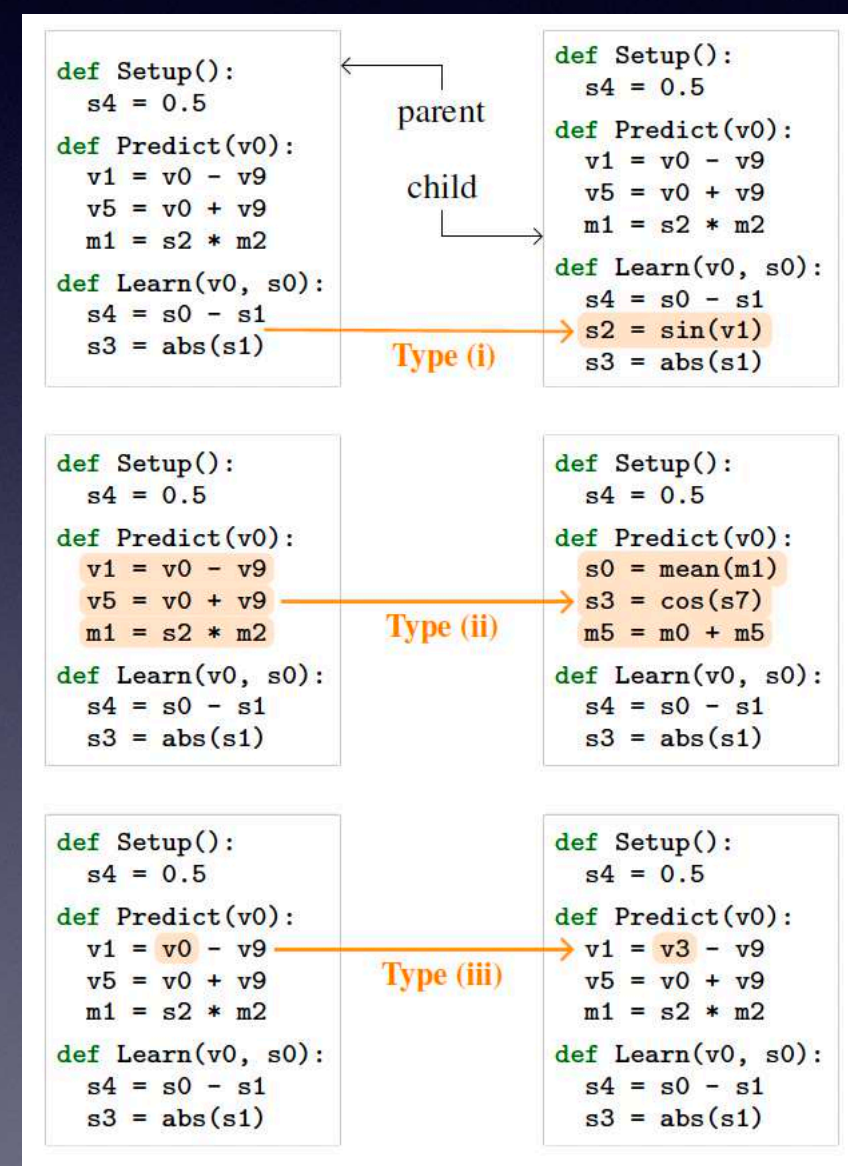
- AutoML Zero has
 - 3 major parts: Setup, Predict, Learn
 - 2 datasets: Dtrain, Dvalid
- Regularized Evolution



Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020

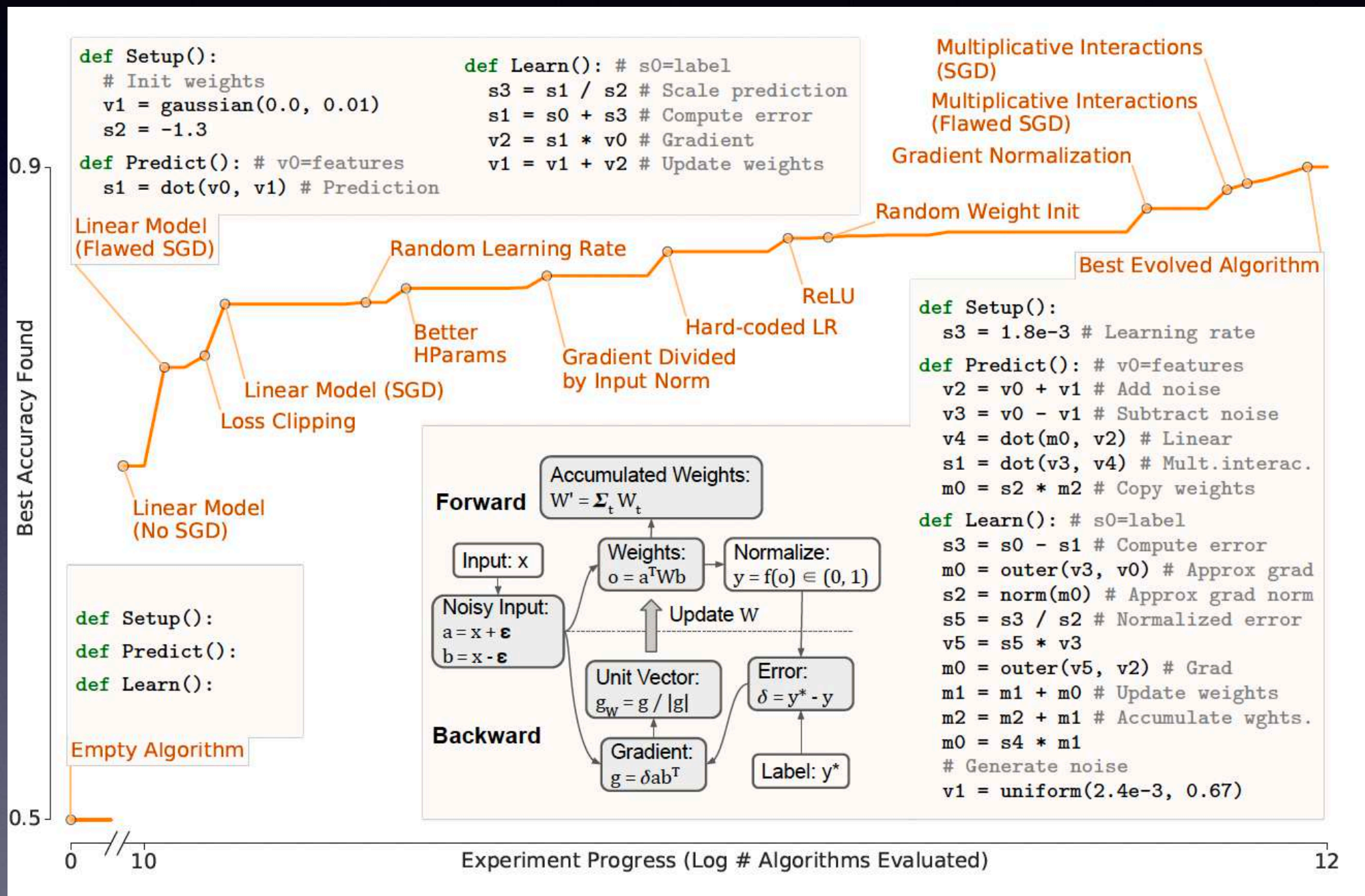
- AutoML Zero has
 - 3 major parts: Setup, Predict, Learn
 - 2 datasets: Dtrain, Dvalid
- Regularized Evolution
- Linear GP system with data structures



AutoML Zero

Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020





AutoML Zero

Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020

- AutoML Zero has
 - 3 major parts: Setup, Predict, Learn
 - 2 datasets: Dtrain, Dvalid
- Regularized Evolution
- Linear GP system with data structures

Starting from empty component functions and using only basic mathematical operations, we evolved linear regressors, neural networks, gradient descent, multiplicative interactions, weight averaging, normalized gradients, etc.



AutoML Zero

Is it possible to automatically discover machine learning algorithms just using mathematical operations as building blocks?

Esteban Real et al., 2020

- AutoML Zero has
 - 3 major parts: Setup, Predict, Learn
 - 2 datasets: Dtrain, Dvalid
- Regularized Evolution
- Linear GP system with data structures

Starting from empty component functions and using only basic mathematical operations, we evolved linear regressors, neural networks, gradient descent, multiplicative interactions, weight averaging, normalized gradients, etc.

We want AI agents that can discover like we can, not which contain what we have discovered. Building in our discoveries only makes it harder to see how the discovering process can be done. R. Sutton, 2019

Broader Goals of GP

- Not just equations from data
- ... but Machine Learning
- ... Meta-Learning
- ... Autonomous Programming
- ... Evolving Structures of arbitrary kind

Challenges and Research Questions

Challenges and Research Questions

- Static Fitness

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code
- No role for non-expressed material

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code
- No role for non-expressed material RNA-type

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code
- No role for non-expressed material RNA-type
- All genes are passed directly from parent to offspring, without any further specification

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code
- No role for non-expressed material RNA-type
- All genes are passed directly from parent to offspring, without any further
● specification Epigenetics

Challenges and Research Questions

- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code
- No role for non-expressed material RNA-type
- All genes are passed directly from parent to offspring, without any further
● specification Epigenetics
- Problems with Scalability

Challenges and Research Questions

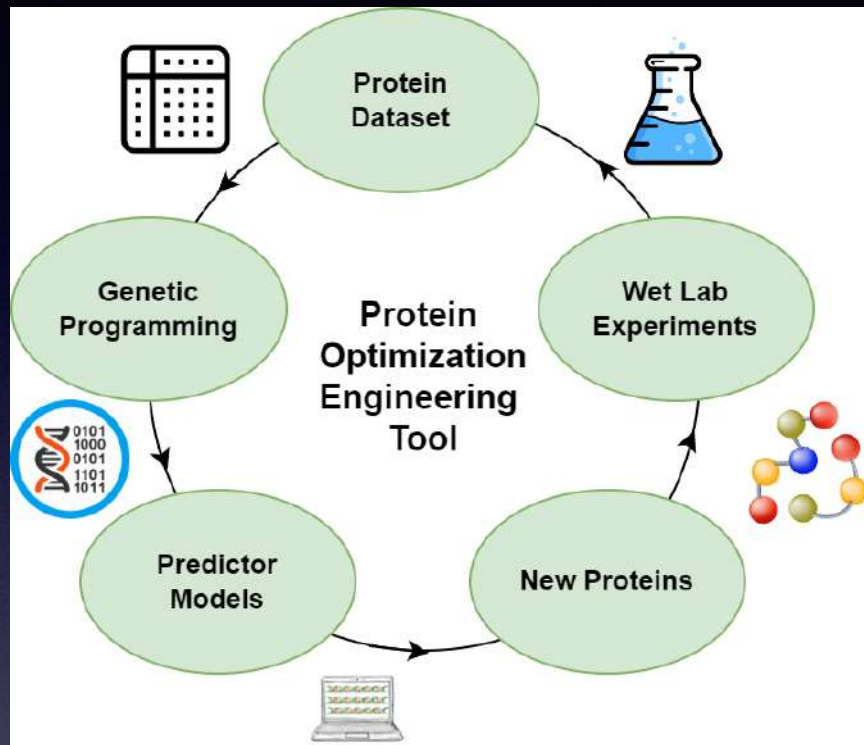
- Static Fitness Dynamic processes, continuously changing environment
- Fixed Representation Interactions of the material under evolution
- Closed Systems Real-time evolution in (open) stream processing
- Genomes composed of discrete and independent genes
- Feedback and Networks
- Simple genotype-phenotype map Regulation
- Predetermined operator features Operators part of the code
- No role for non-expressed material RNA-type
- All genes are passed directly from parent to offspring, without any further
● specification Epigenetics
- Problems with Scalability Development

Our current work in GP

- Artificial Regulatory Networks (since 2004)
- Computational Evolution (since 2005)
- GPGPUs (since 2007)
- Evolvability (since 2010)
- Epigenetics (since 2012)
- Novelty (since 2014)
- Software Engineering - Bug repair, code synthesis (since 2017)
- Policy evolution (RL type tasks) (since 2019)

The POET Project (NIH funded)

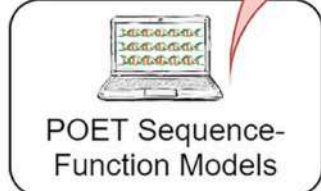
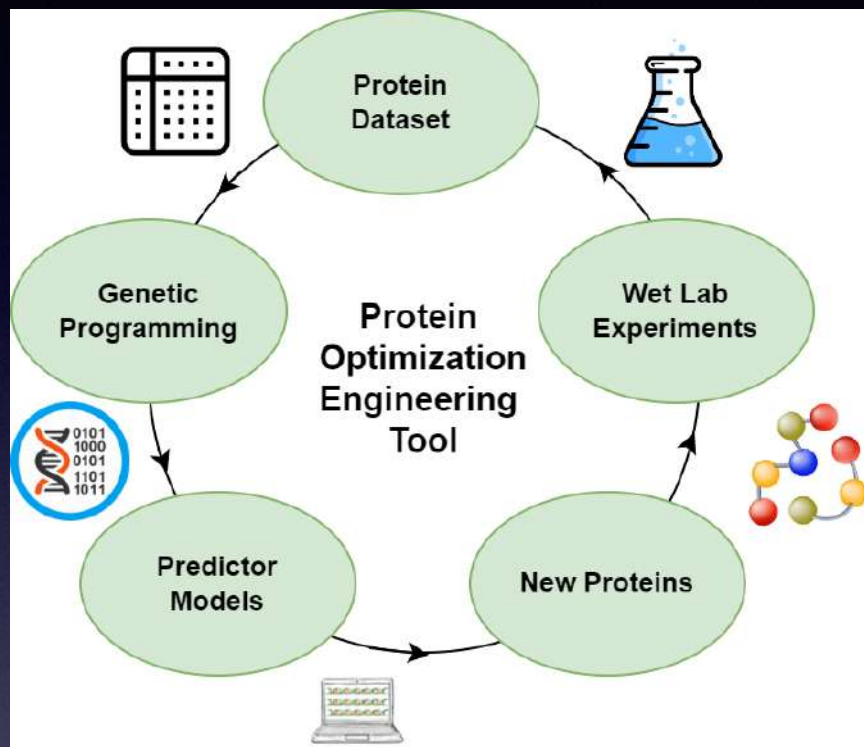
Goal: Find better proteins/oligomers for a specific function -
here Chemical Exchange Saturation Transfer (CEST)
contrast for MRI applications



GP does Amino-acid Sequence -> Function Modelling

The POET Project (NIH funded)

Goal: Find better proteins/oligomers for a specific function -
here Chemical Exchange Saturation Transfer (CEST)
contrast for MRI applications



POET Sequence-Function Models

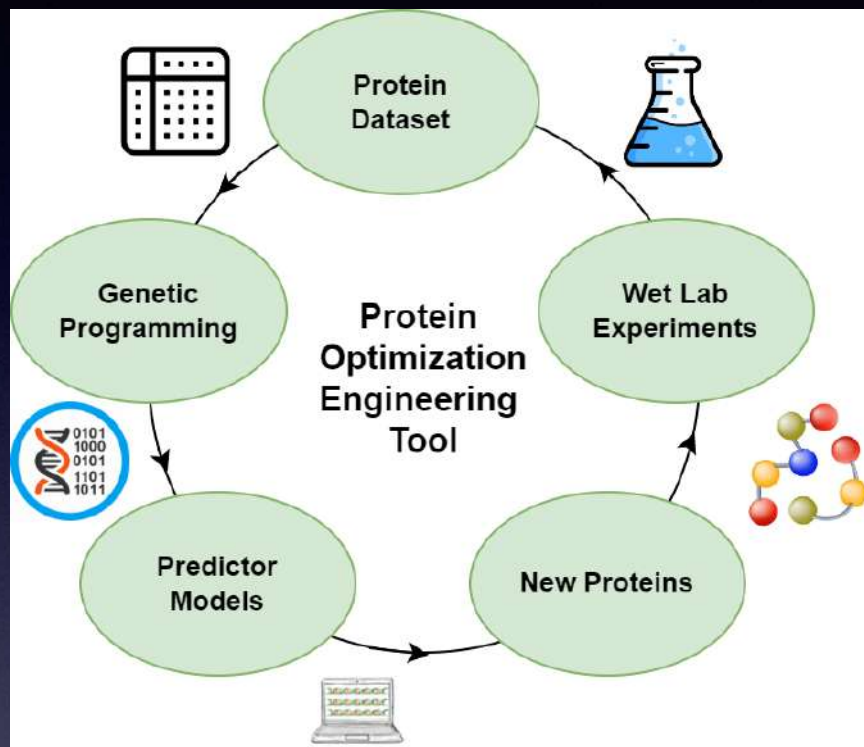
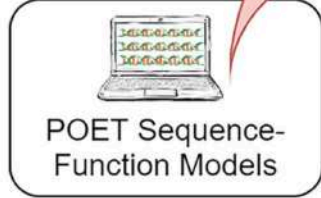
ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1
2	TTTTETT	0.769	0
3	AKK	13.31	1
4	TQ	-1.88	1

Individual: Motif rule set

GP does Amino-acid Sequence -> Function Modelling

The POET Project (NIH funded)

Goal: Find better proteins/oligomers for a specific function -
here Chemical Exchange Saturation Transfer (CEST)
contrast for MRI applications

ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1
2	TTTTETT	0.769	0
3	AKK	13.31	1
4	TQ	-1.88	1

Individual: Motif rule set

Crossover

GP does Amino-acid Sequence -> Function Modelling

Parent A				Parent B			
ID	Motif	Weight	Status	ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1	1	TTTDT	0.419	1
2	TTTTETT	0.769	0	2	RTT	-2.01	1
3	AKK	13.31	1	3	HA	10.6	0
4	TQ	-1.88	1	4	P	8.86	1

Recombination of rules to create a new offspring

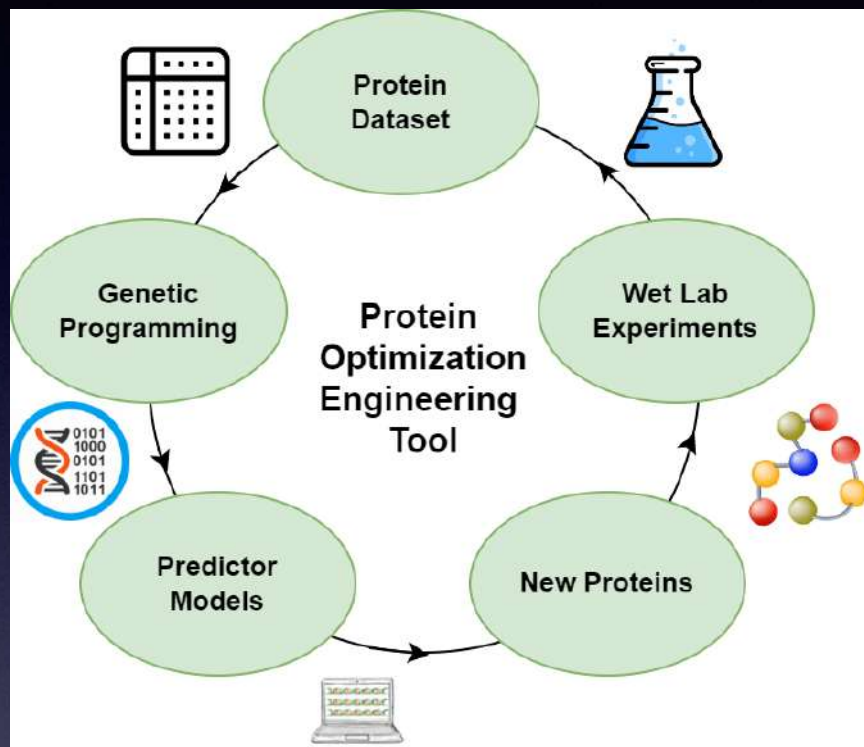
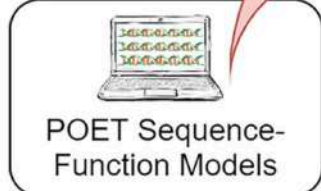
Offspring			
ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1
2	TTTTETT	0.769	0
3	TTTDT	0.419	1
4	RTT	-2.01	1
5	AKK	13.31	1
6	TQ	-1.88	1
7	P	8.86	1

Shrink

ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1
2	TTTTETT	0.769	0
3	TTTDT	0.419	1
4	RTT	-2.01	1
5	AKK	13.31	1
6	TQ	-1.88	1

The POET Project (NIH funded)

Goal: Find better proteins/oligomers for a specific function -
here Chemical Exchange Saturation Transfer (CEST)
contrast for MRI applications

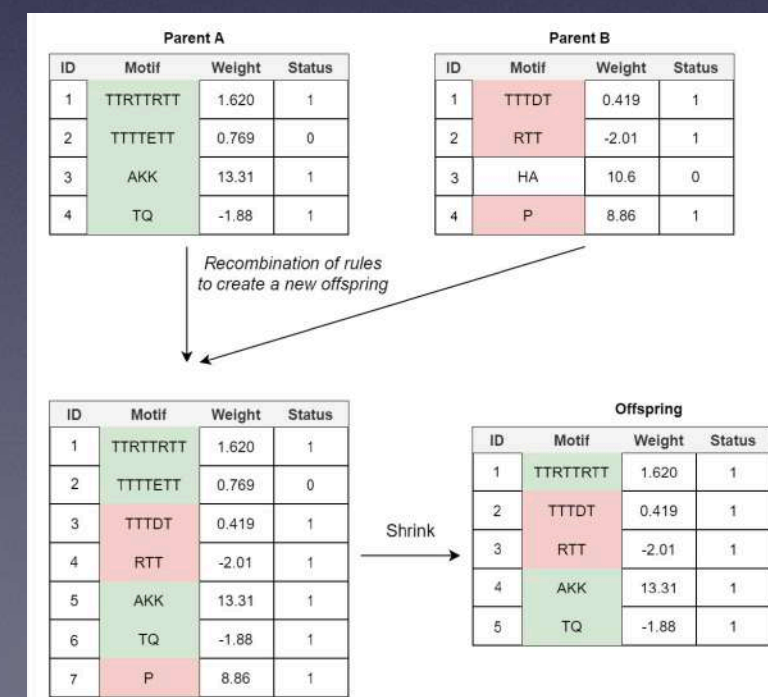
ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1
2	TTTTETT	0.769	0
3	AKK	13.31	1
4	TQ	-1.88	1

Individual: Motif rule set

Crossover

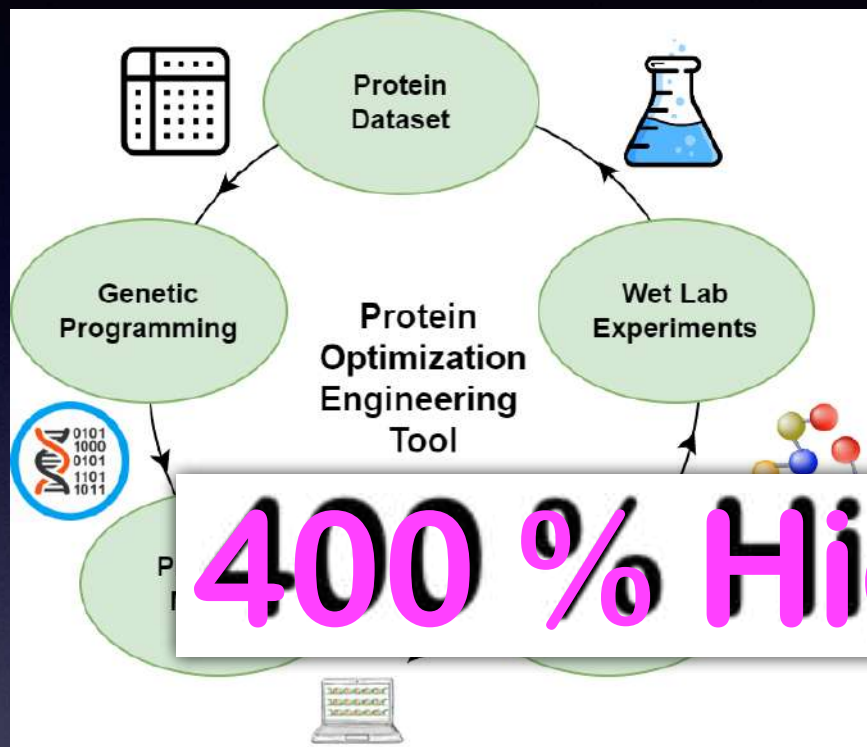
GP does Amino-acid Sequence -> Function Modelling

Epoch	Data points	Training RMSE	Test RMSE	Best overall RMSE	Average overall RMSE	Total rules #	Expressed rules #
E1	42	1.272	1.307	10.189	16.583	97	44
E2	51	1.558	1.576	11.001	15.882	97	45
E3	61	2.238	2.258	10.185	13.276	96	45
E4	71	2.308	2.326	10.096	12.786	96	44
E5	82	2.898	2.919	8.974	11.531	97	44
E6	92	3.651	3.674	8.247	11.349	96	44
E7	102	3.923	3.944	8.686	10.646	97	44
E8	112	4.891	4.916	7.845	9.954	97	44



The POET Project (NIH funded)

Goal: Find better proteins/oligomers for a specific function -
here Chemical Exchange Saturation Transfer (CEST)
contrast for MRI applications



ID	Motif	Weight	Status
1	TTRTTRTT	1.620	1
2	TTTTETT	0.769	0
3	AKK	13.31	1
4	TQ	-1.88	1

400 % Higher CEST Contrast

GP does Amino-acid Sequence -> Function Modelling

Epoch	Data points	Training RMSE	Test RMSE	Best overall RMSE	Average overall RMSE	Total rules #	Expressed rules #
E1	42	1.272	1.307	10.189	16.583	97	44
E2	51	1.558	1.576	11.001	15.882	97	45
E3	61	2.238	2.258	10.185	13.276	96	45
E4	71	2.308	2.326	10.096	12.786	96	44
E5	82	2.898	2.919	8.974	11.531	97	44
E6	92	3.651	3.674	8.247	11.349	96	44
E7	102	3.923	3.944	8.686	10.646	97	44
E8	112	4.891	4.916	7.845	9.954	97	44

PeerJ Physical Chemistry 4 (2022) e24



“The Bitter Lesson”





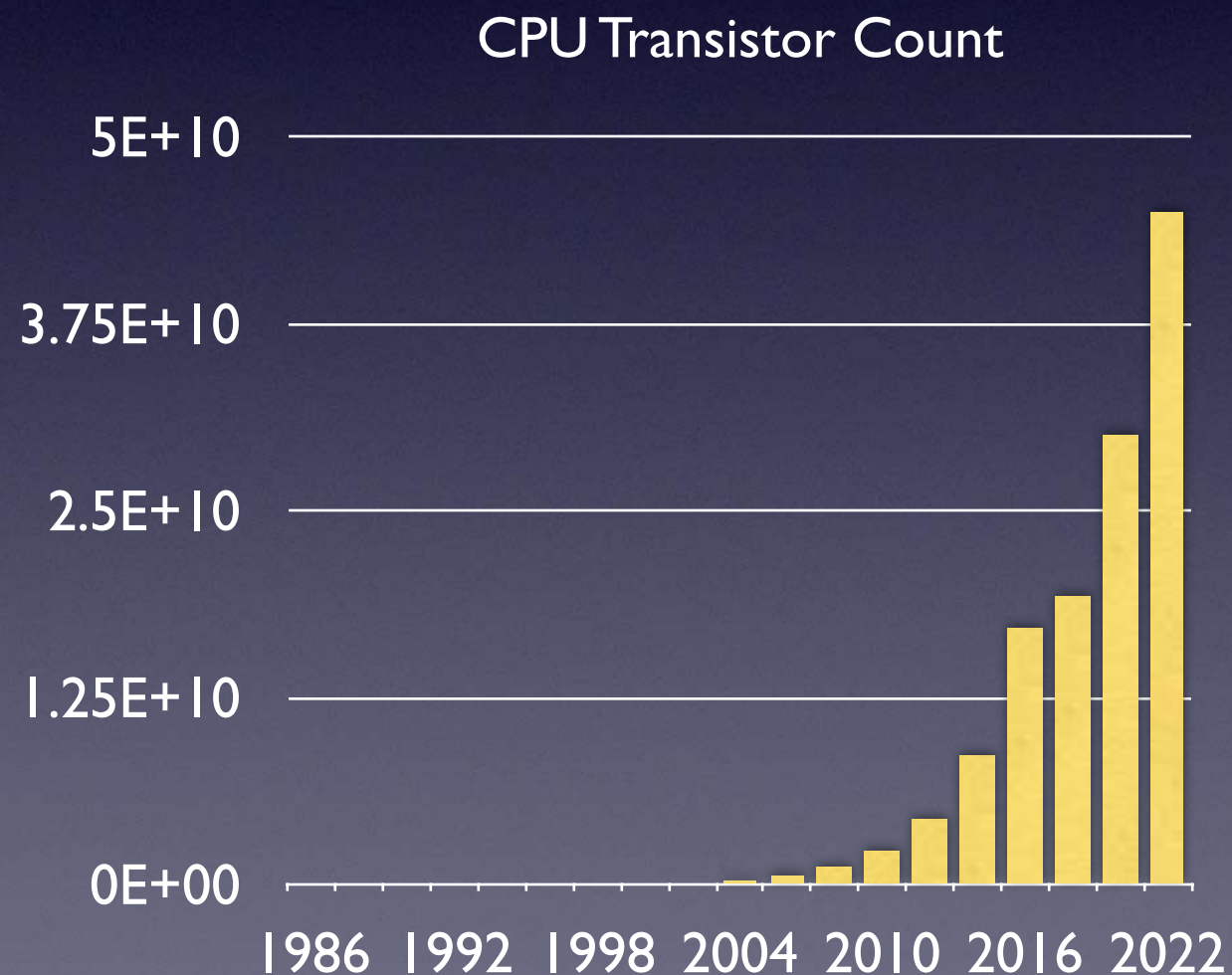
“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge





“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity



“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity
- However, computation over the long run has beaten this ingenuity



“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity
- However, computation over the long run has beaten this ingenuity
 - Computer Chess, 1997



“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity
- However, computation over the long run has beaten this ingenuity
 - Computer Chess, 1997
 - Computer Go, 2017



“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity
- However, computation over the long run has beaten this ingenuity
 - Computer Chess, 1997
 - Computer Go, 2017
 - Speech recognition, computer vision, human language



“The Bitter Lesson”

The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity
- However, computation over the long run has beaten this ingenuity
 - Computer Chess, 1997
 - Computer Go, 2017
 - Speech recognition, computer vision, human language

One thing that should be learnt from the bitter lesson is the great power of general purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are *search* and *learning*.



“The Bitter Lesson”

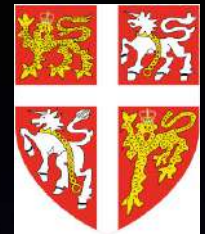
The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin. *Richard Sutton, 2019*

- Moore’s law overtakes leveraging of human knowledge
- Most AI researchers have tried to put human knowledge into their systems which required considerable ingenuity
- However, computation over the long run has beaten this ingenuity
 - Computer Chess, 1997
 - Computer Go, 2017
 - Speech recognition, computer vision, human language

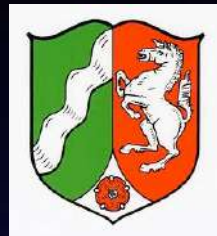
One thing that should be learnt from the bitter lesson is the great power of general purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are *search* and *learning*.

We want AI agents that can discover like we can, not which contain what we have discovered. Building in our discoveries only makes it harder to see how the discovering process can be done.

Funding Sources since 1993



Province of Newfoundland



Land Nordrhein-Westfalen



Students

Markus Brameier (PhD)
Jens Busch (PhD)
Peter Dittrich (PhD)
Udo Feldkamp (PhD)
Benjamin Fowler (PhD)
Nathan Haut (PhD student)
Ting Hu (PhD)
Wolfgang Kantschik (PhD)
Robert E. Keller (PhD)
Christian Lasarczyk (PhD)
Andre Leier (PhD)
Iliya Miralavy (PhD student)
Jens Niehaus (PhD)
Peter Nordin (PhD)
Javad Rahimipour (PhD)
Esteban Ricalde (PhD)
Christoph Richter (PhD)
Ralf Stadelhofer (PhD)
Xiaonan (Shelly) Wu (PhD)
Scott Watson (PhD cand)
Jens Ziegler (PhD)

Postdocs

Peter Dittrich (now Jena)
Simon Harding (now Bristol & York)
Stephen Kelly (now McMaster U)
Taras Kowaliw (now Goldman-Sachs)
Andre Leier (now U of Alabama)
Ken Reid (MSU)
Nicolas Scalzitti (MSU)
Jory Schossau (MSU)
Garnett Wilson (now Verafin Inc.)
Yuan Yuan (now Beihang U.)
Jens Ziegler (Duesseldorf)

Collaborators and Colleagues

Frank Francone (Denver, CO)
Steven Gustafson (GE, New York)
Bill Langdon (UCL, London)
Penousal Machado (Coimbra U)
Thomas Meyer (PIN, Washington)
Daniele Miorandi (U Padova)
Joshua Payne (ETH, Zuerich)
Hilmar Rauhe (Informium, Koeln)
Bas Straatman (U Calgary)
Bing Xue (VUW, Wellington)
Lidia Yamamoto (Brussels)
Mengjie Zhang (VUW, Wellington)

Martyn Amos (MMU, Manchester)
Guillaume Beslon (INRIA, Lyon)
Qi Chen (VUW, Wellington)
Yuanzhu Chen (MUN)
Pierre Collet (U Strasbourg)
Amy Connolly (OSU)
Vinicius de Melo (U Fed Sao Paulo)
Rodolphe Devillers (MUN)
Emily Dolson (MSU)
James Foster (U Idaho)
Assaf Gilad (MSU)
Cedric Gondro (MSU)
Orland Hoeber (U Regina)
Christian Igel (DKU, Copenhagen)
Christian Jacob (U Calgary)
Francois Kepes (ISSB, Evry)
John Koza (3rdMillenium, Los Altos)
Thomas Kron (RTWH, Aachen)
Cheyenne Laue (U Montana)
Fredrik Liljeros (Stockholm U)
Joern Mehnert (Cranfield, UK)

Yi Mei (VUW, Wellington)
Julian Miller (York, UK)
Jason Moore (Dartmouth, NH)
Michael O'Neill (UCD, Dublin)
Miguel Nicolau (UCD, Dublin)
Christoph Niemeyer (TU Dortmund)
Gabriela Ochoa (Stirling U)
Charles Ofria (MSU)
Hannah Peeler (Apple)
Nelishia Pillay (U Pretoria)
Riccardo Poli (Essex U)
Miroslav Radman (Paris V)
Jeremy Ramsden (em, Cranfield, UK)
Gustavo Recio (U Madrid III)
Julie Rolla (NASA)
Conor Ryan (U Limerick)
Marc Schönauer (INRIA)
Lucas Sekanina (U Brno)
Andrew Sloss (ARM Inc)
Lee Spector (Hampshire College)
Susan Stepney (U York, UK)
Dieter Suter (TU Dortmund)
Jonathan Timmis (U York, UK)
Marco Tomassini (em, U Lausanne)
Christian Tschudin (U Basel)
Andy Tyrell (U York, UK)
Leonardo Vanneschi (U Lisbon)
Andrew Vardy (MUN)
David White (Manchester)
Roger White (MUN)
Alden Wright (U Montana)
Annie Wu (U Central Florida)
Tina Yu (res, MUN)



Thanks - Discussion