



Metaheuristic Optimization

14. Genetic Programming

Thomas Weise · 汤卫思

tweise@hfuu.edu.cn · <http://iao.hfuu.edu.cn>

Hefei University, South Campus 2
Faculty of Computer Science and Technology
Institute of Applied Optimization
230601 Shushan District, Hefei, Anhui, China
Econ. & Tech. Devel. Zone, Jinxiu Dadao 99

合肥学院 南艳湖校区 / 南2区
计算机科学与技术系
应用优化研究所
中国 安徽省 合肥市 蜀山区 230601
经济技术开发区 锦绣大道99号

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis



Section Outline

- 1 Introduction
- 2 Symbolic Regression
- 3 Genetic Programming
- 4 Tree Creation
- 5 Data Mining
- 6 Bloat
- 7 Representations in GP
- 8 Epistasis

- What is Genetic Programming?

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
 - population-based optimization algorithm

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
 - population-based optimization algorithm
 - proceeds in iterations (called generations)

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
 - population-based optimization algorithm
 - proceeds in iterations (called generations)
 - population undergoes survival selection in each generation

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
 - population-based optimization algorithm
 - proceeds in iterations (called generations)
 - population undergoes survival selection in each generation
 - selected individuals can reproduce via mutation and recombination

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
 - population-based optimization algorithm
 - proceeds in iterations (called generations)
 - population undergoes survival selection in each generation
 - selected individuals can reproduce via mutation and recombination
- But what does it optimize? What is it good for? What is the difference to a Genetic Algorithm, Evolution Strategy, or Differential Evolution?

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
- But what does it optimize? What is it good for? What is the difference to a Genetic Algorithm, Evolution Strategy, or Differential Evolution?
 - ① It evolves programs or

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
- But what does it optimize? What is it good for? What is the difference to a Genetic Algorithm, Evolution Strategy, or Differential Evolution?
 - ① It evolves programs or
 - ② tree or graph data structures.

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
- But what does it optimize? What is it good for? What is the difference to a Genetic Algorithm, Evolution Strategy, or Differential Evolution?
 - ① It evolves **programs** or
 - ② tree or graph data structures.

Programs in the wider sense of the word: Algorithm-like, interpretable instructions. A formula is a program, a decision tree is a program etc...

- What is Genetic Programming?
- Genetic Programming is an Evolutionary Algorithm.
- But what does it optimize? What is it good for? What is the difference to a Genetic Algorithm, Evolution Strategy, or Differential Evolution?
 - ① It evolves programs or
 - ② tree or graph data structures.

Programs in the wider sense of the word: Algorithm-like, interpretable instructions. A formula is a program, a decision tree is a program etc...

- Made popular by Koza [1–3], but earlier works by e.g., Fogel et al. [4], Smith [5], Forsyth [6–8], Cramer [9], Schmidhuber [10, 11], Hicklin [12], and Fujuki [13]

What's it good for?



So what is Genetic Programming good for?

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression

[3, 9, 14–33]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]
- Systems Security [52, 61, 74, 81, 83, 97]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]
- Systems Security [52, 61, 74, 81, 83, 97]
- Networking and Communication [19, 27, 61, 70–73, 75–77, 82, 83, 90–92, 96–109]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]
- Systems Security [52, 61, 74, 81, 83, 97]
- Networking and Communication [19, 27, 61, 70–73, 75–77, 82, 83, 90–92, 96–109]
- Multi-Agent Systems / Behaviors [70, 90, 91, 95, 102, 110–113]

Just to name a few . . .

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]
- Systems Security [52, 61, 74, 81, 83, 97]
- Networking and Communication [19, 27, 61, 70–73, 75–77, 82, 83, 90–92, 96–109]
- Multi-Agent Systems / Behaviors [70, 90, 91, 95, 102, 110–113]
- Chemistry [38, 39, 71–73, 82, 99, 104–106, 114–118]

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]
- Systems Security [52, 61, 74, 81, 83, 97]
- Networking and Communication [19, 27, 61, 70–73, 75–77, 82, 83, 90–92, 96–109]
- Multi-Agent Systems / Behaviors [70, 90, 91, 95, 102, 110–113]
- Chemistry [38, 39, 71–73, 82, 99, 104–106, 114–118]
- Software (Engineering) [2, 6, 9, 11–13, 29, 64, 69, 92, 103, 119–125]

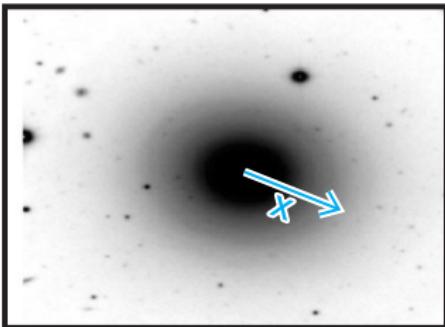
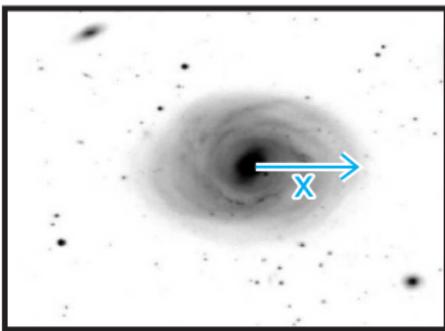
Just to name a few...

Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

Symbolic Regression: Why?

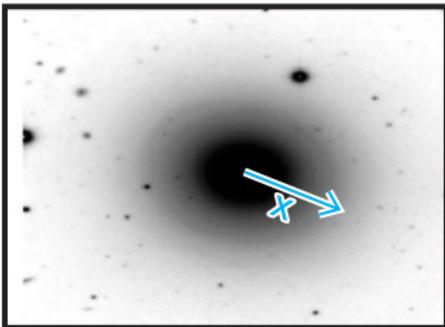
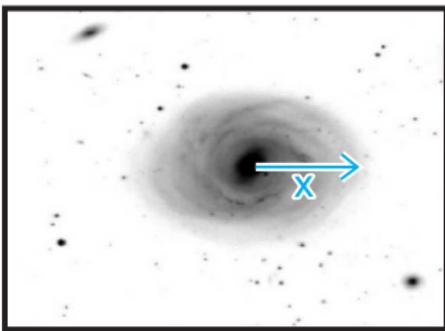
- A physicist runs an experiment.



Example: Find formula for radial brightness of a galaxy. (Just to illustrate the idea, the formulas and data illustrated in the following are **unrelated** to this example)

Symbolic Regression: Why?

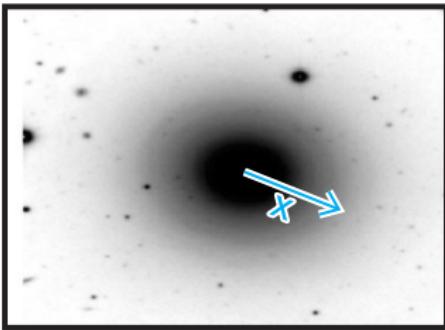
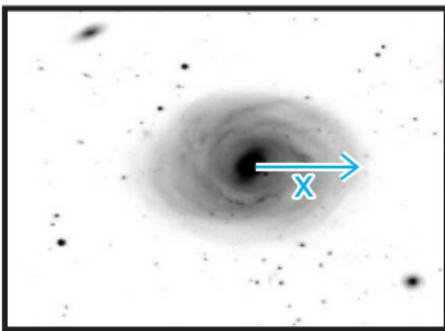
- A physicist runs an experiment.
- She measures two quantities x and y .



Example: Find formula for radial brightness of a galaxy. (Just to illustrate the idea, the formulas and data illustrated in the following are **unrelated** to this example)

Symbolic Regression: Why?

- A physicist runs an experiment.
- She measures two quantities x and y .
- She obtains a set S of n measurements $S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$

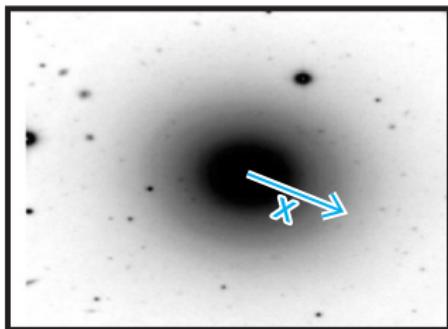
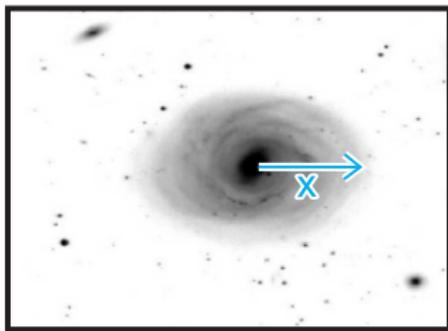


Example: Find formula for radial brightness of a galaxy. (Just to illustrate the idea, the formulas and data illustrated in the following are **unrelated** to this example)

Symbolic Regression: Why?

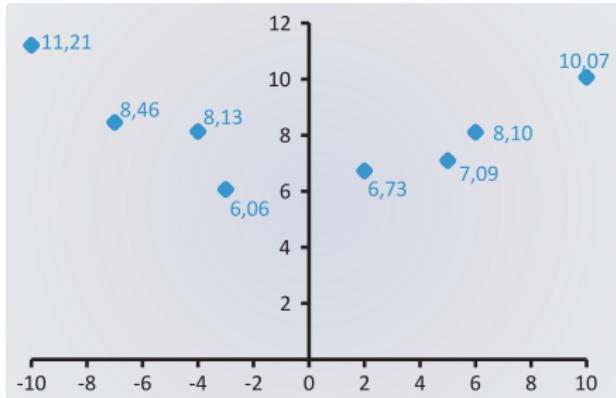
- A physicist runs an experiment.
- She measures two quantities \mathbf{x} and \mathbf{y} .
- She obtains a set S of n measurements $S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$
- Her goal is to find how they are related, i.e., to discover a function $\varphi(\mathbf{x}) = \mathbf{y}$

Example: Find formula for radial brightness of a galaxy. (Just to illustrate the idea, the formulas and data illustrated in the following are **unrelated** to this example)



Symbolic Regression: Why?

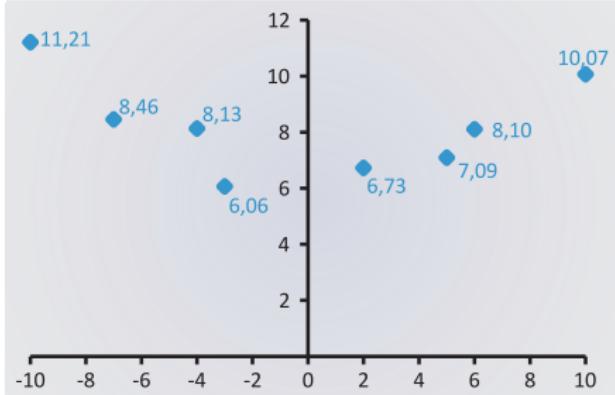
- Given: Set $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$ of points



x	y
-10	11.20975399
-7	8.455665097
-4	8.131449992
-3	6.064537515
2	6.725218415
5	7.091508928
6	8.104694856
10	10.06724264

Symbolic Regression: Why?

- Given: Set $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$ of points
- Wanted: functional relation $\varphi : \mathbb{R} \mapsto \mathbb{R}$



x	y
-10	11.20975399
-7	8.455665097
-4	8.131449992
-3	6.064537515
2	6.725218415
5	7.091508928
6	8.104694856
10	10.06724264

- Approximations: linear and non-linear Regression

- Approximations: linear and non-linear Regression
- Parameterization of a given formula blueprint

- Approximations: linear and non-linear Regression
- Parameterization of a given formula blueprint, e.g.,:

$$\textcolor{red}{y} = \varphi(\textcolor{blue}{x}) = a_4 \textcolor{blue}{x}^4 + a_3 \textcolor{blue}{x}^3 + a_2 \textcolor{blue}{x}^2 + a_1 \textcolor{blue}{x}^1 + a_0 \quad (1)$$

- Approximations: linear and non-linear Reg
- Parameterization of a given formula blue,

$$\mathbf{y} = \varphi(\mathbf{x}) = a_4 \mathbf{x}^4 + a_3 \mathbf{x}^3 + a_2 \mathbf{x}^2 + a_1 \cdot$$



- Method of least squares developed by Gauß [126–128] in the 1790s at the age of 18 and first published by Legendre [129]



- Approximations: linear and non-linear Reg
- Parameterization of a given formula blue,

$$\textcolor{green}{y} = \varphi(\textcolor{blue}{x}) = a_4 \textcolor{blue}{x}^4 + a_3 \textcolor{blue}{x}^3 + a_2 \textcolor{blue}{x}^2 + a_1 \cdot$$

- Method of least squares developed by Gauß [126–128] in the 1790s at the age of 18 and first published by Legendre [129]:

Find values for a_0 to a_4 that minimize $\sum_{i=1}^n (\textcolor{green}{y}_i - \varphi(\textcolor{blue}{x}_i))^2$ (2)



- Approximations: linear and non-linear Reg
- Parameterization of a given formula blue,

$$\textcolor{green}{y} = \varphi(\textcolor{blue}{x}) = a_4 \textcolor{blue}{x}^4 + a_3 \textcolor{blue}{x}^3 + a_2 \textcolor{blue}{x}^2 + a_1 \cdot$$

- Method of least squares developed by Gauß [126–128] in the 1790s at the age of 18 and first published by Legendre [129]:

Find values for a_0 to a_4 that minimize $\sum_{i=1}^n (\textcolor{green}{y}_i - \varphi(\textcolor{blue}{x}_i))^2$ (2)

- Often, we can do this efficiently with the Levenberg-Marquardt [130, 131] algorithm, the standard for non-linear regression

- Approximations: linear and non-linear Reg
- Parameterization of a given formula blue,

$$\textcolor{green}{y} = \varphi(\textcolor{blue}{x}) = a_4 \textcolor{blue}{x}^4 + a_3 \textcolor{blue}{x}^3 + a_2 \textcolor{blue}{x}^2 + a_1 \cdot$$



- Method of least squares developed by Gauß [126–128] in the 1790s at the age of 18 and first published by Legendre [129]:

Find values for a_0 to a_4 that minimize $\sum_{i=1}^n (\textcolor{green}{y}_i - \varphi(\textcolor{blue}{x}_i))^2$ (2)

- Often, we can do this efficiently with the Levenberg-Marquardt [130, 131] algorithm, the standard for non-linear regression
- If that does not work well, we can combine these methods with DE or CMA-ES...

- Approximations: linear and non-linear Reg
- Parameterization of a given formula blueprint

$$\mathbf{y} = \varphi(\mathbf{x}) = a_4 \mathbf{x}^4 + a_3 \mathbf{x}^3 + a_2 \mathbf{x}^2 + a_1 \cdot$$



- Method of least squares developed by Gauß [126–128] in the 1790s at the age of 18 and first published by Legendre [129]:

Find values for a_0 to a_4 that minimize $\sum_{i=1}^n (\mathbf{y}_i - \varphi(\mathbf{x}_i))^2$ (2)

- Often, we can do this efficiently with the Levenberg-Marquardt [130, 131] algorithm, the standard for non-linear regression
- If that does not work well, we can combine these methods with DE or CMA-ES...
- But: Assumption about formula blueprint needed...

- New kind of optimization problem: Symbolic Regression

Symbolic Regression: Idea



- New kind of optimization problem: Symbolic Regression
- We have:

- New kind of optimization problem: Symbolic Regression
- We have:
 - Given set of points $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$

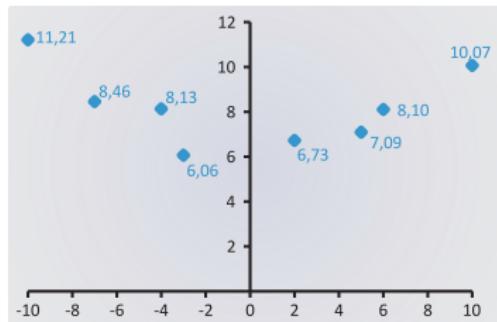
- New kind of optimization problem: Symbolic Regression
- We have:
 - Given set of points $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$
 - A Function Set with elementary functions which can be combined arbitrarily: $\mathbf{F} = \{+, -, *, \sin, \sqrt{}, \exp, \dots\}$

- New kind of optimization problem: Symbolic Regression
- We have:
 - Given set of points $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$
 - A Function Set with elementary functions which can be combined arbitrarily: $\mathbf{F} = \{+, -, *, \sin, \sqrt{}, \exp, \dots\}$
 - A Terminal Set, i.e., the available null-ary functions containing things like the input variable \mathbf{x} and real constants: $\mathbf{T} = \{\mathbf{x}, \mathfrak{R}, \dots\}$

- New kind of optimization problem: Symbolic Regression
- We have:
 - Given set of points $S = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$
 - A Function Set with elementary functions which can be combined arbitrarily: $\mathbf{F} = \{+, -, *, \sin, \sqrt{}, \exp, \dots\}$
 - A Terminal Set, i.e., the available null-ary functions containing things like the input variable \mathbf{x} and real constants: $\mathbf{T} = \{\mathbf{x}, \mathfrak{R}, \dots\}$
- We want: Find a combination of elements from \mathbf{F} and \mathbf{T} which represents a formula that fits to the data.

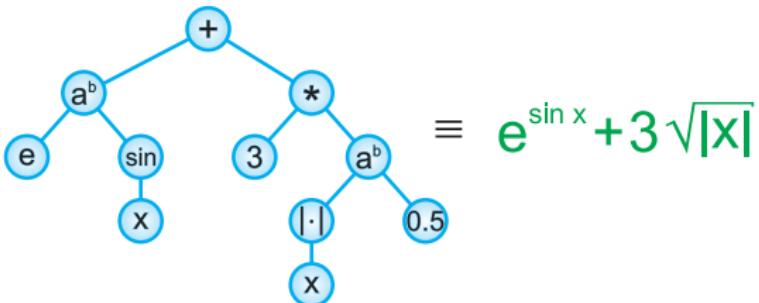
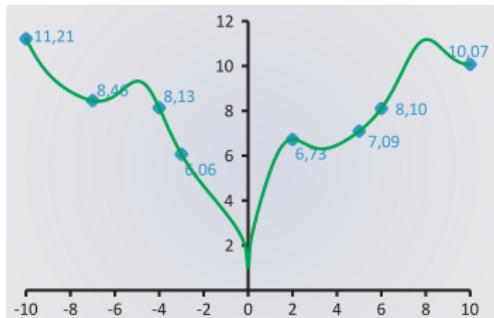
Symbolic Regression: Representation

- Symbolic Regression with Genetic Programming



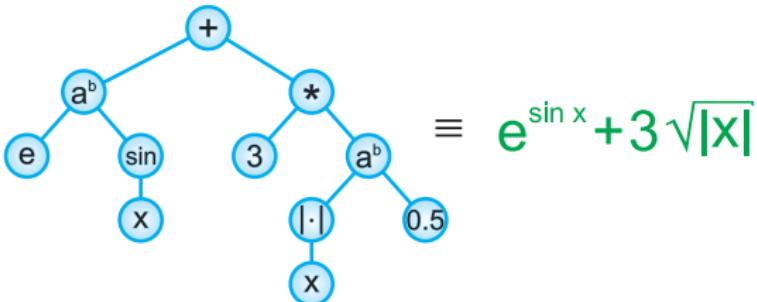
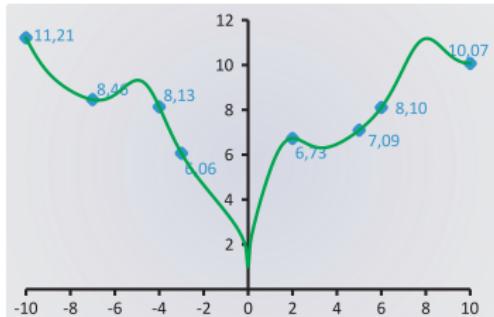
Symbolic Regression: Representation

- Symbolic Regression with Genetic Programming:
 - represent formulas as tree data structures



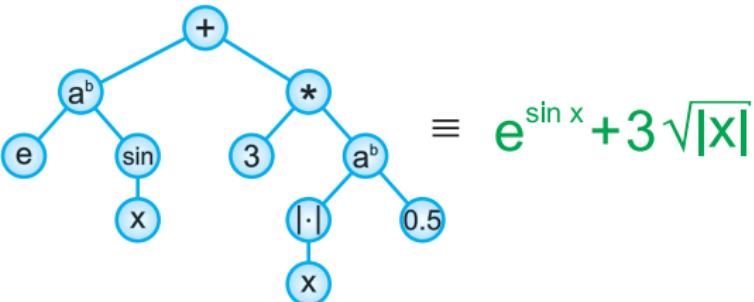
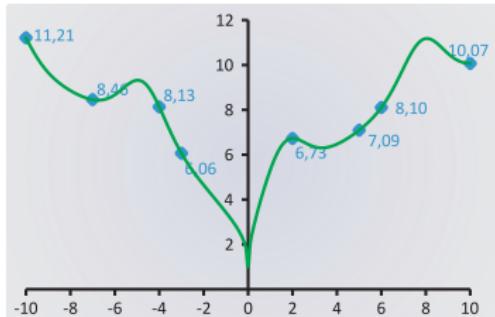
Symbolic Regression: Representation

- Symbolic Regression with Genetic Programming:
 - represent formulas as tree data structures
 - for example: minimize $f(\varphi) = \sum_{i=1}^n (\mathbf{y}_i - \varphi(\mathbf{x}_i))^2$



Symbolic Regression: Representation

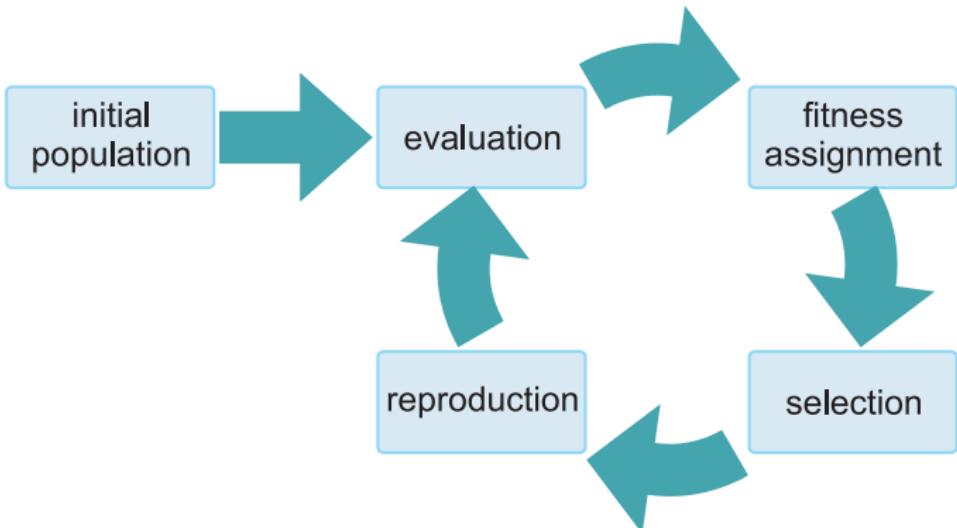
- Symbolic Regression with Genetic Programming:
 - represent formulas as tree data structures
 - for example: minimize $f(\varphi) = \sum_{i=1}^n (\mathbf{y}_i - \varphi(\mathbf{x}_i))^2$
 - construct φ with Genetic Programming!



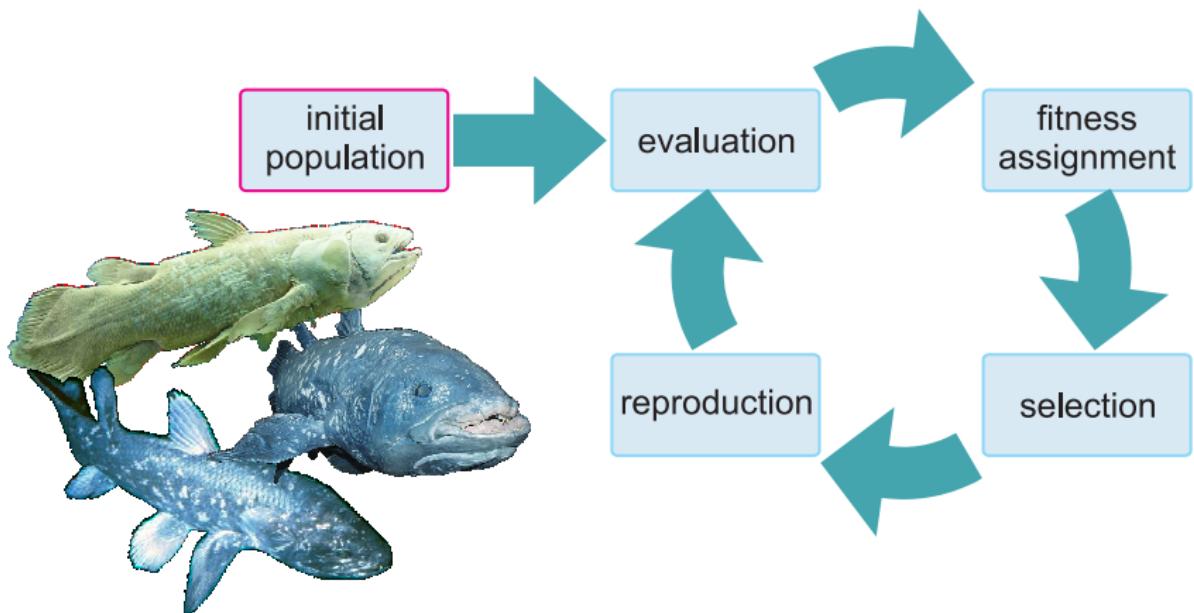
Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

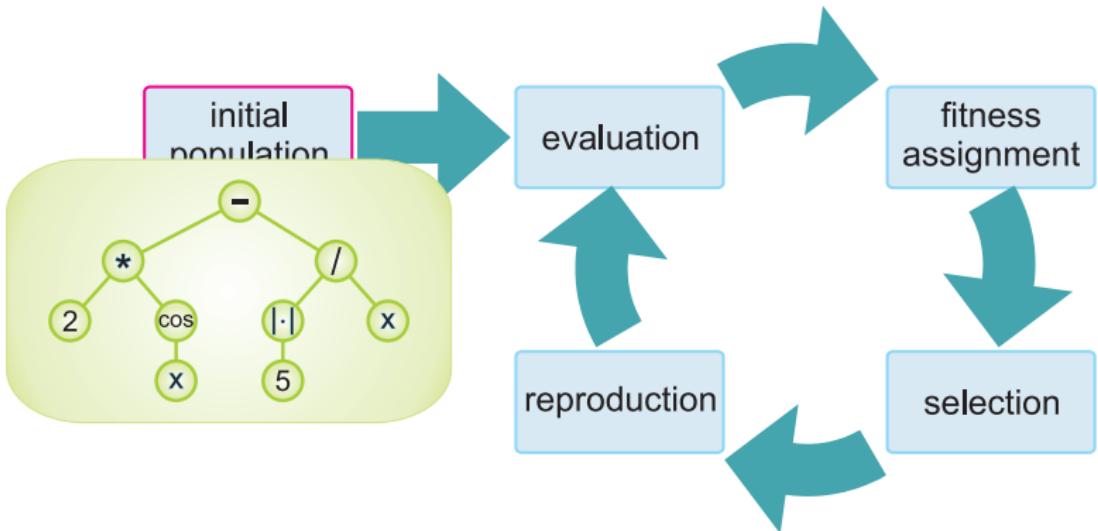
- Population-based optimization according to the cycle below



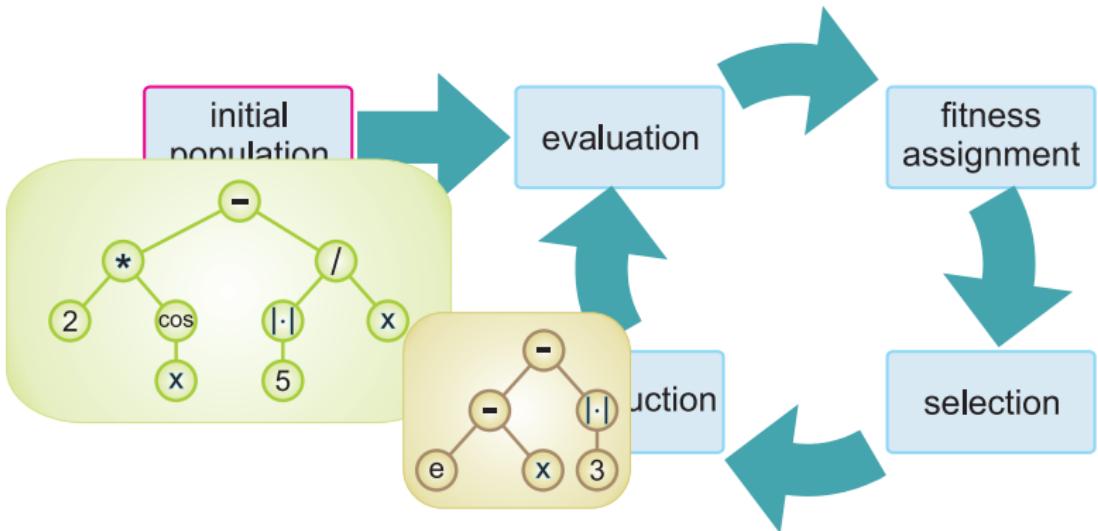
- Start with a random set of individuals



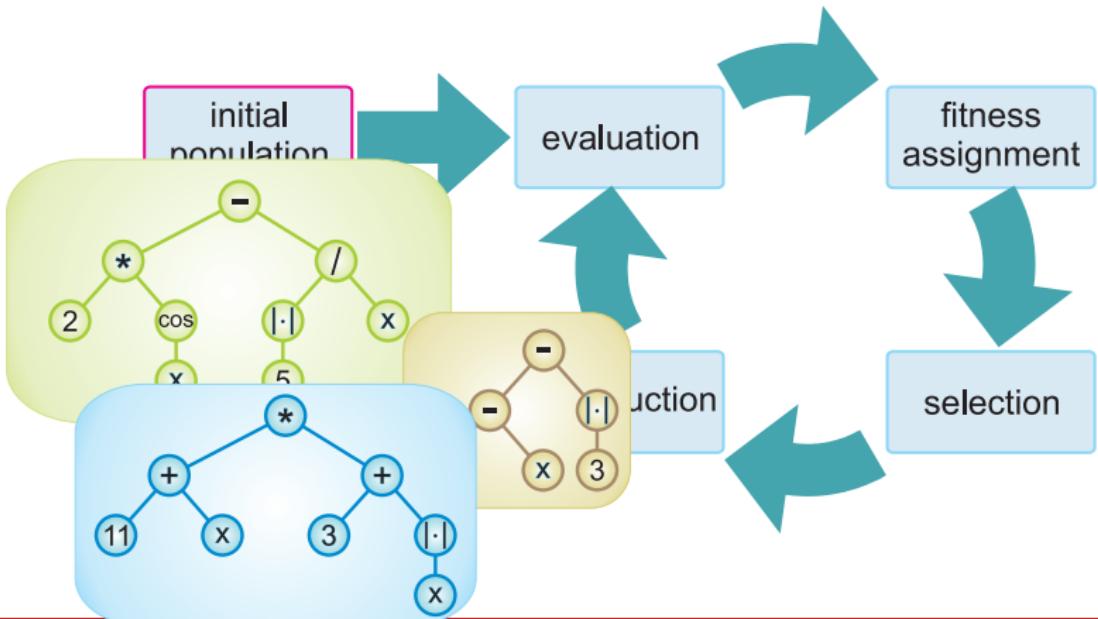
- Start with a random set of programs



- Start with a random set of programs

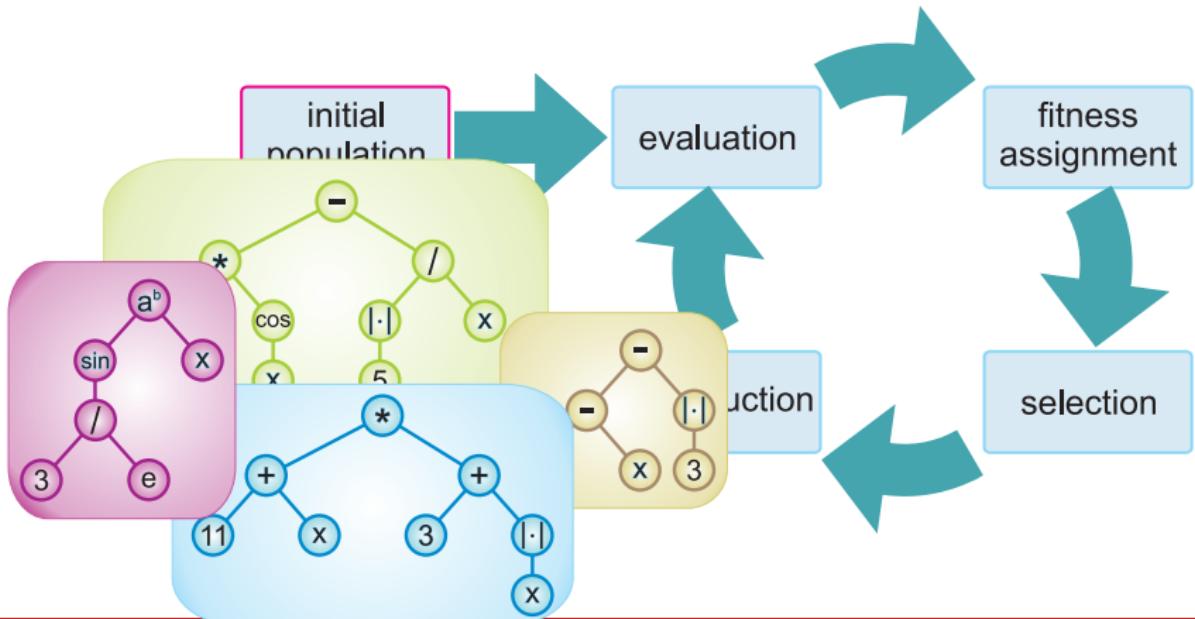


- Start with a random set of programs

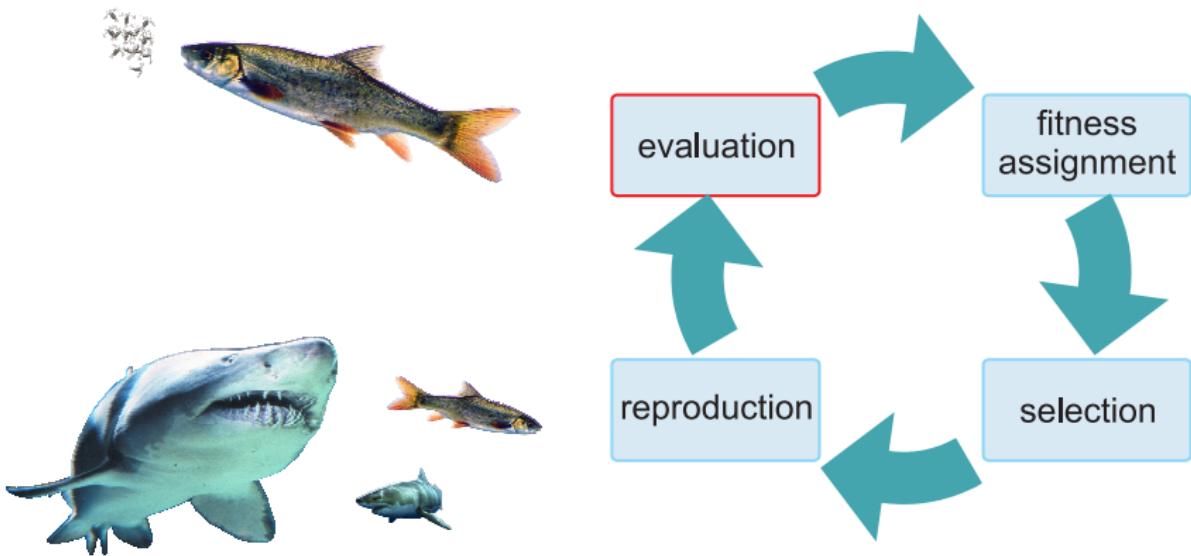


Genetic Programming

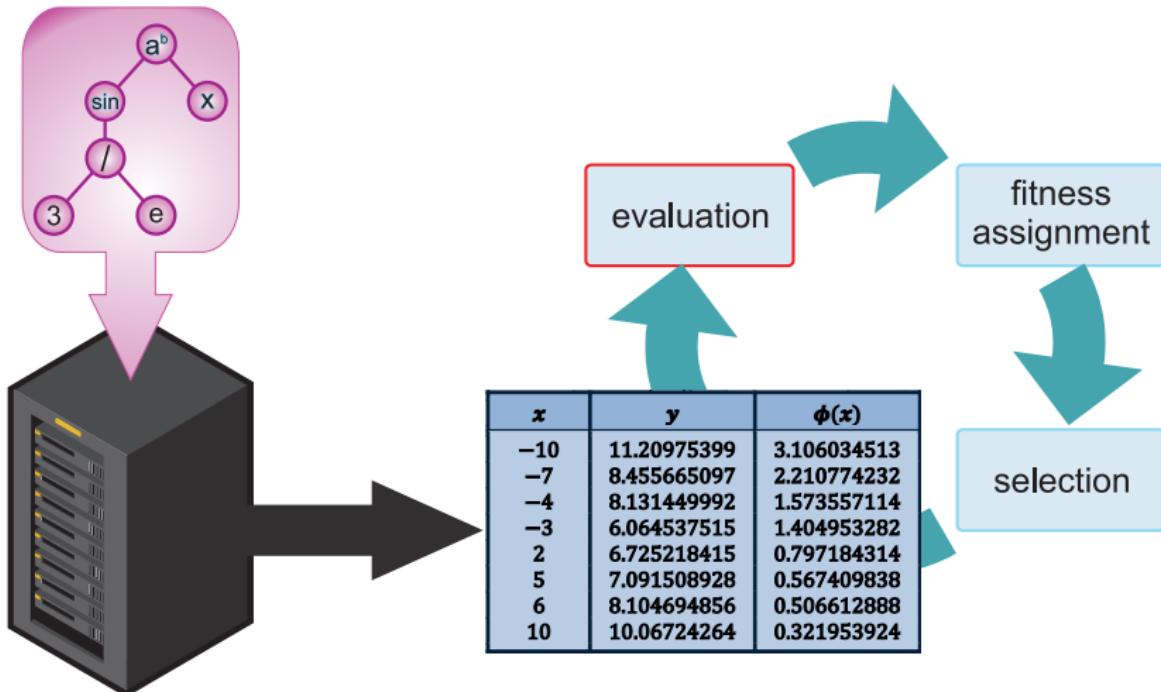
- Start with a random set of programs



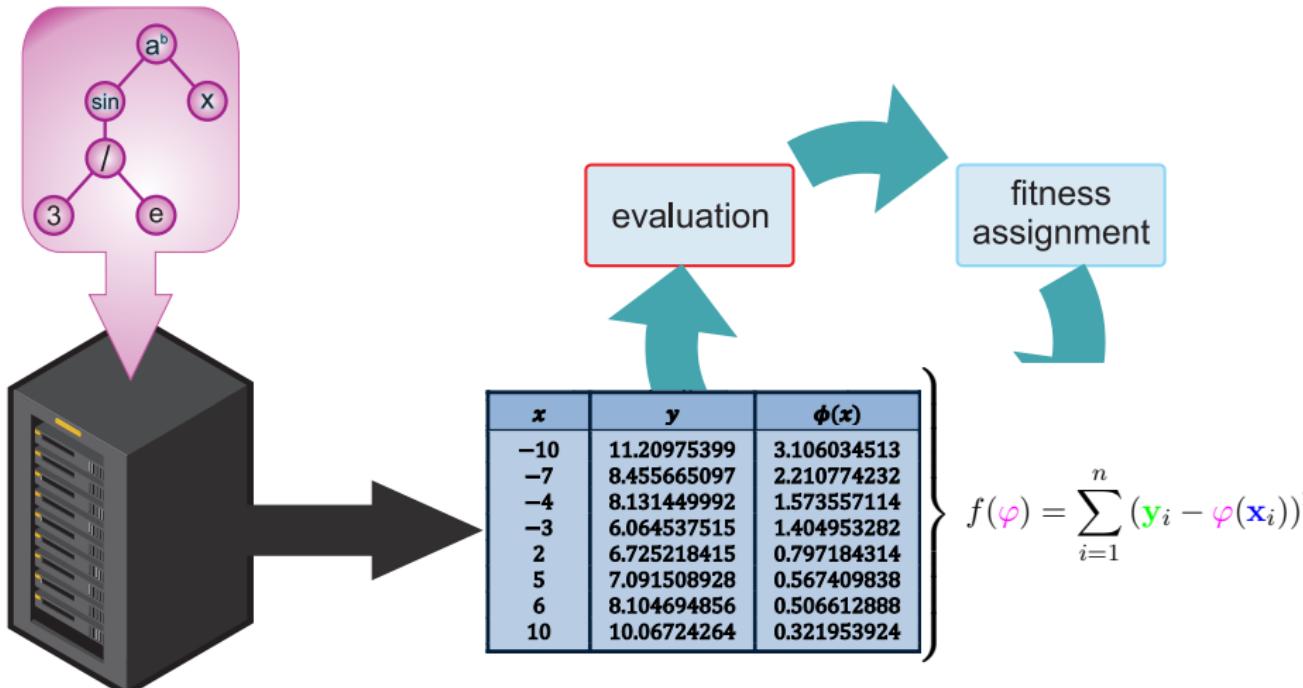
- Test the features of each individual



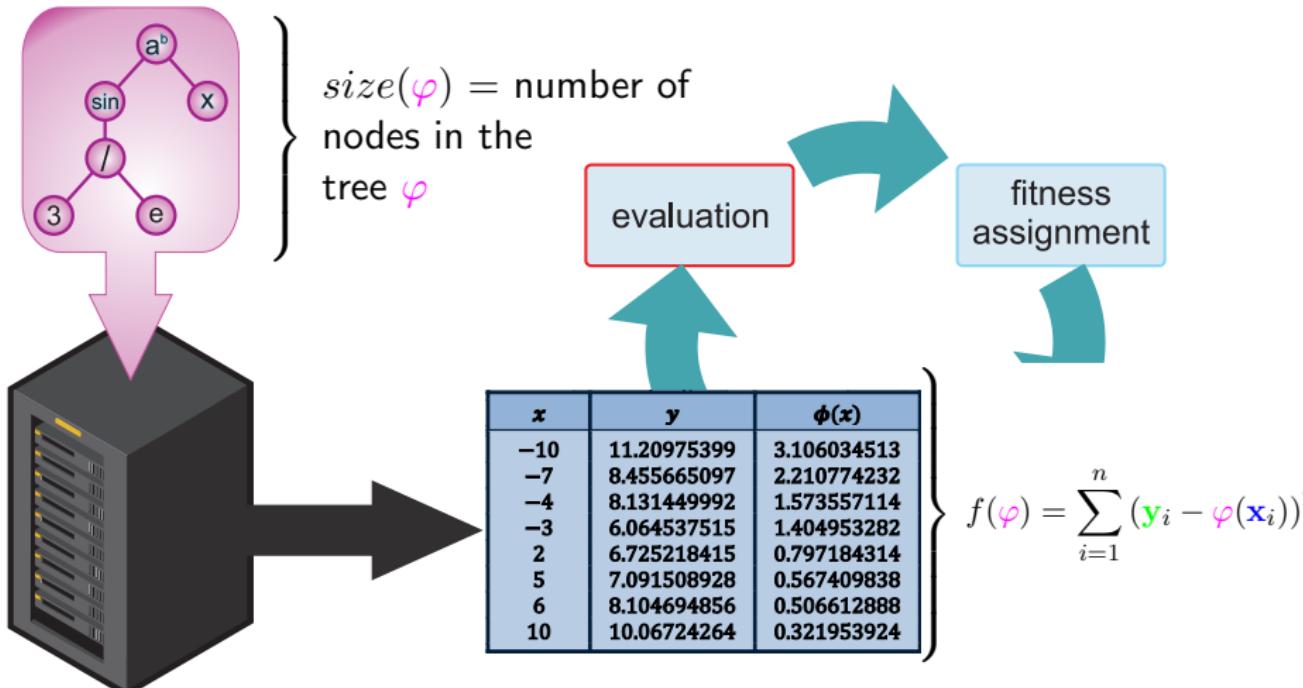
- Performance in simulation/evaluation plus criteria like size



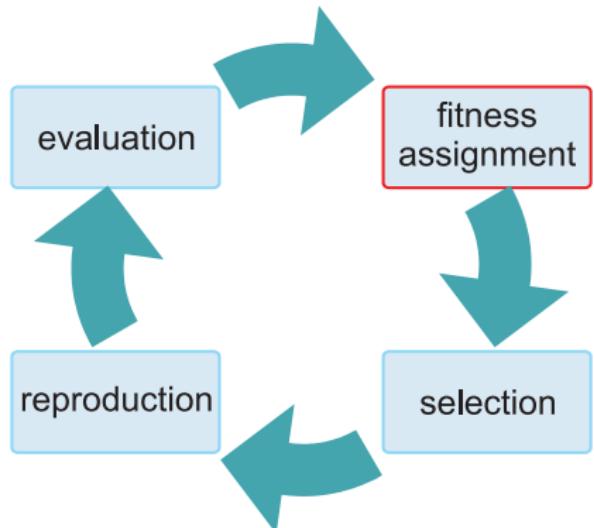
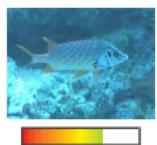
- Performance in simulation/evaluation plus criteria like size



- Performance in simulation/evaluation plus criteria like size

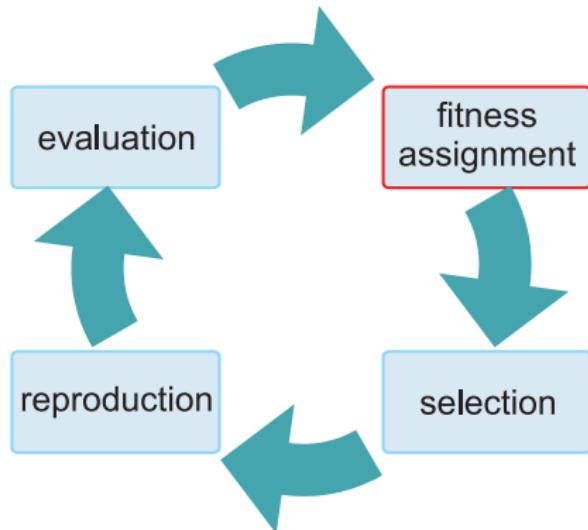


- Fitness is relative, determined/defined as number of offspring



- Fitness can be relative, determines expected number of offsprings

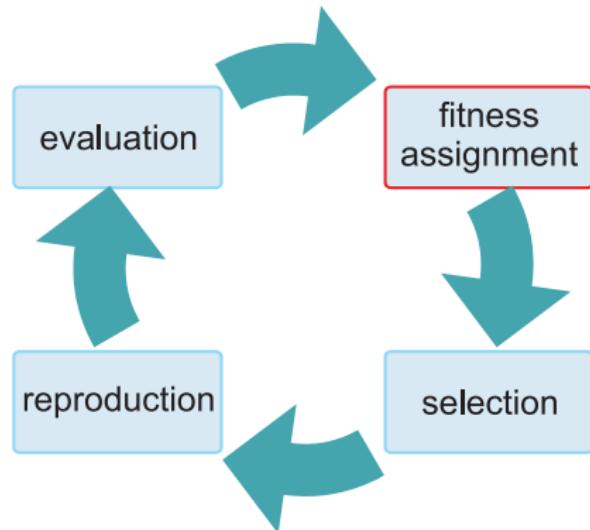
Fitness $\nu(\varphi)$ in Symbolic Regression could be



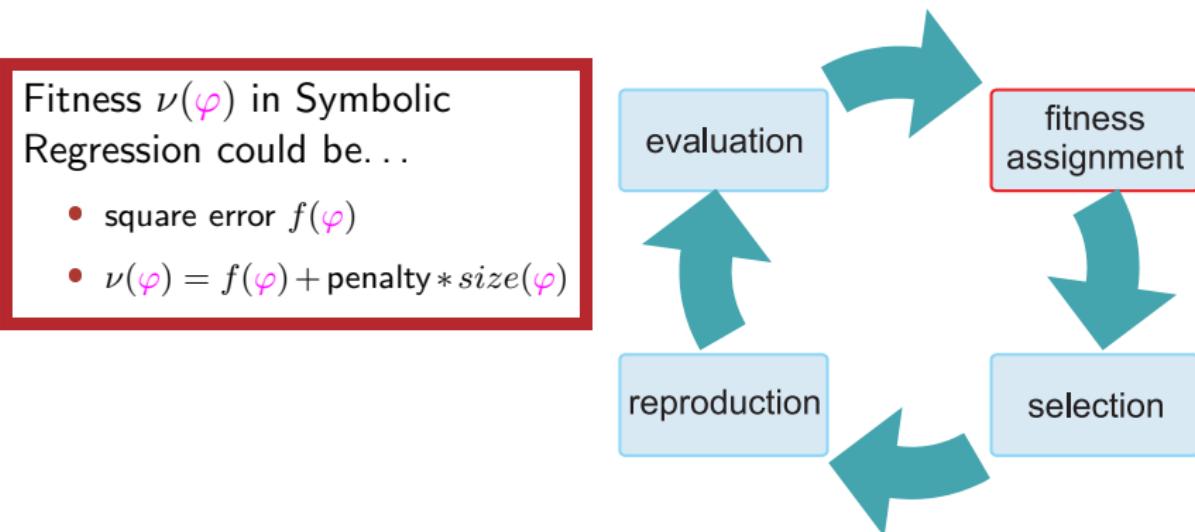
- Fitness can be relative, determines expected number of offsprings

Fitness $\nu(\varphi)$ in Symbolic Regression could be...

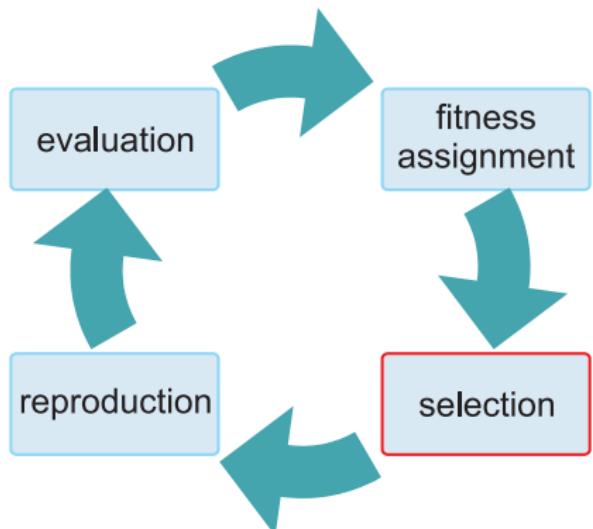
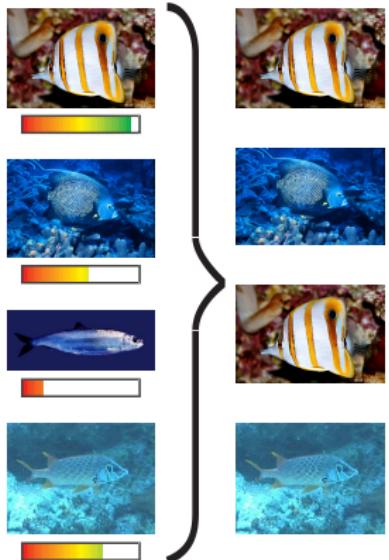
- square error $f(\varphi)$



- Fitness can be relative, determines expected number of offsprings

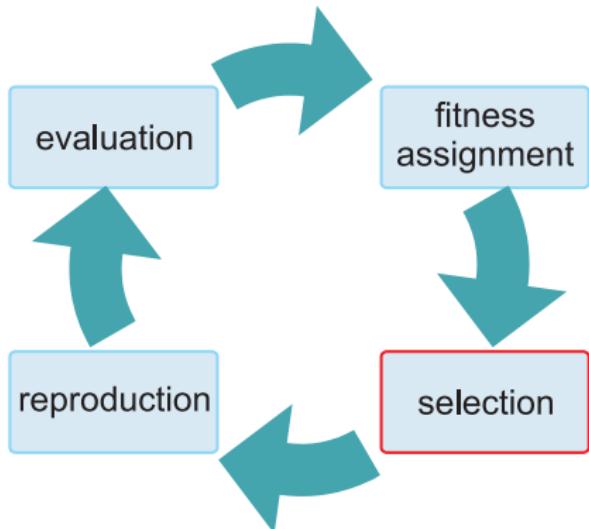


- Fitter individuals have more offspring

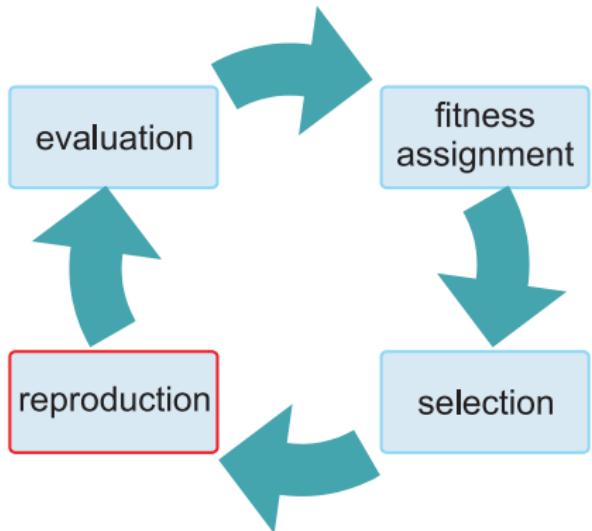


- Selection determines number of offspring based on fitness

Selection in GP usually either fitness proportionate or tournament selection.

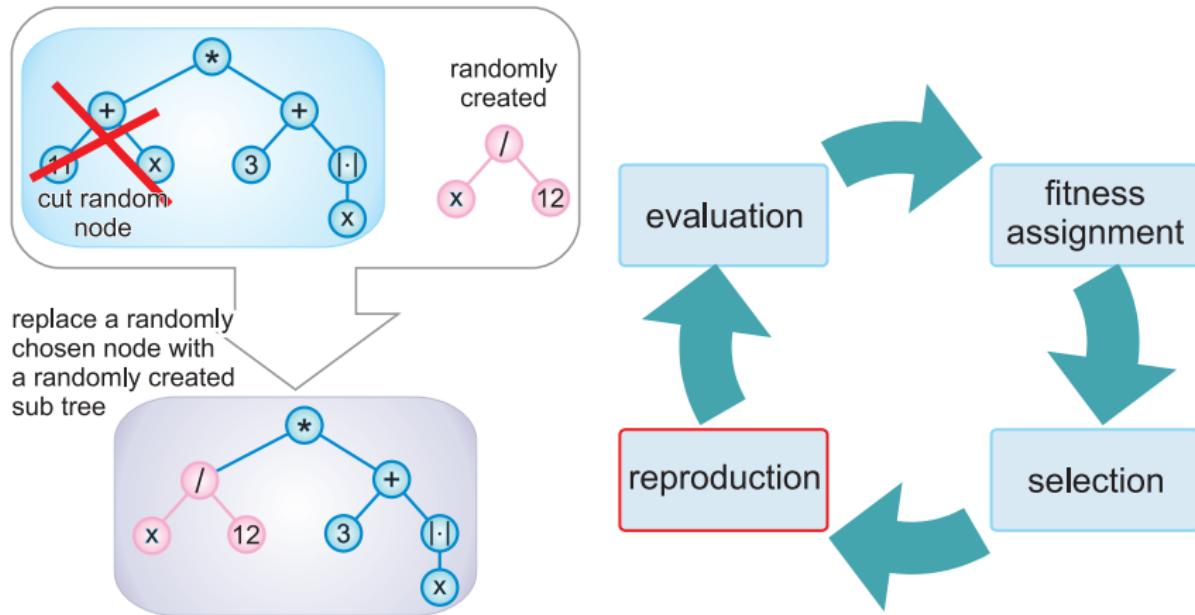


- Asexual and sexual reproduction

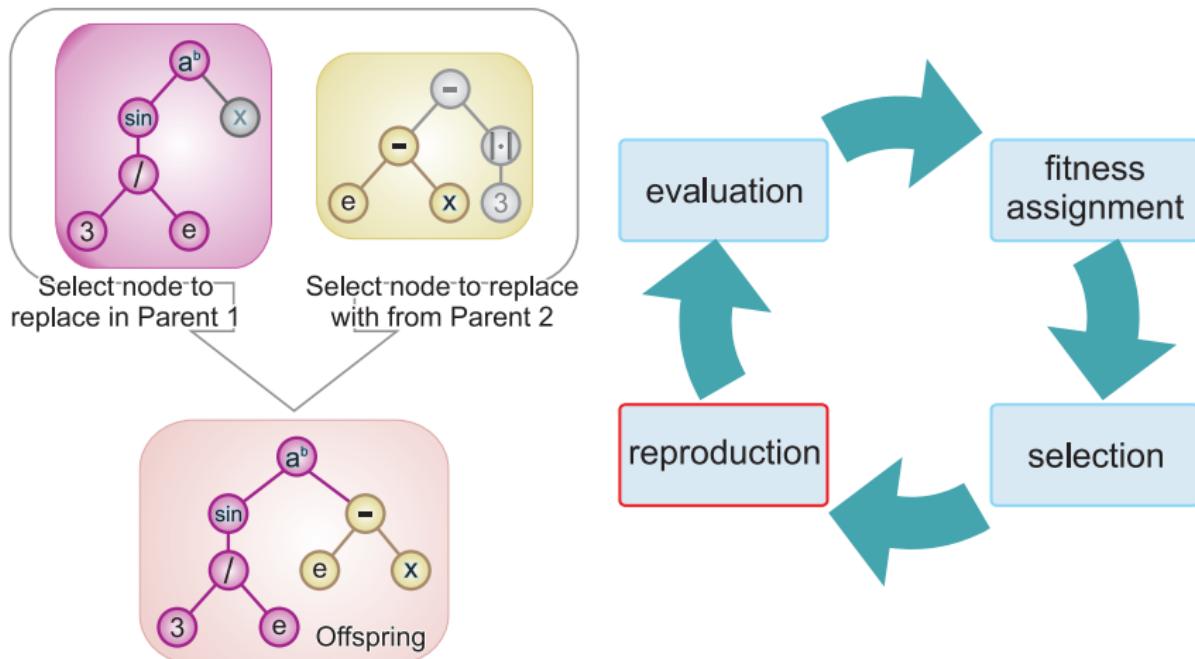


Genetic Programming

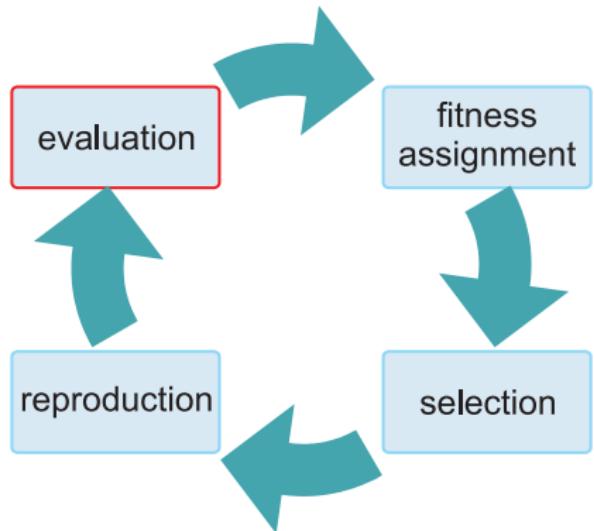
- Asexual reproduction: mutation (insert random sub-tree)



- Sexual reproduction: recombination (exchange sub trees)



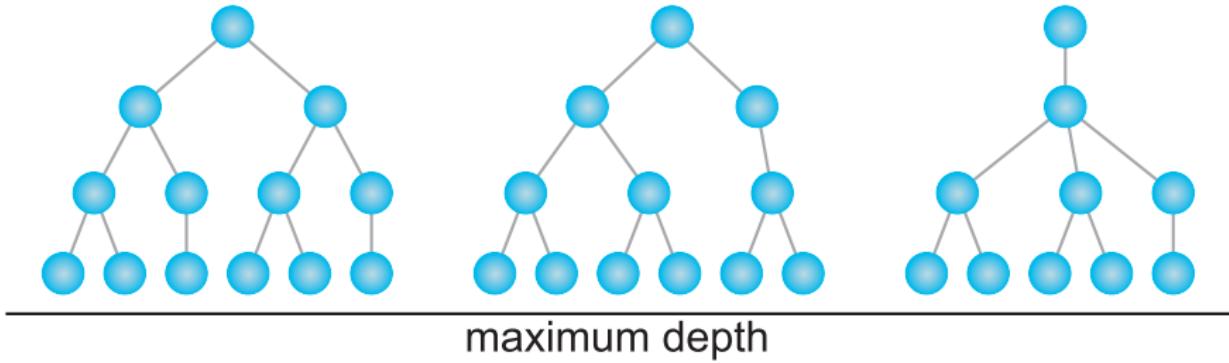
- Cycle starts all over until termination criterion is met



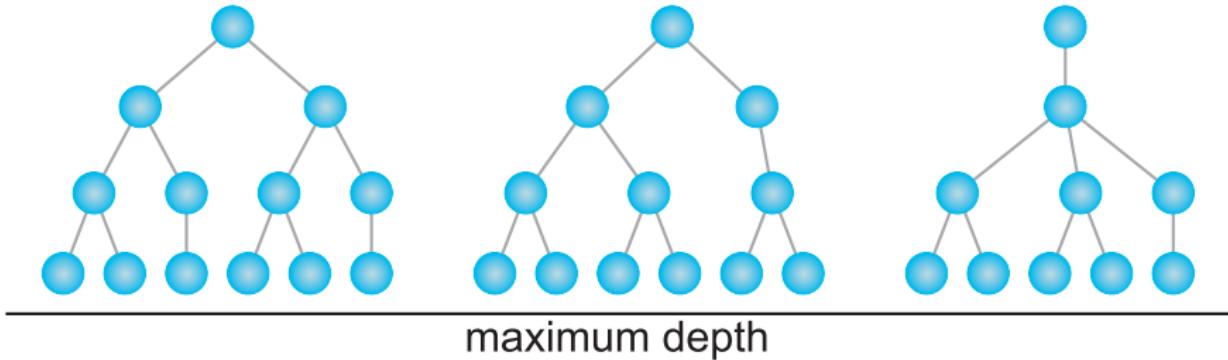
Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

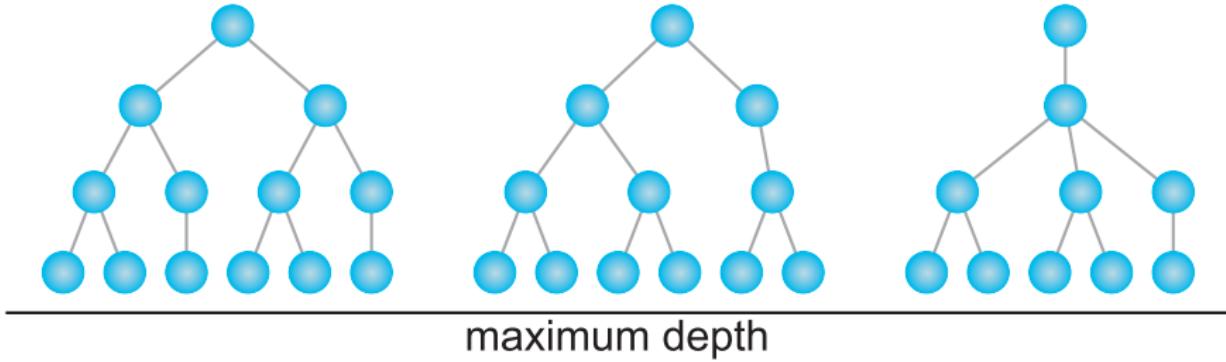
- Creation of random programs of a maximum depth maxDepth : [3]



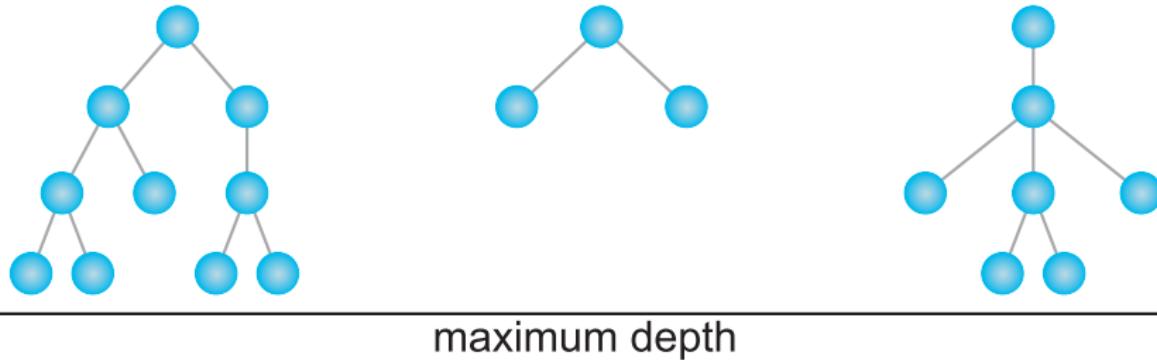
- Creation of random programs of a maximum depth maxDepth : [3]
- **Method A:** $\text{Full}(\text{maxDepth})$
 - ① Add random **function nodes** until $\text{maxDepth} - 1$ is reached in all branches



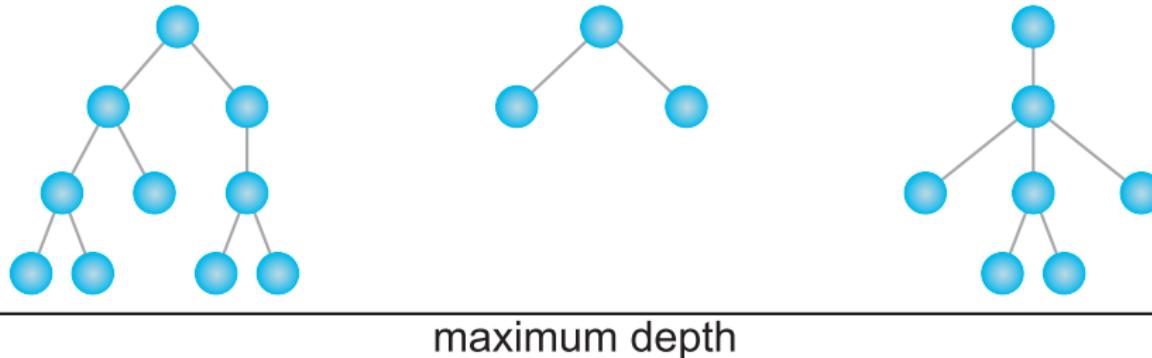
- Creation of random programs of a maximum depth maxDepth : [3]
- **Method A:** $\text{Full}(\text{maxDepth})$
 - ① Add random **function nodes** until $\text{maxDepth} - 1$ is reached in all branches
 - ② Add terminal nodes



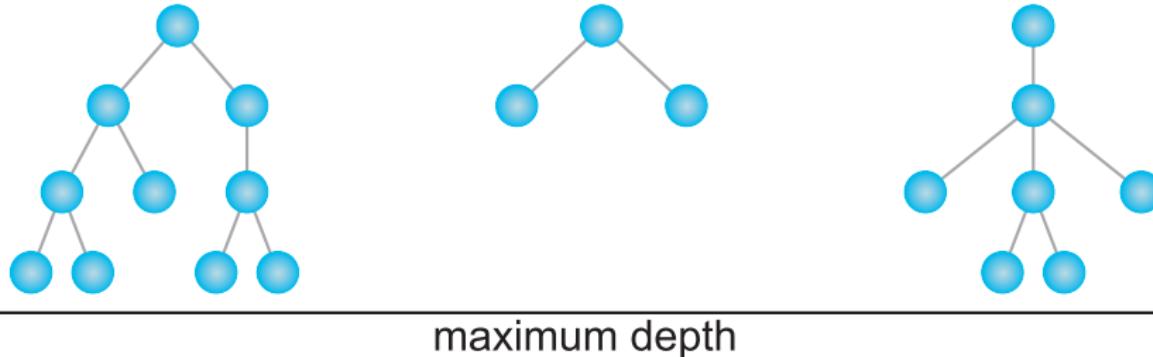
- Creation of random programs of a maximum depth maxDepth : [3]
- **Method B:** $\text{Grow}(\text{maxDepth})$



- Creation of random programs of a maximum depth maxDepth : [3]
- **Method B:** $\text{Grow}(\text{maxDepth})$
 - ① Add random **function or terminal nodes** until all branches have terminals or reached $\text{maxDepth} - 1$



- Creation of random programs of a maximum depth maxDepth : [3]
- **Method B:** $\text{Grow}(\text{maxDepth})$
 - ① Add random **function or terminal nodes** until all branches have terminals or reached $\text{maxDepth} - 1$
 - ② Add terminal nodes to all branches without terminals



- Creation of random programs of a maximum depth maxDepth : [3]
- **Method C:** Ramped-Half-and-Half

- Creation of random programs of a maximum depth maxDepth : [3]
- **Method C: Ramped-Half-and-Half**
 - ① Create $\approx \frac{ps}{2(\text{maxDepth}-1)}$ with (i) for $\forall i \in 2 \dots \text{maxDepth}$
 - ② Create $\approx \frac{ps}{2(\text{maxDepth}-1)}$ with (i) for $\forall i \in 2 \dots \text{maxDepth}$

- Creation of random programs of a maximum depth maxDepth : [3]
- **Method C: Ramped-Half-and-Half**
 - ① Create $\approx \frac{ps}{2(\text{maxDepth}-1)}$ with (i) for $\forall i \in 2 \dots \text{maxDepth}$
 - ② Create $\approx \frac{ps}{2(\text{maxDepth}-1)}$ with (i) for $\forall i \in 2 \dots \text{maxDepth}$
- Fills population with a good variety of trees of different shapes and sizes

Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

Data Mining: Classify Irises



Iris Setosa

petal

sepal



Iris Versicolor



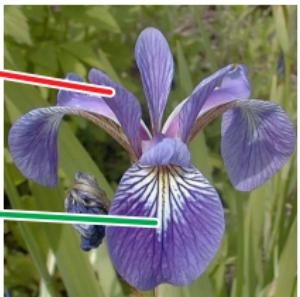
Iris Virginica

- Most classical example for data mining [132, 133]

Data Mining: Classify Irises



Iris Setosa



Iris Versicolor

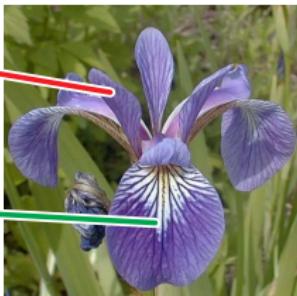


Iris Virginica

- Most classical example for data mining [132, 133]
- The petals and sepals of different iris flowers have been measured



Iris Setosa



Iris Versicolor



Iris Virginica

- Most classical example for data mining [132, 133]
- The petals and sepals of different iris flowers have been measured
- Can we use this data to find a program which tells us to what type a flower belongs on basis of petal and sepal measurements?

Data Mining: Classify Irises



- Data samples $t = (t_1, t_2, \dots, t_n); t_i \in \mathbb{R}$ belong to classes $k \in K$

A { $t \in A \{$

Sepal Length	Sepal Width	Petal Length	Petal Width	Class
5.1	3.5	1.4	0.2	iris setosa
4.9	3.0	1.4	0.2	iris setosa
7.0	3.2	4.7	1.4	iris versicolor
6.3	3.3	6.0	2.5	iris virginica
...
6.4	3.2	4.5	1.4	iris versicolor

$t_1 \in \mathbb{R}$ $t_2 \in \mathbb{R}$ $t_3 \in \mathbb{R}$ $t_4 \in \mathbb{R}$ $K = \{\text{setosa, versicolor, virginica}\}$

$k = \text{class}(t)$

Data Mining: Classify Irises

- Data samples $t = (t_1, t_2, \dots, t_n); t_i \in \mathbb{R}$ belong to classes $k \in K$
- Supervised learning: we use samples $t \in A$ with known classes $class(t) \in K$ to learn a function $C : \mathbb{R}^n \mapsto K$

A { $t \in A \{$

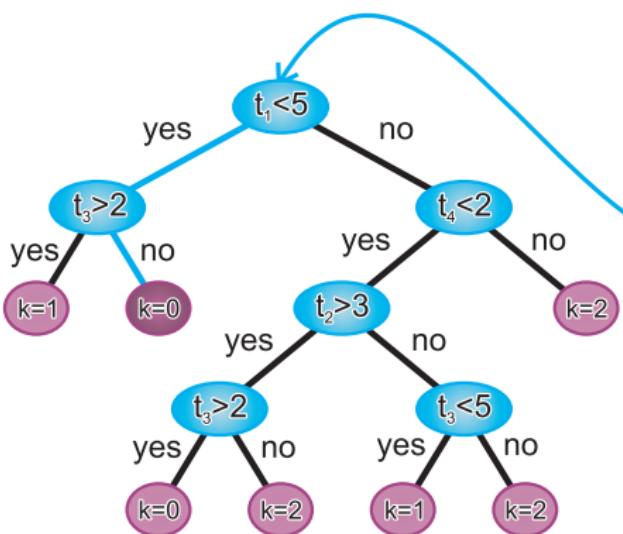
Sepal Length	Sepal Width	Petal Length	Petal Width	Class
5.1	3.5	1.4	0.2	iris setosa
4.9	3.0	1.4	0.2	iris setosa
7.0	3.2	4.7	1.4	iris versicolor
6.3	3.3	6.0	2.5	iris virginica
...
6.4	3.2	4.5	1.4	iris versicolor

$t_1 \in \mathbb{R}$ $t_2 \in \mathbb{R}$ $t_3 \in \mathbb{R}$ $t_4 \in \mathbb{R}$ $K = \{setosa, versicolor, virginica\}$

$k = class(t)$

Classify Irises with Decision Trees

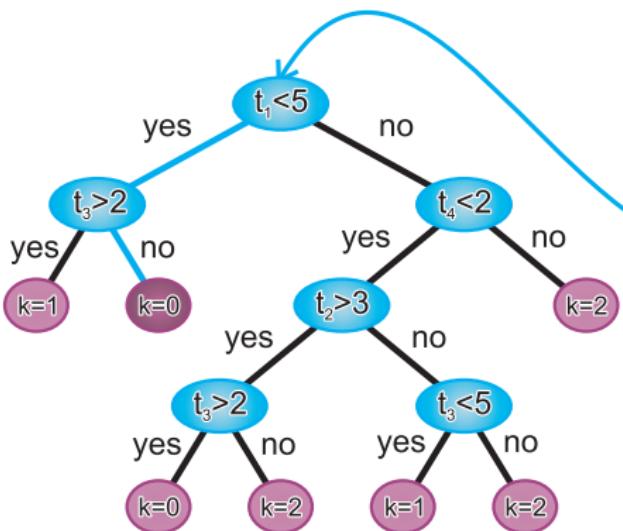
- Very common approach: decision trees



t_1	t_2	t_3	t_4	k
5.1	3.5	1.4	0.2	0
4.9	3.0	1.4	0.2	0
7.0	3.2	4.7	1.4	1
6.3	3.3	6.0	2.5	2
...
6.4	3.2	4.5	1.4	1

Classify Irises with Decision Trees

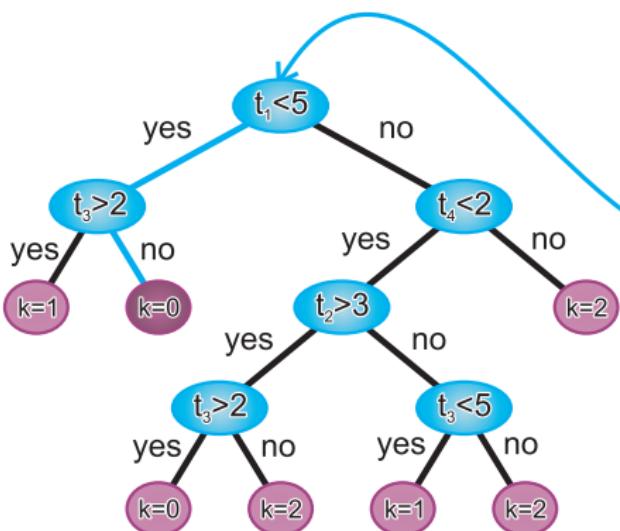
- Very common approach: decision trees
- Classical decision tree builders are limited to simple decision node structures



t_1	t_2	t_3	t_4	k
5.1	3.5	1.4	0.2	0
4.9	3.0	1.4	0.2	0
7.0	3.2	4.7	1.4	1
6.3	3.3	6.0	2.5	2
...
6.4	3.2	4.5	1.4	1

Classify Irises with Decision Trees

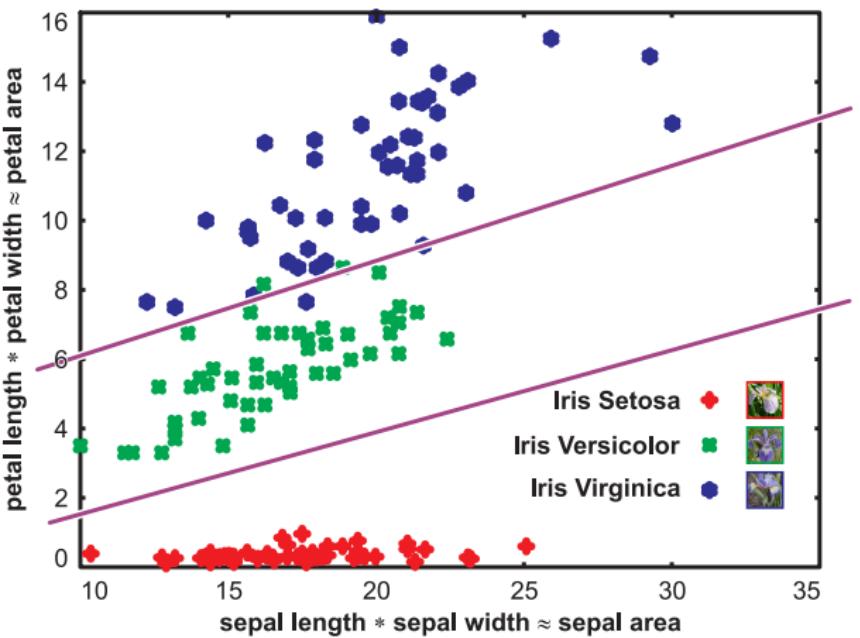
- Very common approach: decision trees
- Classical decision tree builders are limited to simple decision node structures ⇒ Some decisions cannot be represented



t ₁	t ₂	t ₃	t ₄	k
5.1	3.5	1.4	0.2	0
4.9	3.0	1.4	0.2	0
7.0	3.2	4.7	1.4	1
6.3	3.3	6.0	2.5	2
...
6.4	3.2	4.5	1.4	1

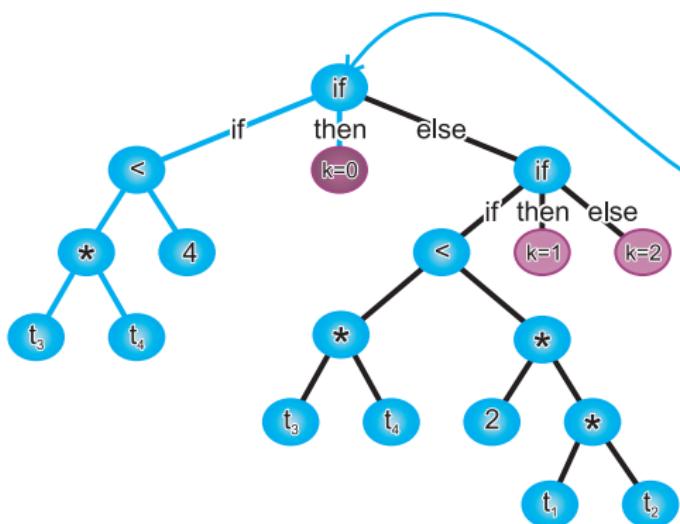
Classify Irises with Decision Trees

- Classical decision tree builders are limited to simple decision node structures \Rightarrow Some decisions cannot be represented



Classify Irises with Genetic Programming

- Genetic Programming: maximize $f(C) = |\{t \in A : C(t) = \text{class}(t)\}|$
- Decisions and tree shapes not limited to a certain shape



t_1	t_2	t_3	t_4	k
5.1	3.5	1.4	0.2	0
4.9	3.0	1.4	0.2	0
7.0	3.2	4.7	1.4	1
6.3	3.3	6.0	2.5	2
...
6.4	3.2	4.5	1.4	1

if $t_3 * t_4 < 4 \Rightarrow \text{class } 0$
else if $t_3 * t_4 < 2 * t_1 * t_2 \Rightarrow \text{class } 1$
else class 2

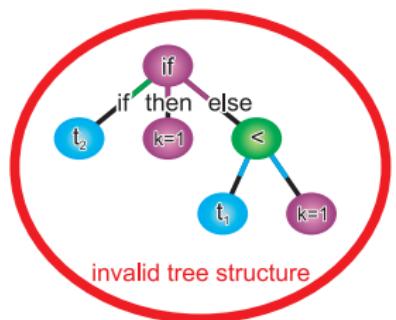
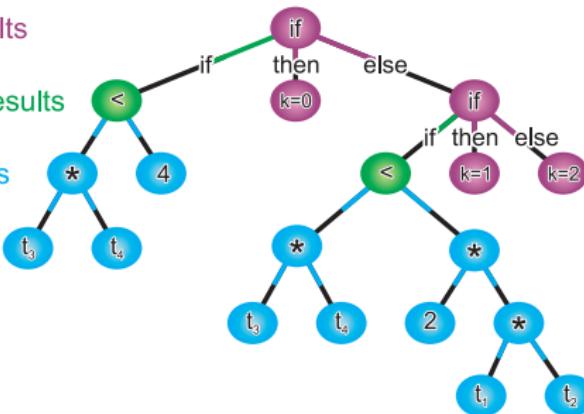
- Standard Genetic Programming (STGP): different node **types** [22, 134–137]

Standard Genetic Programming

class-valued results

boolean-valued results

real-valued results



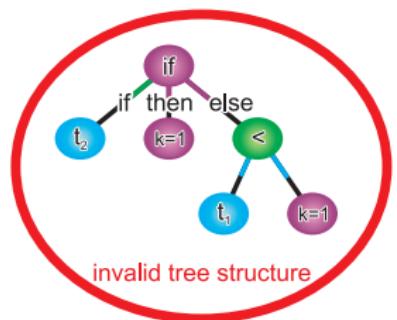
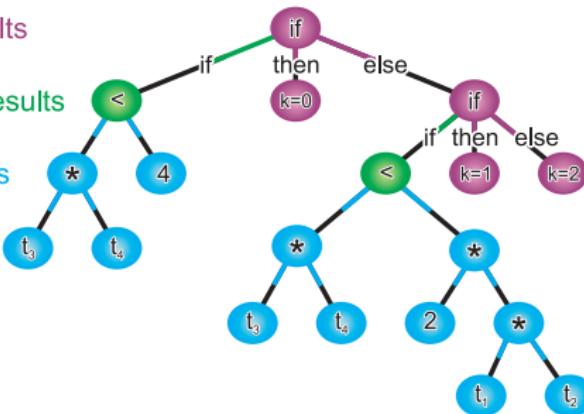
- Standard Genetic Programming (STGP): different node **types** [22, 134–137]

Standard Genetic Programming

class-valued results

boolean-valued results

real-valued results



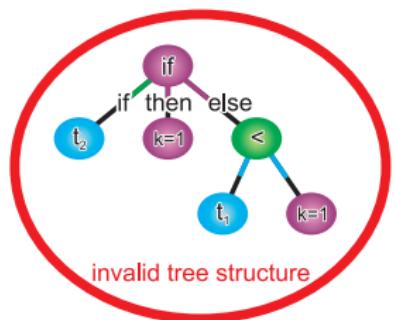
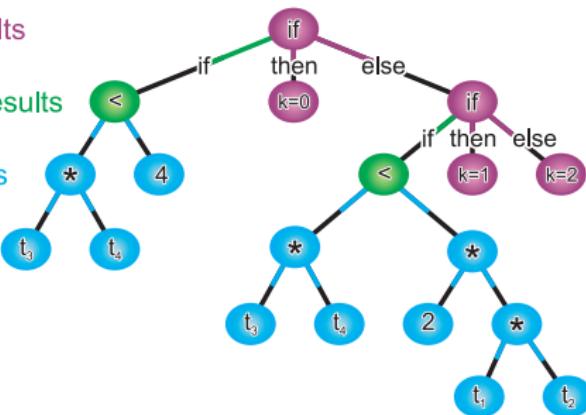
- Standard Genetic Programming (STGP): different node **types** [22, 134–137]
- Non-terminal nodes: type for each child

Standard Genetic Programming

class-valued results

boolean-valued results

real-valued results



invalid tree structure

- Standard Genetic Programming (STGP): different node **types** [22, 134–137]
- Non-terminal nodes: type for each child
- Reproduction operations: must obey to these specifications

Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

Bloat in Genetic Programming



- Tree-representations are of variable size

Bloat in Genetic Programming

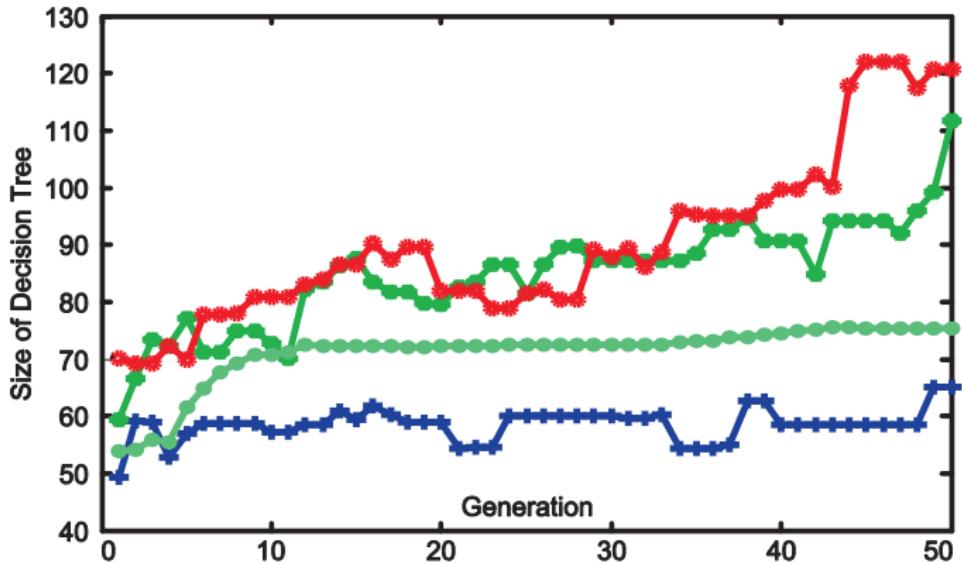


- Tree-representations are of variable size
- There may be big trees, there may be small trees

- Tree-representations are of variable size
- There may be big trees, there may be small trees
- What happens with the tree size during the course of the evolution?

Bloat in Genetic Programming

- Tree-representations are of variable size
- There may be big trees, there may be small trees
- What happens with the tree size during the course of the evolution?



Bloat in Genetic Programming

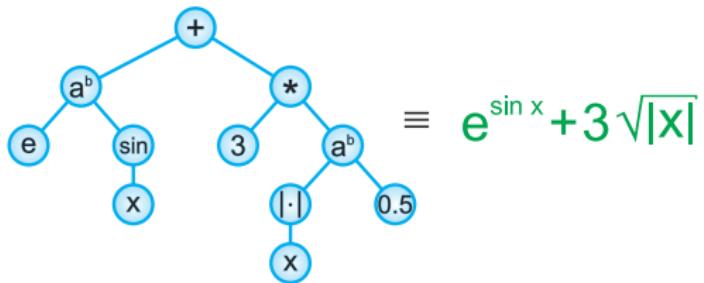


- **Bloat**: uncontrolled growth of programs [138–141]

- **Bloat**: uncontrolled growth of programs [138–141]
- **Intron**: useless part of program, one type of bloat [142]

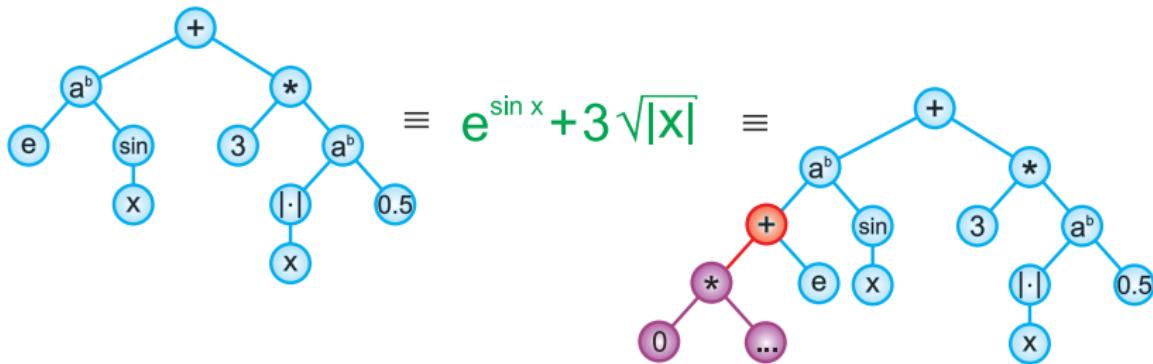
Bloat in Genetic Programming

- **Bloat**: uncontrolled growth of programs [138–141]
- **Intron**: useless part of program, one type of bloat [142]



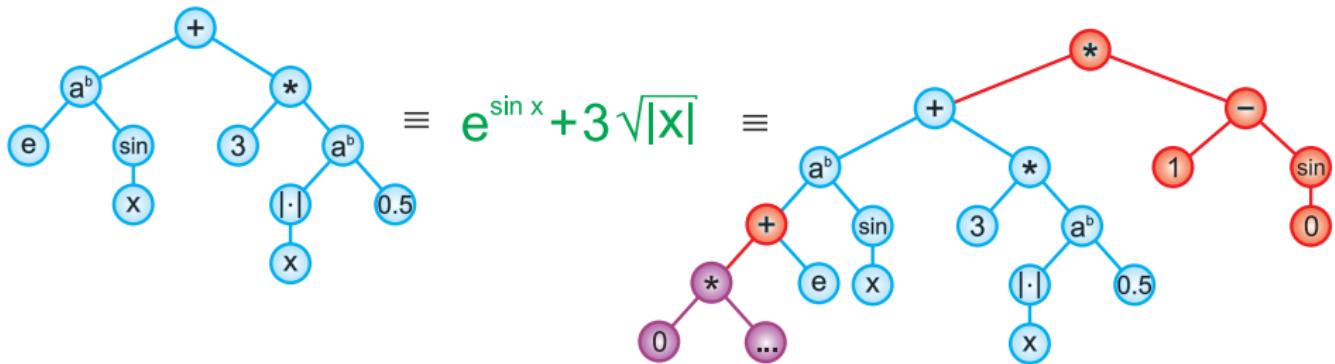
Bloat in Genetic Programming

- **Bloat**: uncontrolled growth of programs [138–141]
- **Intron**: useless part of program, one type of bloat [142]



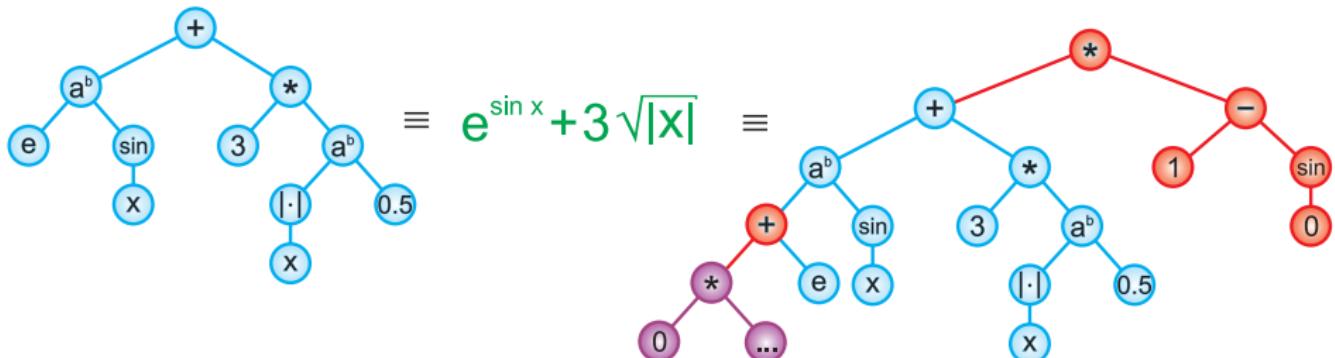
Bloat in Genetic Programming

- **Bloat**: uncontrolled growth of programs [138–141]
- **Intron**: useless part of program, one type of bloat [142]



Bloat in Genetic Programming

- **Bloat**: uncontrolled growth of programs [138–141]
- **Intron**: useless part of program, one type of bloat [142]



- Is it good or bad?

Bloat in Genetic Programming



- **Bloat**: uncontrolled growth of programs
- Is it good or bad?

Bloat in Genetic Programming



- **Bloat**: uncontrolled growth of programs
- Is it good or bad?

- **Bloat**: uncontrolled growth of programs
- Is it good or bad?
 - ① Elegant solutions are always simple and small

- **Bloat**: uncontrolled growth of programs
- Is it good or bad?
 - ① Elegant solutions are always simple and small
 - ② Larger programs = longer processing time for both, reproduction operations and evaluation

- **Bloat**: uncontrolled growth of programs
- Is it good or bad?
 - ① Elegant solutions are always simple and small
 - ② Larger programs = longer processing time for both, reproduction operations and evaluation
 - ③ Larger programs = danger of overfitting

- **Bloat**: uncontrolled growth of programs
- Is it good or bad?
 - ① Elegant solutions are always simple and small
 - ② Larger programs = longer processing time for both, reproduction operations and evaluation
 - ③ Larger programs = danger of overfitting
 - ④ Larger programs occupy more memory

- **Bloat**: uncontrolled growth of programs
- Is it good or bad?
- Well, OK, if it is bad... then why is there bloat?
- So what can we do against it?

- **Bloat**: uncontrolled growth of programs
- Is it good or **bad**?
- Well, OK, if it is bad... then why is there bloat? Some possible reasons:
 - ① Useless code hitchhiking and reproducing with good individuals (high selection pressure \Rightarrow more bloat) [28, 138, 143, 144]
- So what can we do against it?
 - ① Use multi-objective optimization: minimize also program size [30, 153, 154]

- **Bloat**: uncontrolled growth of programs
- Is it good or **bad**?
- Well, OK, if it is bad... then why is there bloat? Some possible reasons:
 - ① Useless code hitchhiking and reproducing with good individuals (high selection pressure \Rightarrow more bloat) [28, 138, 143, 144]
 - ② Protection against reproduction operators: mutation in useless code has no impact \Rightarrow program can survive [144–151]
- So what can we do against it?
 - ① Use multi-objective optimization: minimize also program size [30, 153, 154]
 - ② Use penalties in single-objective optimization [140]

- **Bloat:** uncontrolled growth of programs
- Is it good or **bad?**
- Well, OK, if it is bad... then why is there bloat? Some possible reasons:
 - ① Useless code hitchhiking and reproducing with good individuals (high selection pressure \Rightarrow more bloat) [28, 138, 143, 144]
 - ② Protection against reproduction operators: mutation in useless code has no impact \Rightarrow program can survive [144–151]
 - ③ Removal bias: finite portion of removable useless nodes, but no direct limit to amount of nodes which can be inserted [152]
- So what can we do against it?
 - ① Use multi-objective optimization: minimize also program size [30, 153, 154]
 - ② Use penalties in single-objective optimization [140]
 - ③ Set a conservative upper bound for program size [139]

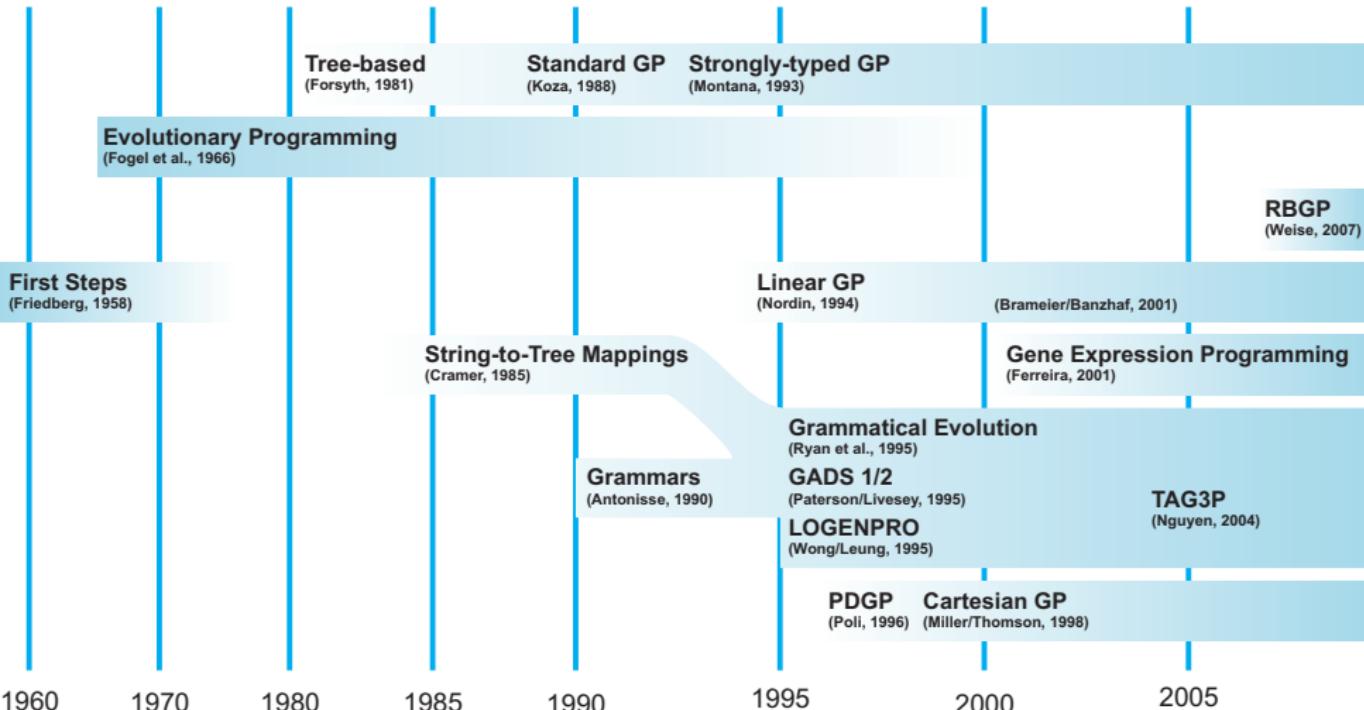
- **Bloat:** uncontrolled growth of programs
- Is it good or **bad?**
- Well, OK, if it is bad... then why is there bloat? Some possible reasons:
 - ① Useless code hitchhiking and reproducing with good individuals (high selection pressure \Rightarrow more bloat) [28, 138, 143, 144]
 - ② Protection against reproduction operators: mutation in useless code has no impact \Rightarrow program can survive [144–151]
 - ③ Removal bias: finite portion of removable useless nodes, but no direct limit to amount of nodes which can be inserted [152]
 - ④ Overfitting: code one decision for each instance of the training and get perfect fitness (but no generalization)
- So what can we do against it?
 - ① Use multi-objective optimization: minimize also program size [30, 153, 154]
 - ② Use penalties in single-objective optimization [140]
 - ③ Set a conservative upper bound for program size [139]
 - ④ Use specialized mutation and crossover operators which minimize bloat [143, 151]

Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

- Genetic Programming evolves programs or trees or graph data structures

- Genetic Programming evolves programs or trees or graph data structures
- There are many ways for an Evolutionary Algorithm to match this definition...



- Linear Genetic Programming [68, 150, 155–178]

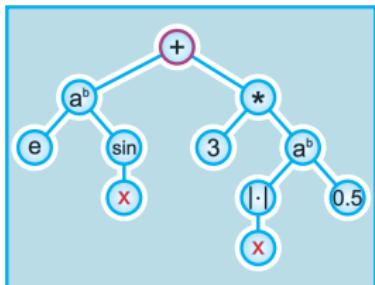
- Linear Genetic Programming [68, 150, 155–178]
- 1-dimensional genome: list of instructions

- Linear Genetic Programming [68, 150, 155–178]
- 1-dimensional genome: list of instructions
- Usually variable-length integer string encoding

- Linear Genetic Programming [68, 150, 155–178]
- 1-dimensional genome: list of instructions
- Usually variable-length integer string encoding
- Similar to (sometimes equivalent to) assembler language with register memory

Linear Genetic Programming

- Linear Genetic Programming [68, 150, 155–178]
- 1-dimensional genome: list of instructions
- Usually variable-length integer string encoding
- Similar to (sometimes equivalent to) assembler language with register memory



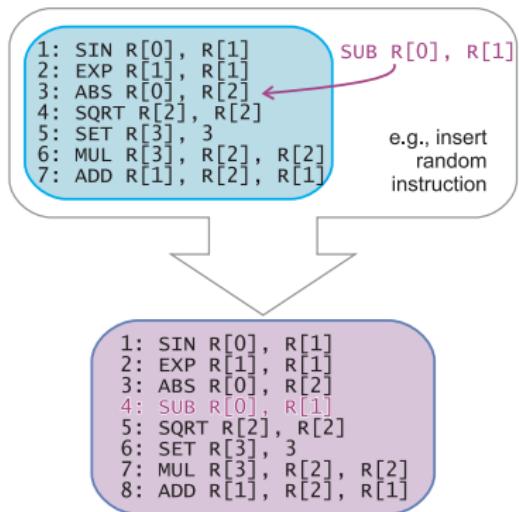
$$\varphi(x) = e^{\sin x} + 3\sqrt{|x|}$$



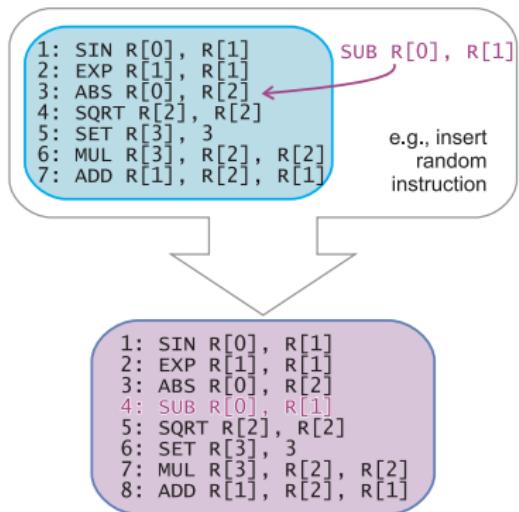
```
1: SIN R[0], R[1]
2: EXP R[1], R[1]
3: ABS R[0], R[2]
4: SQRT R[2], R[2]
5: SET R[3], 3
6: MUL R[3], R[2], R[2]
7: ADD R[1], R[2], R[1]
```

- Mutation: Insert or delete instructions

- Mutation: Insert or delete instructions

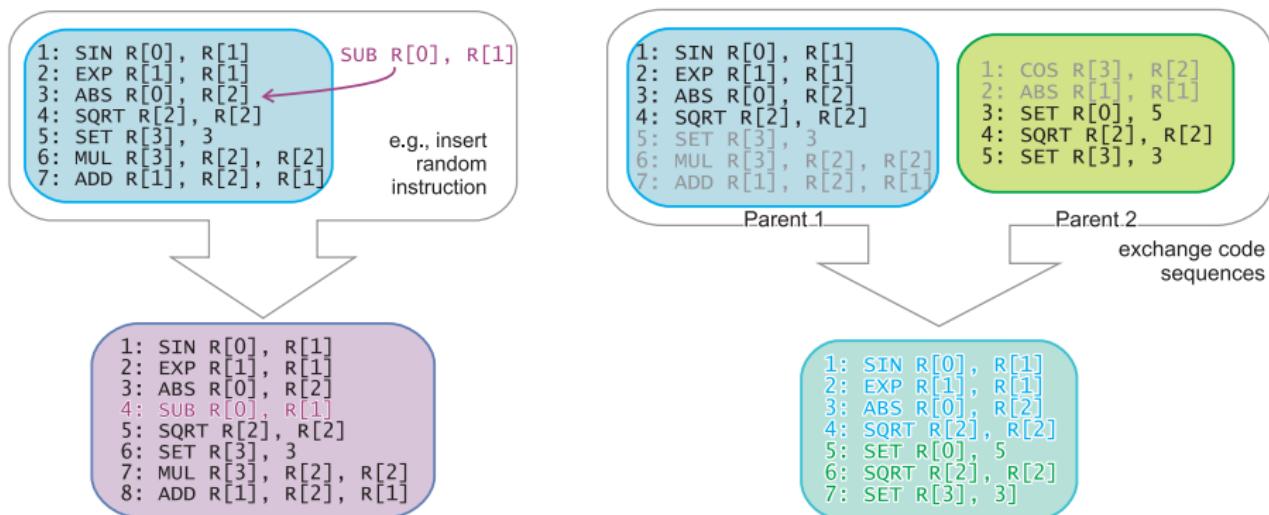


- Mutation: Insert or delete instructions
- Crossover: Exchange code between parents



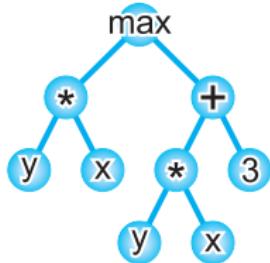
Linear Genetic Programming

- Mutation: Insert or delete instructions
- Crossover: Exchange code between parents



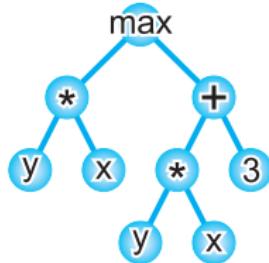
- Graphs are more general than trees

- Graphs are more general than trees

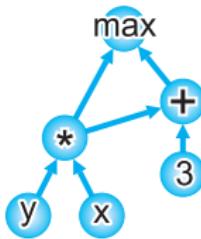


Some function in tree representation.

- Graphs are more general than trees

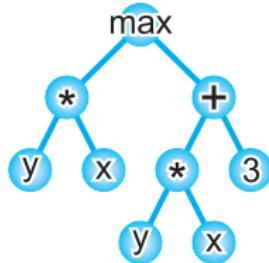


Some function in tree representation.

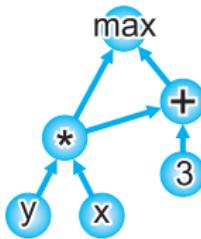


The same function as graph.

- Graphs are more general than trees
- PDGP: Nodes are arranged in a rectangular grid [179–183]

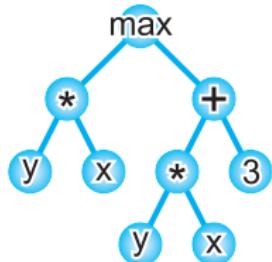


Some function in tree representation.

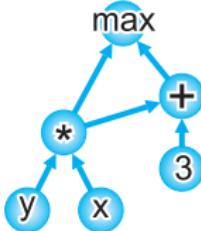


The same function as graph.

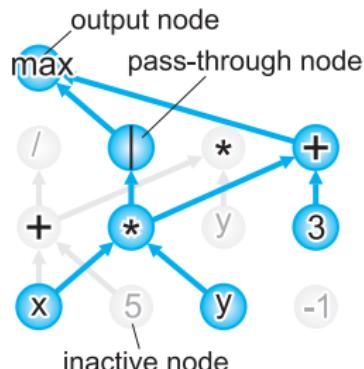
- Graphs are more general than trees
- PDGP: Nodes are arranged in a rectangular grid [179–183]



Some function in tree representation.

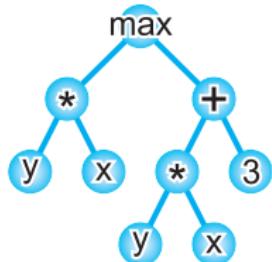


The same function as graph.

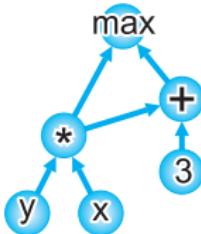


The function as PDGP program.

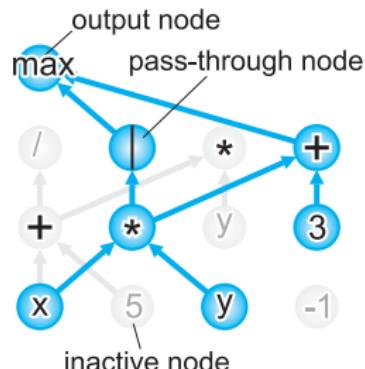
- Graphs are more general than trees
- PDGP: Nodes are arranged in a rectangular grid [179–183]
- Evolution: node functions and node connections



Some function in tree representation.

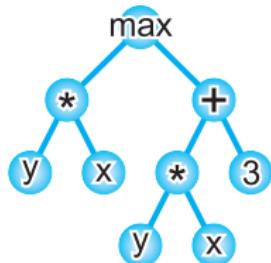


The same function as graph.

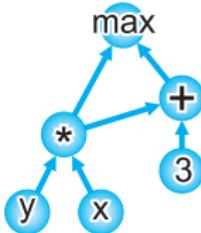


The function as PDGP program.

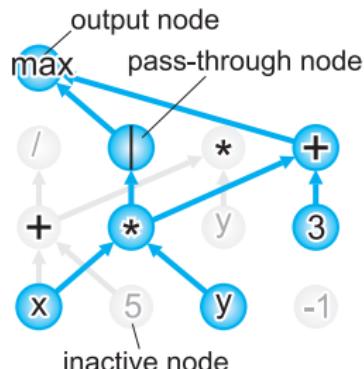
- Graphs are more general than trees
- PDGP: Nodes are arranged in a rectangular grid [179–183]
- Evolution: node functions and node connections
- Crossover: Exchange sub-graphs



Some function in tree representation.



The same function as graph.



The function as PDGP program.

Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

What's it good for?



So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Graphics [2, 16, 26, 37–40]
- Data Mining [6, 8, 37–39, 41–62]
- Circuit Design / Digital Technology [2, 3, 16, 24, 30, 33, 63–69]
- Prediction [38, 39, 43, 46, 47, 59]
- Economy and Finance [31, 33, 43–47, 70]
- Engineering [2, 3, 16, 22, 24, 27, 30, 33, 38, 39, 52, 63–67, 69, 71–96]
- Systems Security [52, 61, 74, 81, 83, 97]
- Networking and Communication [19, 27, 61, 70–73, 75–77, 82, 83, 90–92, 96–109]
- Multi-Agent Systems / Behaviors [70, 90, 91, 95, 102, 110–113]
- Chemistry [38, 39, 71–73, 82, 99, 104–106, 114–118]
- Software (Engineering) [2, 6, 9, 11–13, 29, 64, 69, 92, 103, 119–125]

Just to name a few...

What's it good for?



So what is Genetic Programming good for?

- Maths / Symbolic Regression [3, 9, 14–33]
- Art [34–36]
- Computer Games [37–41]
- Data Mining [42–45]
- Circuit Design [46–49]
- Prediction [50–53]
- Economics [54–57]
- Engineering [58–61]
- Systems Biology [62–65]
- Networking and Communication [19, 27, 61, 70–73, 75–77, 82, 83, 90–92, 96–109]
- Multi-Agent Systems / Behaviors [70, 90, 91, 95, 102, 110–113]
- Chemistry [38, 39, 71–73, 82, 99, 104–106, 114–118]
- Software (Engineering) [2, 6, 9, 11–13, 29, 64, 69, 92, 103, 119–125]

**But where are the
„real“ programs?**

**The one's we write in Java
and C++??**

Just to name a few...

So what is Genetic Programming good for?

- Maths / Symbolic Regression
- Art
- Computer Games
- Data Mining
- Circuit Design
- Predictive Modelling
- Economics
- Engineering
- System Biology
- Networking and Communication
- Multi-Agent Systems / Behaviors
- Chemistry
- Software (Engineering)

[3, 9, 14–33]

[34–36]

But where are the
„real“ programs?
The one's we write
and C++??

Compute factorial $p = a!$ of
natural number a

```
1: p = 1;  
2: while(a > 0) {  
3:   p = p * a;  
4:   a = a - 1;  
5: }
```

Just to name a few...

Evolution of Real Programs



- Evolution of “real” programs is hard

- Evolution of “real” programs is hard because



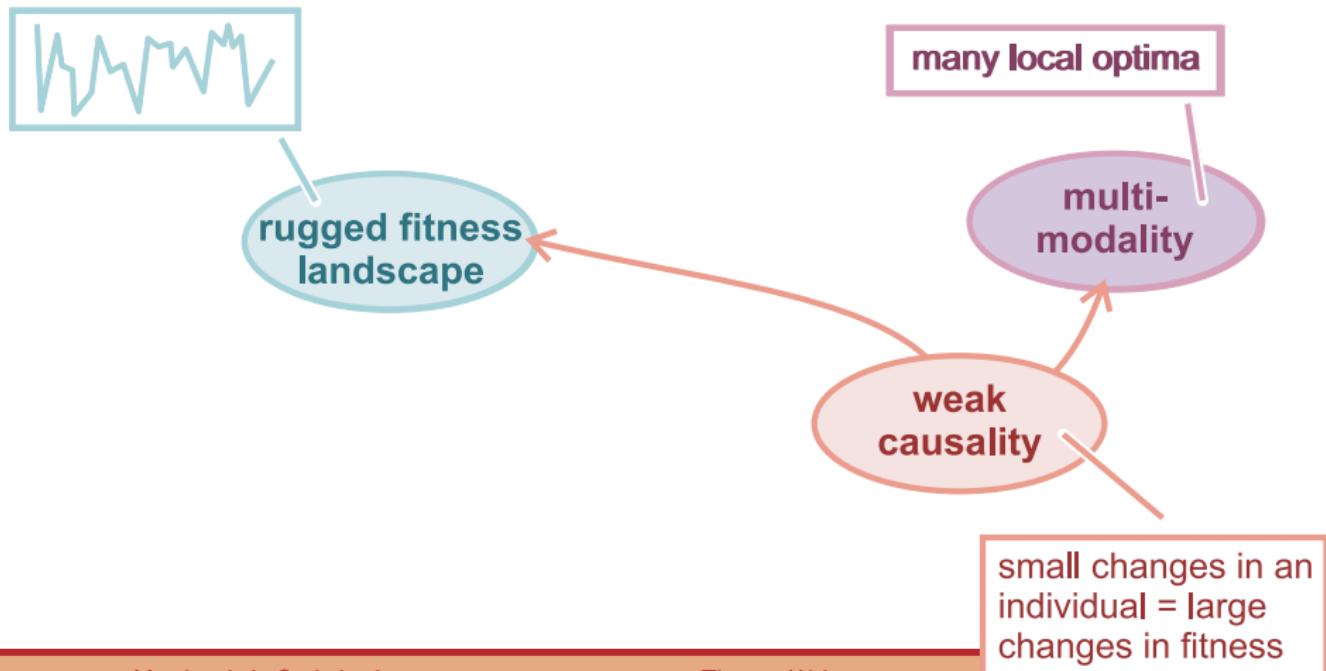
rugged fitness
landscape

many local optima

multi-
modality

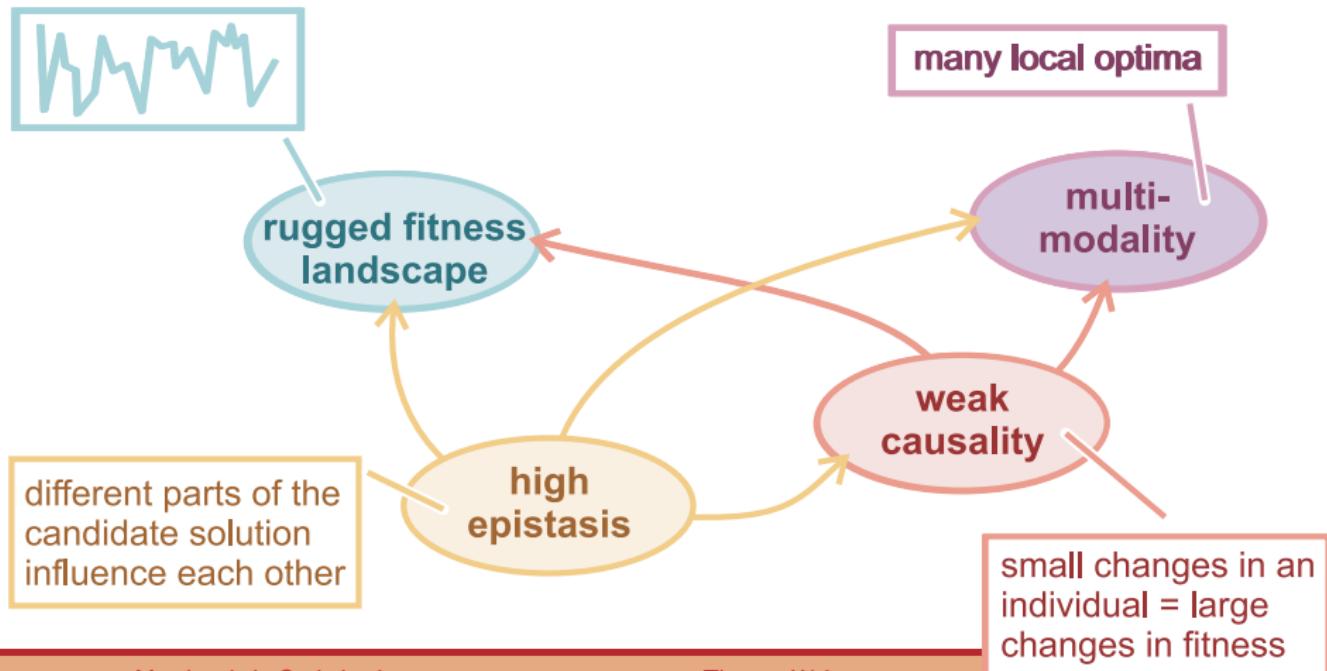
Evolution of Real Programs

- Evolution of “real” programs is hard because



Evolution of Real Programs

- Evolution of “real” programs is hard because



- Epistasis: The **type** of one instructions determines the behavior of much of the program

Epistasis in Genetic Programming

- Epistasis: The **type** of one instructions determines the behavior of much of the program

```
1: p = 1;  
2: while(a > 0) {  
3:   p = p * a;  
4:   a = a - 1;  
5: }
```

factorial p of a

vs.

```
1: p = 1;  
2: while(a > 0) {  
3:   p = p + a;  
4:   a = a - 1;  
5: }
```

$1+0.5a(a+1)$

- Epistasis: The **order** of instructions determines their behavior

Epistasis in Genetic Programming

- Epistasis: The **order** of instructions determines their behavior

```
1: p = 1;  
2: while(a > 0) {  
3:   p = p * a;  
4:   a = a - 1;  
5: }
```

factorial p of a

vs.

```
1: p = 1;  
2: while(a > 0) {  
3:   a = a - 1;  
4:   p = p * a;  
5: }
```

always 0

- Important lesson: Epistasis aka. Linkage aka. separability is a problem in optimization! [184–192]

- Important lesson: Epistasis aka. Linkage aka. separability is a problem in optimization! [184–192]
- We can only solve problems efficiently if we can make small changes to a candidate solution and obtain small changes in the behavior

- Important lesson: Epistasis aka. Linkage aka. separability is a problem in optimization! [184–192]
- We can only solve problems efficiently if we can make small changes to a candidate solution and obtain small changes in the behavior
- Active research area: Finding program representations in GP which have less epistasis

- Important lesson: Epistasis aka. Linkage aka. separability is a problem in optimization! [184–192]
- We can only solve problems efficiently if we can make small changes to a candidate solution and obtain small changes in the behavior
- Active research area: Finding program representations in GP which have less epistasis
- Rule-based Genetic Programming (RBGP): programs as rule sets – order of instructions becomes unimportant [101, 193–197]

Section Outline

- ① Introduction
- ② Symbolic Regression
- ③ Genetic Programming
- ④ Tree Creation
- ⑤ Data Mining
- ⑥ Bloat
- ⑦ Representations in GP
- ⑧ Epistasis

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures
- Many known applications such as data mining or Symbolic Regression

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures
- Many known applications such as data mining or Symbolic Regression
- . . . but danger: bloat!

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures
- Many known applications such as data mining or Symbolic Regression
- ... but danger: bloat!
- Many, many different representations

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures
- Many known applications such as data mining or Symbolic Regression
- ... but danger: bloat!
- Many, many different representations
- Suitable for many types of problems, but not all

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures
- Many known applications such as data mining or Symbolic Regression
- ... but danger: bloat!
- Many, many different representations
- Suitable for many types of problems, but not all
- Epistasis: a problem from nature

- Genetic Programming is an Evolutionary Algorithm for synthesizing program-like or tree-like structures
- Many known applications such as data mining or Symbolic Regression
- ... but danger: bloat!
- Many, many different representations
- Suitable for many types of problems, but not all
- Epistasis: a problem from nature
- Use GP for what it is suitable for (which is not much...), not for other stuff...

谢谢

Thank you

Thomas Weise [汤卫思]

tweise@hfuu.edu.cn

<http://iao.hfuu.edu.cn>

Hefei University, South Campus 2
Institute of Applied Optimization
Shushan District, Hefei, Anhui,
China



Caspar David Friedrich, "Der Wanderer über dem Nebelmeer", 1818
http://en.wikipedia.org/wiki/Wanderer_above_the_Sea_of_Fog

Bibliography



Bibliography I



1. John R. Koza. Non-linear genetic algorithms for solving problems. United States Patent 4,935,877, Alexandria, VA, USA: United States Patent and Trademark Office (USPTO), 1988. Filed May 20, 1988. Issued June 19, 1990. Australian patent 611,350 issued september 21, 1991. Canadian patent 1,311,561 issued december 15, 1992.
2. John R. Koza. Hierarchical genetic algorithms operating on populations of computer programs. In Natesa Sastri Sridharan, editor, *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI'89)*, volume 1, pages 768–774, Detroit, MI, USA, August 1989. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://www.genetic-programming.com/jkpdf/ijcai1989.pdf>.
3. John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Bradford Books. Cambridge, MA, USA: MIT Press, December 1992. ISBN 0-262-11170-5 and 978-0-262-11170-6. URL <http://books.google.de/books?id=Bhtxo60BV0EC>. 1992 first edition, 1993 second edition.
4. Lawrence Jerome Fogel, Alvin J. Owens, and Michael J. Walsh. *Artificial Intelligence through Simulated Evolution*. New York, NY, USA: John Wiley & Sons Ltd., 1966. ISBN 0471265160 and 978-0471265160. URL <http://books.google.de/books?id=EerbAAAACAAJ>.
5. Stephen Frederick Smith. *A Learning System based on Genetic Adaptive Algorithms*. PhD thesis, Pittsburgh, PA, USA: University of Pittsburgh, 1980.
6. Richard S. Forsyth. Beagle – a darwinian approach to pattern recognition. *Kybernetes*, 10(3):159–166, 1981. doi: 10.1108/eb005587. URL http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/papers/kybernetes_forsyth.pdf. Received December 17, 1980. (copy from British Library May 1994).
7. Richard S. Forsyth and Roy Rada. *Machine Learning Applications in Expert Systems and Information Retrieval*. Ellis Horwood Series in Artificial Intelligence. Milton, QLD, Australia: John Wiley & Sons Australia and New York, NY, USA: Halsted Press. ISBN 0-470-20309-9, 0-7458-0045-9, and 9780470203095. URL <http://books.google.de/books?id=TcnaAAAAMAAJ>. Contains chapters on BEAGLE.
8. Richard S. Forsyth. The evolution of intelligence. In Richard S. Forsyth, editor, *Machine Learning: Principles and Techniques*, chapter 4, pages 65–82. London, UK: Chapman & Hall, 1989. Refers also to PC/BEAGLE.
9. Michael Lynn Cramer. A representation for the adaptive generation of simple sequential programs. In John J. Grefenstette, editor, *Proceedings of the 1st International Conference on Genetic Algorithms and their Applications (ICGA'85)*, pages 183–187, Pittsburgh, PA, USA: Carnegie Mellon University (CMU), June 24–26, 1985. Hillsdale, NJ, USA: Lawrence Erlbaum Associates. URL <http://www.sover.net/~nichael/nlc-publications/icga85/index.html>.

Bibliography II



10. Jürgen Schmidhuber. Evolutionary principles in self-referential learning. (on learning how to learn: The meta-meta... hook.). Master's thesis, Munich, Bavaria, Germany: Technische Universität München, Institut für Informatik, Lehrstuhl Professor Radig, May 14, 1987. URL <http://www.idsia.ch/~juergen/diploma.html>.
11. Dirk Dickmanns, Jürgen Schmidhuber, and Andreas Winkhofer. Der genetische algorithmus: Eine implementierung in prolog, 1987. URL <http://www.idsia.ch/~juergen/geneticprogramming.html>. Fortgeschrittenenpraktikum.
12. Joseph F. Hicklin. Application of the genetic algorithm to automatic program generation. Master's thesis, Moscow, ID, USA: University of Idaho, Computer Science Department, 1986. URL <http://books.google.de/books?id=oC5IOAAACAAJ>.
13. Cory Fujiki. An evaluation of hollands genetic algorithm applied to a program generator. Master's thesis, Moscow, ID, USA: University of Idaho, Computer Science Department, 1986.
14. Douglas A. Augusto and Helio Joséé Correa Barbosa. Symbolic regression via genetic programming. In Felipe M. G. Fran  a and Carlos H. C. Ribeiro, editors, *Proceedings of the VI Brazilian Symposium on Neural Networks (SBRN'00)*, pages 173–178, Rio de Janeiro, RJ, Brazil, November 22–25, 2000. Washington, DC, USA: IEEE Computer Society. doi: 10.1109/SBRN.2000.889734.
15. Timothy Lai. Discovery of understandable math formulas using genetic programming. In *Genetic Algorithms and Genetic Programming at Stanford*, pages 118–127. Stanford, CA, USA: Stanford University Bookstore, Stanford University, Fall 2003. URL <http://www.genetic-programming.org/sp2003/Lai.pdf>.
16. Victor Ciesielski and Xiang Li. Analysis of genetic programming runs. In *Proceedings of the 7th Asia-Pacific Conference on Complex Systems (Complex'04)*, Cairns, Australia: Cairns Convention Centre, December 6–10, 2004. URL <http://www.cs.rmit.edu.au/~vc/papers/aspgp04.pdf>.
17. Jenny Rose Finkel. Using genetic programming to evolve an algorithm for factoring numbers. In *Genetic Algorithms and Genetic Programming at Stanford*, pages 52–60. Stanford, CA, USA: Stanford University Bookstore, Stanford University, Fall 2003. URL <http://www.genetic-programming.org/sp2003/Finkel.pdf>.
18. Shengwu Xiong, Weiwu Wang, and Feng Li. A new genetic programming approach in symbolic regression. In *15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'03)*, pages 161–167, Sacramento, CA, USA, November 3–5, 2003. Los Alamitos, CA, USA: IEEE Computer Society Press. doi: 10.1109/TAI.2003.1250185.
19. Abdellah Salhi, Hugh Glaser, and David de Roure. Parallel implementation of a genetic-programming based tool for symbolic regression. *Information Processing Letters*, 66(6):299–307, June 30, 1998. doi: 10.1016/S0020-0190(98)00056-8.

Bibliography III

20. Günther R. Raidl. A hybrid gp approach for numerically robust symbolic regression. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David Edward Goldberg, Hitoshi Iba, and Rick L. Riolo, editors, *Proceedings of the Third Annual Genetic Programming Conference (GP'98)*, pages 323–328, Madison, WI, USA: University of Wisconsin, July 22–25, 1998. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://www.ads.tuwien.ac.at/publications/bib/pdf/raidl-98c.pdf>.
21. Nicholas Freitag McPhee and Riccardo Poli. Memory with memory: Soft assignment in genetic programming. In Maarten Keijzer, Giuliano Antoniol, Clare Bates Congdon, Kalyanmoy Deb, Benjamin Doerr, Nikolaus Hansen, John H. Holmes, Gregory S. Hornby, Daniel Howard, James Kennedy, Sanjeev P. Kumar, Fernando G. Lobo, Julian Francis Miller, Jason H. Moore, Frank Neumann, Martin Pelikan, Jordan B. Pollack, Kumara Sastry, Kenneth Owen Stanley, Adrian Stoica, El-Ghazali Talbi, and Ingo Wegener, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'08)*, pages 1235–1242, Atlanta, GA, USA: Renaissance Atlanta Hotel Downtown, July 12–16, 2008. New York, NY, USA: ACM Press. doi: 10.1145/1389095.1389336. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/McPhee_2008_gecco.html.
22. David J. Montana. Strongly typed genetic programming. *Evolutionary Computation*, 3(2):199–230, Summer 1995. doi: 10.1162/evco.1995.3.2.199. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/montana_stgpEC.html.
23. Heitor Silverio Lopes and Wagner R. Weinert. Egipsys: an enhanced gene expression programming approach for symbolic regression problems. *International Journal of Applied Mathematics and Computer Science (AMCS)*, 14(3), September 2004. URL <http://matwbn.icm.edu.pl/ksiazki/amc/amc14/amc1437.pdf>. Special Issue: Evolutionary Computation. AMCS Centro Federal de Educacao Tecnologica do Parana / CPGEI Av. 7 de setembro, 3165, 80230-901 Curitiba (PR), Brazil.
24. Sean Luke and Liviu Panait. A survey and comparison of tree generation algorithms. In Lee Spector, Erik D. Goodman, Annie S. Wu, William Benjamin Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund K. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'01)*, pages 81–88, San Francisco, CA, USA: Holiday Inn Golden Gateway Hotel, July 7–11, 2001. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://www.cs.gmu.edu/~sean/papers/treegenalgs.pdf>.
25. Maarten Keijzer. Scaled symbolic regression. *Genetic Programming and Evolvable Machines*, 5(3):259–269, September 2004. doi: 10.1023/B:GENP.0000030195.77571.f9.
26. John R. Koza. Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems. Technical Report STAN-CS-90-1314, Stanford, CA, USA: Stanford University, Computer Science Department, June 1990. URL <http://www.genetic-programming.com/jkpdf/tr1314.pdf>.

Bibliography IV



27. Derek M. Johnson, Ankur M. Teredesai, and Robert T. Saltarelli. Genetic programming in wireless sensor networks. In Maarten Keijzer, Andrea G. B. Tettamanzi, Pierre Collet, Jano I. van Hemert, and Marco Tomassini, editors, *Proceedings of the 8th European Conference on Genetic Programming (EuroGP'05)*, volume 3447/2005 of *Lecture Notes in Computer Science (LNCS)*, pages 96–107, Lausanne, Switzerland, March 30–April 1, 2005. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/b107383. URL <http://www.cs.rit.edu/~amt/pubs/EuroGP05FinalTeredesai.pdf>.
28. Steven Matt Gustafson, Anikó Ekárt, Edmund K. Burke, and Graham Kendall. Problem difficulty and code growth in genetic programming. *Genetic Programming and Evolvable Machines*, 5(3):271–290, September 2004. doi: 10.1023/B:GENP.0000030194.98244.e3. URL <http://www.gustafsonresearch.com/research/publications/gustafson-gpem2004.pdf>. Submitted March 4, 2003; Revised August 7, 2003.
29. Cory Fujiko and John Dickinson. Using the genetic algorithm to generate lisp source code to solve the prisoner's dilemma. In John J. Grefenstette, editor, *Proceedings of the Second International Conference on Genetic Algorithms and their Applications (ICGA'87)*, pages 236–240, Cambridge, MA, USA: Massachusetts Institute of Technology (MIT), July 28–31, 1987. Mahwah, NJ, USA: Lawrence Erlbaum Associates, Inc. (LEA).
30. Anikó Ekárt and Sándor Zoltán Németh. Selection based on the pareto nondomination criterion for controlling code growth in genetic programming. *Genetic Programming and Evolvable Machines*, 2(1):61–73, March 2001. doi: 10.1023/A:1010070616149.
31. John Duffy and Jim Engle-Warnick. Using symbolic regression to infer strategies from experimental data. In Shu-Heng Chen, editor, *Evolutionary Computation in Economics and Finance*, volume 100 of *Studies in Fuzziness and Soft Computing*, chapter 4, pages 61–84. Berlin, Germany: Springer-Verlag GmbH, August 5, 2002. URL <http://www.pitt.edu/~jduffy/docs/Usr.ps>.
32. Jing Chen, Zeng-zhi Li, Zhi-Gang Liao, and Yun-lan Wang. Distributed service performance management based on linear regression and genetic programming. In *Proceedings of the 2005 International Conference on Machine Learning and Cybernetics (ICMLC'05)*, volume 1, pages 560–563, Guangzhou, Guangdong, China: Ramada Pearl Hotel, August 18–21, 2005. doi: 10.1109/ICMLC.2005.1527007.

Bibliography V



33. Edmund K. Burke, Steven Matt Gustafson, Graham Kendall, and Natalio Krasnogor. Advanced population diversity measures in genetic programming. In Juan Julián Merelo-Guervós, Panagiotis Adamidis, Hans-Georg Beyer, José Luis Fernández-Villacañas Martín, and Hans-Paul Schwefel, editors, *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature (PPSN VII)*, volume 2439/2002 of *Lecture Notes in Computer Science (LNCS)*, pages 341–350, Granada, Spain, September 7–11, 2002. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/3-540-45712-7_33. URL <http://www.gustafsonresearch.com/research/publications/ppsn-2002.ps>.
34. Tatsuo Unemi. Sbart 2.4: An iec tool for creating 2d images, movies, and collage. In *Workshop “Genetic Algorithms in Visual Art and Music” (GAVAM’00), Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program (GECCO’00 WS)*, pages 153–157, Las Vegas, NV, USA, July 8, 2000. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.3280>.
35. Colin G. Johnson and Riccardo Poli. Gp-music: An interactive genetic programming system for music generation with automated fitness raters. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David Edward Goldberg, Hitoshi Iba, and Rick L. Riolo, editors, *Proceedings of the Third Annual Genetic Programming Conference (GP’98)*, pages 181–186, Madison, WI, USA: University of Wisconsin, July 22–25, 1998. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/johanson_1998_GP-Music.html.
36. Brad Johanson and Riccardo Poli. Gp-music: An interactive genetic programming system for music generation with automated fitness raters. Technical Report CSRP-98-13, Birmingham, UK: University of Birmingham, School of Computer Science, 1998. URL <http://graphics.stanford.edu/~bjohanso/gp-music/tech-report/>.
37. Xiang Li and Victor Ciesielski. Using loops in genetic programming for a two class binary image classification problem. In Geoffrey I. Webb and Xinghuo Yu, editors, *Advances in Artificial Intelligence – Proceedings of the 17th Australian Joint Conference on Artificial Intelligence (AI’04)*, volume 3339/2004 of *Lecture Notes in Computer Science (LNCS)*, pages 898–909, Cairns, Australia: Central Queensland University, December 4–6, 2004. Berlin, Germany: Springer-Verlag GmbH. URL <http://goanna.cs.rmit.edu.au/~vc/papers/aust-ai04.pdf>.
38. William Benjamin Langdon and Wolfgang Banzhaf. A simd interpreter for genetic programming on gpu graphics cards. In Michael O’Neill, Leonardo Vanneschi, Steven Matt Gustafson, Anna Isabel Esparcia-Alcázar, Ivano Falco, Antonio Della Cioppa, and Ernesto Tarantino, editors, *Genetic Programming – Proceedings of the 11th European Conference on Genetic Programming (EuroGP’08)*, volume 4971/2008 of *Lecture Notes in Computer Science (LNCS)*, pages 73–85, Naples, Italy, March 26–28, 2008. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/978-3-540-78671-9_7. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/langdon_2008_eurogp.html.

Bibliography VI



39. William Benjamin Langdon. A simd interpreter for genetic programming on gpu graphics cards. Computer Science Technical Report CSM-470, Wivenhoe Park, Colchester, Essex, UK: University of Essex, Departments of Mathematical and Biological Sciences, July 3, 2007. URL http://cswww.essex.ac.uk/technical-reports/2007/csm_470.pdf.
40. David Andre. Learning and upgrading rules for an optical character recognition system using genetic programming. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, Computational Intelligence Library. New York, NY, USA: Oxford University Press, Inc., Dirac House, Temple Back, Bristol, UK: Institute of Physics Publishing Ltd. (IOP), and Boca Raton, FL, USA: CRC Press, Inc., January 1, 1997.
41. Pu Wang, Thomas Weise, and Raymond Chiong. Novel evolutionary algorithms for supervised classification problems: An experimental study. *Evolutionary Intelligence*, 4(1):3–16, January 12, 2011. doi: 10.1007/s12065-010-0047-7.
42. Alex Alves Freitas. *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Natural Computing Series. New York, NY, USA: Springer New York, 2002. ISBN 3-540-43331-7 and 978-3-540-43331-6. URL <http://books.google.de/books?id=KkdZlfQJvbYC>.
43. Alma Lilia García-Almanza and Edward P. K. Tsang. The repository method for chance discovery in financial forecasting. In Bogdan Gabrys, Robert J. Howlett, and Lakhmi C. Jain, editors, *Proceedings of the 10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, Part III (KES'06)*, volume 4253/2006 of *Lecture Notes in Computer Science (LNCS)*, pages 30–37, Bournemouth, UK: Bournemouth International Centre, October 9–11, 2006. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/11893011_5.
44. Pu Wang, Edward P. K. Tsang, Thomas Weise, Ke Tang, and Xin Yao. Using gp to evolve decision rules for classification in financial data sets. In Fuchun Sun, Yingxu Wang, Jianhua Lu, Bo Zhang, Witold Kinsner, and Lotfi A. Zadeh, editors, *Proceedings of the 9th IEEE International Conference on Cognitive Informatics (ICCI'10)*, pages 722–727, Beijing, China: Tsinghua University, July 7–9, 2010. Los Alamitos, CA, USA: IEEE Computer Society Press. doi: 10.1109/COGINF.2010.5599820.
45. Alma Lilia García-Almanza, Edward P. K. Tsang, and Edgar Galván-López. Evolving decision rules to discover patterns in financial data sets. In Erricos John Kontoghiorghes, Berç Rustem, and Peter Winker, editors, *Computational Methods in Financial Engineering – Essays in Honour of Manfred Gilli*, chapter II-5, pages 239–255. Berlin/Heidelberg: Springer-Verlag, 2008. doi: 10.1007/978-3-540-77958-2_12. URL <http://www.bracil.net/finance/papers/GarciaTsangGalvan-Ecr-Book-2007.pdf>.
46. Jin Li. *FGP: A Genetic Programming Based Tool for Financial Forecasting*. PhD thesis, Wivenhoe Park, Colchester, Essex, UK: University of Essex, October 6, 2001.

Bibliography VII



47. Edward P. K. Tsang, Paul Yung, and Jin Li. Eddie-automation – a decision support tool for financial forecasting. *Decision Support Systems*, 37(4):559–565, September 2004. doi: 10.1016/S0167-9236(03)00087-3. URL <http://www.bracil.net/finance/papers/TsYuLi-Eddie-Dss2004.pdf>.
48. Durga Prasand Muni, Nikhil R. Pal, and Jyotirmoy Das. A novel approach to design classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation (IEEE-EC)*, 8(2):183–196, April 2004. doi: 10.1109/TEVC.2004.825567.
49. Hajira Jabeen and Abdul Rauf Baig. Review of classification using genetic programming. *International Journal of Engineering Science and Technology*, 2(2):94–103, February 2010. URL <http://www.ijest.info/docs/IJEST10-02-02-06.pdf>.
50. Chi Zhou, Weimin Xiao, Thomas M. Tirpak, and Peter C. Nelson. Evolving accurate and compact classification rules with gene expression programming. *IEEE Transactions on Evolutionary Computation (IEEE-EC)*, 7(6):519–531, December 2003. doi: 10.1109/TEVC.2003.819261.
51. Christian Voigtmann. Integration evolutionärer klassifikatoren in weka. Bachelor's thesis, Kassel, Hesse, Germany: University of Kassel, Fachbereich 16: Elektrotechnik/Informatik, Distributed Systems Group, October 26, 2008.
52. Jaroslaw Skaruz and Franciszek Seredynski. Detecting web application attacks with use of gene expression programming. pages 2029–2035. doi: 10.1109/CEC.2009.4983190.
53. Fernando E. B. Otero, Monique M. S. Silva, Alex Alves Freitas, and Julio C. Nievola. Genetic programming for attribute construction in data mining. In Conor Ryan, Terence Soule, Maarten Keijzer, Edward P. K. Tsang, Riccardo Poli, and Ernesto Jorge Fernandes Costa, editors, *Proceedings of the 6th European Conference on Genetic Programming (EuroGP'03)*, volume 2610/2003 of *Lecture Notes in Computer Science (LNCS)*, pages 101–121, Wivenhoe Park, Colchester, Essex, UK: University of Essex, Departments of Mathematical and Biological Sciences, April 14–16, 2003. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/3-540-36599-0_36. URL http://www.cs.kent.ac.uk/people/staff/aaf/pub_papers.dir/EuroGP-2003-Fernando.pdf.
54. Roberto R. F. Mendes, Fabricio de B. Voznika, Alex Alves Freitas, and Julio C. Nievola. Discovering fuzzy classification rules with genetic programming and co-evolution. In Luc de Raedt and Arno Siebes, editors, *5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'01)*, volume 2168 of *Lecture Notes in Computer Science (LNCS)*, pages 314–325, Freiburg, Germany, September 3–5, 2001. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/3-540-44794-6_26. URL http://www.cs.kent.ac.uk/people/staff/aaf/pub_papers.dir/PKDD-2001.ps.

Bibliography VIII



55. John R. Koza. Concept formation and decision tree induction using the genetic programming paradigm. In Hans-Paul Schwefel and Reinhard Männer, editors, *Proceedings of the 1st Workshop on Parallel Problem Solving from Nature (PPSN I)*, volume 496/1991 of *Lecture Notes in Computer Science (LNCS)*, pages 124–128, Dortmund, North Rhine-Westphalia, Germany: FRG Dortmund, October 1–3, 1990. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/BFb0029742. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.53.5800>.
56. John R. Koza. Concept formation and decision tree induction using the genetic programming paradigm. In Hans-Paul Schwefel and Reinhard Männer, editors, *Proceedings of the 1st Workshop on Parallel Problem Solving from Nature (PPSN I)*, volume 496/1991 of *Lecture Notes in Computer Science (LNCS)*, pages 124–128, Dortmund, North Rhine-Westphalia, Germany: FRG Dortmund, October 1–3, 1990. Berlin, Germany: Springer-Verlag GmbH. URL <http://citeseer.ist.psu.edu/61578.html>.
57. Alex Alves Freitas. A genetic programming framework for two data mining tasks: Classification and generalized rule induction. In John R. Koza, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, Hitoshi Iba, and Rick L. Riolo, editors, *Proceedings of the Second Annual Conference on Genetic Programming (GP'97)*, pages 96–101, Stanford, CA, USA: Stanford University, July 13–16, 1997. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://kar.kent.ac.uk/21483/>.
58. Erick Cantú-Paz and Chandrika Kamath. Using evolutionary algorithms to induce oblique decision trees. In L. Darrell Whitley, David Edward Goldberg, Erick Cantú-Paz, Lee Spector, Ian C. Parmee, and Hans-Georg Beyer, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'00)*, pages 1053–1060, Las Vegas, NV, USA: Riviera Hotel and Casino, July 8–12, 2000. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <https://computation.llnl.gov/casc/sapphire/dtrees/oc1.html>.
59. Celia C. Bojarczuk, Heitor Silverio Lopes, and Alex Alves Freitas. An innovative application of a constrained-syntax genetic programming system to the problem of predicting survival of patients. In Conor Ryan, Terence Soule, Maarten Keijzer, Edward P. K. Tsang, Riccardo Poli, and Ernesto Jorge Fernandes Costa, editors, *Proceedings of the 6th European Conference on Genetic Programming (EuroGP'03)*, volume 2610/2003 of *Lecture Notes in Computer Science (LNCS)*, pages 11–59, Wivenhoe Park, Colchester, Essex, UK: University of Essex, Departments of Mathematical and Biological Sciences, April 14–16, 2003. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/3-540-36599-0_2. URL http://www.cs.kent.ac.uk/people/staff/aaf/pub_papers.dir/EuroGP-2003-Celia.pdf.

Bibliography IX



60. Celia C. Bojarczuk, Heitor Silverio Lopes, and Alex Alves Freitas. Data mining with constrained-syntax genetic programming: Applications to medical data sets. In Blaz Zupan, Elpida Keravnou, and Nada Lavrac, editors, *Proceedings Intelligent Data Analysis in Medicine and Pharmacology (IDAMAP'01)*, The Springer International Series in Engineering and Computer Science, London, UK: Custom House Hotel at ExCeL, September 4, 2001. Boston, MA, USA: Springer US. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/bojarczuk_2001_idemap.html.
61. Riyad Alshammari, Peter I. Lichodzijewski, Malcom Ian Heywood, and A. Nur Zincir-Heywood. Classifying ssh encrypted traffic with minimum packet header features using genetic programming. In Franz Rothlauf, Günther R. Raidl, Anna Isabel Esparcia-Alcázar, Ying-Ping Chen, Gabriela Ochoa, Ender Ozcan, Marc Schoenauer, Anne Auger, Hans-Georg Beyer, Nikolaus Hansen, Steffen Finck, Raymond Ros, L. Darrell Whitley, Garnett Wilson, Simon Harding, William Benjamin Langdon, Man Leung Wong, Laurence D. Merkle, Frank W. Moore, Sevan G. Ficici, William Rand, Rick L. Riolo, Nawwaf Kharma, William R. Buckley, Julian Francis Miller, Kenneth Owen Stanley, Jaume Bacardit i Peñarroya, Will N. Browne, Jan Drugowitsch, Nicola Beume, Mike Preuß, Stephen L. Smith, Stefano Cagnoni, Alexandru Floares, Aaron Baughman, Steven Matt Gustafson, Maarten Keijzer, Arthur Kordon, and Clare Bates Congdon, editors, *Proceedings of the 11th Annual Conference – Companion on Genetic and Evolutionary Computation Conference (GECCO'09)*, pages 2539–2546, Montréal, QC, Canada: Delta Centre-Ville Hotel, July 8–12, 2009. New York, NY, USA: Association for Computing Machinery (ACM). doi: 10.1145/1570256.1570358.
62. David Andre, Forrest H. Bennett III, and John R. Koza. Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. pages 3–11. URL <http://citeseer.ist.psu.edu/33008.html>.
63. Stewart W. Wilson. Generalization in the xcs classifier system. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David Edward Goldberg, Hitoshi Iba, and Rick L. Riolo, editors, *Proceedings of the Third Annual Genetic Programming Conference (GP'98)*, pages 665–674, Madison, WI, USA: University of Wisconsin, July 22–25, 1998. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <ftp://ftp.cs.bham.ac.uk/pub/authors/T.Kovacs/lcs.archive/Wilson1998a.ps.gz>.
64. Kevin J. Lang. Hill climbing beats genetic search on a boolean circuit synthesis problem of koza's. In Armand Prieditis and Stuart J. Russell, editors, *Proceedings of the Twelfth International Conference on Machine Learning (ICML'95)*, pages 340–343, Tahoe City, CA, USA, July 9–12, 1995. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://www.genetic-programming.com/lang-ml95.ps>.

Bibliography X

65. Tatiana Kalganova. An extrinsic function-level evolvable hardware approach. In Riccardo Poli, Wolfgang Banzhaf, William Benjamin Langdon, Julian Francis Miller, Peter Nordin, and Terence Claus Fogarty, editors, *Proceedings of the Third European Conference on Genetic Programming (EuroGP'00)*, volume 1802/2000 of *Lecture Notes in Computer Science (LNCS)*, pages 60–75, Edinburgh, Scotland, UK, April 15–16, 2000. France: EvoGP, The Genetic Programming Working Group of EvoNET, The Network of Excellence in Evolutionary Computing, Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/b75085. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.1386>.
66. John R. Koza. A response to the ml-95 paper entitled "hill climbing beats genetic search on a boolean circuit synthesis of koza's". URL <http://www.genetic-programming.com/jktahoe24page.html>. Version 1 distributed at the Twelfth International Machine Learning Conference (ML-95). Version 2 distributed at the Sixth International Conference on Genetic Algorithms (ICGA-95). Version 3 deposited at WWW site on May 26, 1996.
67. John R. Koza. A hierarchical approach to learning the boolean multiplexer function. In Bruce M. Spatz and Gregory J. E. Rawlins, editors, *Proceedings of the First Workshop on Foundations of Genetic Algorithms (FOGA'90)*, pages 171–191, Bloomington, IN, USA: Indiana University, Bloomington Campus, July 15–18, 1990. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://www.genetic-programming.com/jkpdf/foga1990.pdf>.
68. Sven E. Eklund. A massively parallel gp engine in vlsi. In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Hitoshi Iba, Paul Marrow, and Mark Shackleton, editors, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'02), 2002 IEEE World Congress on Computation Intelligence (WCCI'02)*, volume 1-2, pages 629–633, Honolulu, HI, USA: Hilton Hawaiian Village Hotel (Beach Resort & Spa), May 12–17, 2002. Piscataway, NJ, USA: IEEE Computer Society, Los Alamitos, CA, USA: IEEE Computer Society Press. doi: 10.1109/CEC.2002.1006999.
69. Sérgio Granato de Araújo, Antônio Carneiro de Mesquita Filho, and Aloysio de Castro Pinto Pedroza. Síntese de circuitos digitais otimizados via programação genética. In *XXX Seminário Integrado de Software e Hardware in Proceedings of XXIII Congresso da Sociedade Brasileira de Computação (SEMISH'03)*, volume III, pages 273–285, August 4–5, 2003. URL <http://www.gta.ufrj.br/ftp/gta/TechReports/AMP03d.pdf>.
70. Kenneth J. Mackin and Eiichiro Tazaki. Emergent agent communication in multi-agent systems using automatically defined function genetic programming (adf-gp). In *IEEE International Conference on Systems, Man, and Cybernetics – Human Communication and Cybernetics (SMC'99)*, volume 5, pages 138–142, Tokyo, Japan: Tokyo University, Department of Aeronautics and Space Engineering, October 12–15, 1999. Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/ICSMC.1999.815536.

Bibliography XI



71. Lidia A. R. Yamamoto and Christian F. Tschudin. Experiments on the automatic evolution of protocols using genetic programming. Technical Report CS-2005-002, Basel, Switzerland: University of Basel, Computer Science Department, Computer Networks Group, April 21, 2005. URL <http://cn.cs.unibas.ch/people/ly/doc/wac2005tr-lyct.pdf>.
72. Christian F. Tschudin and Lidia A. R. Yamamoto. A metabolic approach to protocol resilience. In Mikhail I. Smirnov, editor, *Autonomic Communication – Revised Selected Papers from the First International IFIP Workshop (WAC'04)*, volume 3457/2005 of *Lecture Notes in Computer Science (LNCS)*, pages 191–206, Berlin, Germany, October 18–19, 2004. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/11520184_15. URL <http://cn.cs.unibas.ch/pub/doc/2004-wac-lncs.pdf>.
73. Christian F. Tschudin. Fraglets – a metabolic execution model for communication protocols. In *The Second Annual Symposium on Autonomous Intelligent Networks and Systems (AINS'03)*, Menlo Park, CA, USA: SRI International, June 30–July 1, 2003. Los Angeles, CA, USA: Center for Autonomous Intelligent Networks and Systems (CAINS), University of California (UCLA). URL <http://cn.cs.unibas.ch/pub/doc/2003-ains.pdf>.
74. Jaroslaw Skaruz and Franciszek Seredyński. Web application security through gene expression programming. In Mario Giacobini, Penousal Machado, Anthony Brabazon, Jon McCormack, Stefano Cagnoni, Michael O'Neill, Gianni A. Di Caro, Ferrante Neri, Anikó Ekárt, Mike Preuß, Anna Isabel Esparcia-Alcázar, Franz Rothlauf, Muddassar Farooq, Ernesto Tarantino, Andreas Fink, and Shengxiang Yang, editors, *Applications of Evolutionary Computing – Proceedings of EvoWorkshops 2009: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoGAMES, EvoHOT, EvoASP, EvoINTERACTION, EvoMUSART, EvoNUM, EvoSTOC, EvoTRANSLOG (EvoWorkshops'09)*, volume 5484/2009 of *Lecture Notes in Computer Science (LNCS)*, pages 1–10, Tübingen, Germany: Eberhard-Karls-Universität Tübingen, Fakultät für Informations- und Kognitionswissenschaften, April 15–17, 2009. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/978-3-642-01129-0_1.
75. I. M. A. Kirkwood, S. H. Shami, and Mark C. Sinclair. Discovering simple fault-tolerant routing rules using genetic programming. In George D. Smith, Nigel C. Steele, and Rudolf F. Albrecht, editors, *Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms (ICANNGA'97)*, SpringerComputerScience, Vienna, Austria, April 1–4, 1997. Vienna, Austria: Springer Verlag GmbH. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/kirkam_1997_dsftrr.html.
76. Mark C. Sinclair. Node-pair encoding genetic programming for optical mesh network topology design. In David Wolfe Corne, Martin J. Oates, and George D. Smith, editors, *Telecommunications Optimization: Heuristic and Adaptive Techniques*, chapter 6, pages 99–114. New York, NY, USA: John Wiley & Sons Ltd., September 2000.

Bibliography XII



77. Mark C. Sinclair. Optical mesh network topology design using node-pair encoding genetic programming. In Wolfgang Banzhaf, Jason M. Daida, Ágoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark J. Jakielka, and Robert Elliott Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)*, volume 2, pages 1192–1197, Orlando, FL, USA, July 13–17, 1999. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/sinclair_1999_OMNTDNEGP.html.
78. Lee Spector. Simultaneous evolution of programs and their control structures. In Peter John Angeline and Kenneth E. Kinnear, Jr, editors, *Advances in Genetic Programming II*, Bradford Books, chapter 7, pages 137–154. Cambridge, MA, USA: MIT Press, October 26, 1996. URL <http://helios.hampshire.edu/lspector/pubs/AiGP2-post-final-e.pdf>.
79. Lee Spector. Evolving control structures with automatically defined macros. In Eric V. Siegel and John R. Koza, editors, *Papers from the 1995 AAAI Fall Symposium on Genetic Programming*, pages 99–105, Cambridge, MA, USA: Massachusetts Institute of Technology (MIT), November 10–12, 1995. Menlo Park, CA, USA: AAAI Press. URL <http://helios.hampshire.edu/lspector/pubs/ADM-fallsymp-e.pdf>.
80. Kaisa Miettinen, Marko M. Mäkelä, Pekka Neittaanmäki, and Jacques Periaux, editors. *Invited Lectures of the European Short Course on Genetic Algorithms and Evolution Strategies (EUROGEN'99)*, Jyväskylä, Finland: University of Jyväskylä, May 30–June 3, 1999. Chichester, West Sussex, UK: Wiley Interscience. ISBN 0471999024 and 978-0471999027. Full title: Evolutionary Algorithms in Engineering and Computer Science – Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming, Genetic Programming and Industrial Applications.
81. Agustín Orfila, Juan M. Estevez-Tapiador, and Arturo Ribagorda. Evolving high-speed, easy-to-understand network intrusion detection rules with genetic programming. In Mario Giacobini, Penousal Machado, Anthony Brabazon, Jon McCormack, Stefano Cagnoni, Michael O'Neill, Gianni A. Di Caro, Ferrante Neri, Anikó Ekárt, Mike Preuß, Anna Isabel Esparcia-Alcázar, Franz Rothlauf, Muddassar Farooq, Ernesto Tarantino, Andreas Fink, and Shengxiang Yang, editors, *Applications of Evolutionary Computing – Proceedings of EvoWorkshops 2009: EvoCOMNET, EvoENVIRONMENT, EvoFIN, EvoGAMES, EvoHOT, EvoASP, EvoINTERACTION, EvoMUSART, EvoNUM, EvoSTOC, EvoTRANSLOG (EvoWorkshops'09)*, volume 5484/2009 of *Lecture Notes in Computer Science (LNCS)*, pages 93–98, Tübingen, Germany: Eberhard-Karls-Universität Tübingen, Fakultät für Informations- und Kognitionswissenschaften, April 15–17, 2009. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/978-3-642-01129-0_11.
82. Daniele Miorandi, Paolo Dini, Eitan Altman, and Hisao Kameda. Wp 2.2 – paradigm applications and mapping, d2.2.2 framework for distributed on-line evolution of protocols and services, June 14, 2007. URL http://www.bionets.eu/docs/BIONETS_D2_2_2.pdf. BIONETS/CN/wp2.2/v2.1. Status: Final, Public.

Bibliography XIII

83. Wei Lu and Issa Traore. Detecting new forms of network intrusions using genetic programming. *Computational Intelligence*, 20(3):475–494, August 2004. doi: 10.1111/j.0824-7935.2004.00247.x. URL <http://www.isot.ece.uvic.ca/publications/journals/coi-2004.pdf>.
84. Jason D. Lohn, Gregory S. Hornby, and Derek Linden. An evolved antenna for deployment on nasa's space technology 5 mission. In Una-May O'Reilly, Gwoing Tina Yu, Rick L. Riolo, and Bill Worzel, editors, *Genetic Programming Theory and Practice II, Proceedings of the Second Workshop on Genetic Programming (GPTP'04)*, volume 8 of *Genetic Programming Series*, pages 301–315, Ann Arbor, MI, USA: University of Michigan, Center for the Study of Complex Systems (CSCS), May 13–15, 2004. Boston, MA, USA: Kluwer Publishers. doi: 10.1007/b101112. URL http://ic.arc.nasa.gov/people/hornby/papers/lohn_gptp04.ps.gz.
85. John R. Koza, Martin A. Keane, Matthew J. Streeter, William Mydlowec, Jessen Yu, and Guido Lanza. *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*, volume 5 of *Genetic Programming Series*. New York, NY, USA: Springer Science+Business Media, Inc., 2003. ISBN 0-387-25067-0, 0-387-26417-5, 1402074468, 6610611831, 978-0-387-25067-0, 978-0-387-26417-2, 978-1402074462, and 9786610611836. URL <http://books.google.de/books?id=YQxWzAEnINIC>.
86. John R. Koza, Forrest H. Bennett III, David Andre, and Martin A. Keane. The design of analog circuits by means of genetic programming. In Peter John Bentley, editor, *Evolutionary Design by Computers*, chapter 16, pages 365–385. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., May 1999. URL <http://www.genetic-programming.com/jkpdf/edc1999.pdf>.
87. John R. Koza, Forrest H. Bennett III, David Andre, and Martin A. Keane. Automatic design of analog electrical circuits using genetic programming. In Hugh Cartwright, editor, *Intelligent Data Analysis in Science*, volume 4 of *Oxford Chemistry Masters*, chapter 8, pages 172–200. New York, NY, USA: Oxford University Press, Inc., June 2000. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/koza_2000_idas.html.
88. John R. Koza, David Andre, Forrest H. Bennett III, and Martin A. Keane. Use of automatically defined functions and architecture-altering operations in automated circuit synthesis using genetic programming. In John R. Koza, David Edward Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Proceedings of the First Annual Conference of Genetic Programming (GP'96)*, Bradford Books, pages 132–149, Stanford, CA, USA: Stanford University, July 28–31, 1996. Cambridge, MA, USA: MIT Press. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.51.3933>.

Bibliography XIV



89. John R. Koza, David Andre, Forrest H. Bennett III, and Martin A. Keane. Evolution of a low-distortion, low-bias 60 decibel op amp with good frequency generalization using genetic programming. In John R. Koza, editor, *Late Breaking Papers at the First Annual Conference Genetic Programming (GP'96 LBP)*, pages 94–100, Stanford, CA, USA: Stanford University, July 28–31, 1996. Stanford, CA, USA: Stanford University Bookstore, Stanford University. URL <http://www.genetic-programming.com/jkpdf/gp1996lpampifier.pdf>.
90. Hitoshi Iba, Toshihide Nozoe, and Kanji Ueda. Evolving communicating agents based on genetic programming. In Thomas Bäck, Zbigniew Michalewicz, and Xin Yao, editors, *Proceedings of the IEEE International Conference on Evolutionary Computation (CEC'97)*, pages 297–302, Indianapolis, IN, USA, April 13–16, 1997. Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/ICEC.1997.592321.
91. Hitoshi Iba. Emergent cooperation for multiple agents using genetic programming. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature (PPSN IV)*, volume 1141/1996 of *Lecture Notes in Computer Science (LNCS)*, pages 32–41, Berlin, Germany, September 22–24, 1996. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/3-540-61723-X_967.
92. Paul Grace. Genetic programming and protocol configuration. Master's thesis, Lancaster, Lancashire, UK: Lancaster University, Computing Department, September 2000. URL <http://www.lancs.ac.uk/postgrad/gracep/msc.pdf>.
93. Forrest H. Bennett III, John R. Koza, David Andre, and Martin A. Keane. Evolution of a 60 decibel op amp using genetic programming. In Tetsuya Higuchi, Masaya Iwata, and Weixin Liu, editors, *The First International Conference on Evolvable Systems: From Biology to Hardware, Revised Papers (ICES'96)*, volume 1259/1997 of *Lecture Notes in Computer Science (LNCS)*, Tsukuba, Japan, October 7–8, 1996. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/3-540-63173-9_65. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.6051>.
94. Hans-Georg Beyer, Eva Brucherseifer, Wilfried Jakob, Hartmut Pohlheim, Bernhard Sendhoff, and Thanh Binh To. Evolutionary algorithms – terms and definitions, February 25, 2002. URL <http://ls11-www.cs.uni-dortmund.de/people/beyer/EA-glossary/>.
95. David Andre and Astro Teller. Evolving team darwin united. In Minoru Asada and Hiroaki Kitano, editors, *RoboCup-98: Robot Soccer World Cup II*, volume 1604 of *Lecture Notes in Computer Science (LNCS)*, pages 346–351, Paris, France, July 1998. Berlin, Germany: Springer-Verlag GmbH. URL http://www.cs.cmu.edu/afs/cs/usr/astro/public/papers/Teller_Astro.ps.

Bibliography XV



96. P. Aiyarak, A. S. Saket, and Mark C. Sinclair. Genetic programming approaches for minimum cost topology optimisation of optical telecommunication networks. In Ali M. S. Zalzala, editor, *Proceedings of the Second IEE/IEEE International Conference On Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA'97)*, volume 1997 of *IET Conference Publications*, pages 415–420, Glasgow, Scotland, UK: University of Strathclyde, September 2–4, 1997. Stevenage, Herts, UK: Institution of Engineering and Technology (IET). doi: 10.1049/cp:19971216. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/aiyarak_1997_GPtototn.html.
97. Mark Crosbie and Gene Spafford. Applying genetic programming to intrusion detection. In John R. Koza and Eric V. Siegel, editors, *Proceedings of 1995 AAAI Fall Symposium Series, Genetic Programming Track*, Cambridge, MA, USA: Massachusetts Institute of Technology (MIT), November 10–12, 1995. URL <http://ftp.cerias.purdue.edu/pub/papers/mark-crosbie/mcrossbie-spaf-AAAI.ps.Z>. The paper appeared also as technical report COAST TR 95-05 of COAST Laboratory, Department of Computer Sciences, Purdue University, West Lafayette, IN, USA.
98. Lerina Aversano, Massililiano di Penta, and Kunal Taneja. A genetic programming approach to support the design of service compositions. *International Journal of Computer Systems Science and Engineering (CSSE)*, 21(4):247–254, July 2006. URL <http://www.rcost.unisannio.it/mdipenta/papers/csse06.pdf>.
99. Lidia A. R. Yamamoto, Daniel Schreckling, and Thomas Meyer. Self-replicating and self-modifying programs in fraglets. In *Proceedings of the 2nd International Conference on Bio-Inspired Models of Network, Information, and Computing Systems (BIONETICS'07)*, Budapest, Hungary: Radisson SAS Beke Hotel, December 10–13, 2007. Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/BIMNICS.2007.4610104. URL <http://cn.cs.unibas.ch/people/ly/doc/bionetics2007-ysm.pdf>.
100. Thomas Weise, Kurt Geihs, and Philipp Andreas Baer. Genetic programming for proactive aggregation protocols. In Bartłomiej Beliczyński, Andrzej Dzieliński, Marcin Iwanowski, and Bernardete Ribeiro, editors, *Proceedings of the 8th International Conference on Adaptive and Natural Computing Algorithms, Part I (ICANNGA'07)*, volume 4431/2007 of *Lecture Notes in Computer Science (LNCS)*, pages 167–173, Warsaw, Poland: Warsaw University, April 11–17, 2007. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/978-3-540-71618-1_19.
101. Thomas Weise, Michael Zapf, and Kurt Geihs. Rule-based genetic programming. In *Proceedings of the 2nd International Conference on Bio-Inspired Models of Network, Information, and Computing Systems (BIONETICS'07)*, pages 8–15, Budapest, Hungary: Radisson SAS Beke Hotel, December 10–13, 2007. Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/BIMNICS.2007.4610073.

Bibliography XVI



102. Mohammad Adil Qureshi. Evolving agents. Research Note RN/96/4, London, UK: University College London (UCL), January 1996. URL <http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/papers/AQ.gp96.ps.gz>.
103. Michael Iles and Dwight Lorne Deugo. A search for routing strategies in a peer-to-peer network using genetic programming. In *Proceedings of the 21st IEEE Symposium on Reliable Distributed Systems (SRDS'02)*, pages 341–346, Suita, Ōsaka, Japan: Ōsaka University, October 13–16, 2002. Washington, DC, USA: IEEE Computer Society. doi: 10.1109/RELDIS.2002.1180207.
104. Christian F. Tschudin and Lidia A. R. Yamamoto. Fraglets instruction set, September 24, 2007. URL <http://www.fraglets.net/frag-instrset-20070924.txt>.
105. Paolo Dini. *Digital Business Ecosystems, WP 18: Socio-economic Perspectives of Sustainability and Dynamic Specification of Behaviour in Digital Business Ecosystems, Deliverable 18.4: Report on Self-Organisation from a Dynamical Systems and Computer Science Viewpoint*, March 5, 2007. URL http://files.opaals.org/DBE/deliverables/Del_18.4_DBE_Report_on_self-organisation_from_a_dynamical_systems_Contract_nr. 507953.
106. Paolo Dini. Section 1: Science, new paradigms. subsection 1: A scientific foundation for digital ecosystems. In Francesco Nachira, Paolo Dini, Andrea Nicolai, Marion Le Louarn, and Lorena Rivera Léon, editors, *Digital Business Ecosystems*. Luxembourg: Office for Official Publications of the European Communities, 2007. URL <http://www.digital-ecosystems.org/book/Section1.pdf>. See also: Keynote Talk "Structure and Outlook of Digital Ecosystems Research", IEEE Digital Ecosystems Technologies Conference, DEST07, Cairns, Australia, February 20–23, 2007, and.
107. Jing Chen, Zeng-zhi Li, and Yun-lan Wang. Distributed service management based on genetic programming. In Piotr S. Szczepaniak, Janusz Kacprzyk, and Adam Niewiadomski, editors, *Advances in Web Intelligence, Proceedings of the Third International Atlantic Web Intelligence Conference (AWIC'05)*, volume 3528/2005 of *Lecture Notes in Computer Science (LNCS)*, pages 83–88, Lodz, Poland, June 6–9, 2005. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/11495772_14.
108. Francesc Comellas and G. Giménez. Genetic programming to design communication algorithms for parallel architectures. *Parallel Processing Letters (PPL)*, 8(4):549–560, December 1998. doi: 10.1142/S0129626498000547. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/Comellas_1998_GPD.html.

Bibliography XVII



109. Francesc Comellas. Using genetic programming to design broadcasting algorithms for manhattan street networks. In Günther R. Raidl, Stefano Cagnoni, Jürgen Branke, David Wolfe Corne, Rolf Drechsler, Yaochu Jin, Colin G. Johnson, Penousal Machado, Elena Marchiori, Franz Rothlauf, George D. Smith, and Giovanni Squillero, editors, *Applications of Evolutionary Computing – Proceedings of EvoWorkshops 2004: EvoBIO, EvoCOMNET, EvoHOT, EvoASP, EvoMUSART, and EvoSTOC (EvoWorkshops'04)*, volume 3005/2004 of *Lecture Notes in Computer Science (LNCS)*, pages 170–177, Coimbra, Portugal, April 5–7, 2004. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/978-3-540-24653-4_18.
110. Mohammad Adil Qureshi. *The Evolution of Agents*. PhD thesis, London, UK: University College London (UCL), Department of Computer Science, July 2001. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/qureshi_thesis.html.
111. Sean Luke and Lee Spector. Evolving teamwork and coordination with genetic programming. In John R. Koza, David Edward Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Proceedings of the First Annual Conference of Genetic Programming (GP'96)*, Bradford Books, pages 150–156, Stanford, CA, USA: Stanford University, July 28–31, 1996. Cambridge, MA, USA: MIT Press. URL <http://www.ifi.uzh.ch/groups/ailab/people/nitschke/refs/Cooperation/cooperation.pdf>.
112. Forrest H. Bennett III. Automatic creation of an efficient multi-agent architecture using genetic programming with architecture-altering operations. pages 30–38.
113. Forrest H. Bennett III. Emergence of a multi-agent architecture and new tactics for the ant colony foraging problem using genetic programming. In Pattie Maes, Maja J. Mataric, Jean-Arcady Meyer, Jordan B. Pollack, and Stewart W. Wilson, editors, *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior: From Animals to Animats 4 (SAB'96)*, pages 430–439, Cape Code, MA, USA, September 9–13, 1996. Cambridge, MA, USA: MIT Press.
114. Thomas Weise. Evolving distributed algorithms with genetic programming, January 15, 2009.
115. Christian W. G. Lasarczyk and Wolfgang Banzhaf. Total synthesis of algorithmic chemistries. In Hans-Georg Beyer, Una-May O'Reilly, Dirk V. Arnold, Wolfgang Banzhaf, Christian Blum, Eric W. Bonabeau, Erick Cantú-Paz, Dipankar Dasgupta, Kalyanmoy Deb, James A. Foster, Edwin D. de Jong, Hod Lipson, Xavier Llorà, Spiros Mancoridis, Martin Pelikan, Günther R. Raidl, Terence Soule, Jean-Paul Watson, and Eckart Zitzler, editors, *Proceedings of Genetic and Evolutionary Computation Conference (GECCO'05)*, volume 2, pages 1635–1640, Washington, DC, USA: Loews L'Enfant Plaza Hotel, June 25–27, 2005. New York, NY, USA: ACM Press. doi: 10.1145/1068009.1068285. URL <http://www.cs.bham.ac.uk/~wbl/biblio/gecco2005/docs/p1635.pdf>.

Bibliography XVIII



116. Christian W. G. Lasarczyk and Wolfgang Banzhaf. An algorithmic chemistry for genetic programming. In Maarten Keijzer, Andrea G. B. Tettamanzi, Pierre Collet, Jano I. van Hemert, and Marco Tomassini, editors, *Proceedings of the 8th European Conference on Genetic Programming (EuroGP'05)*, volume 3447/2005 of *Lecture Notes in Computer Science (LNCS)*, pages 1–12, Lausanne, Switzerland, March 30–April 1, 2005. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/978-3-540-31989-4_1. URL <http://ls11-www.cs.uni-dortmund.de/downloads/papers/LaBa05.pdf>.
117. Walter Fontana. Algorithmic chemistry. In Christopher Gale Langdon, Charles E. Taylor, Doyne J. Farmer, and Steen Rasmussen, editors, *Proceedings of the Workshop on Artificial Life (Artificial Life II)*, volume X of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 159–210. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc. and Boulder, CO, USA: Westview Press, February 1990. URL <http://fontana.med.harvard.edu/www/Documents/WF/Papers/alchemy.pdf>.
118. Hongqing Cao, Jingxian Yu, and Lishan Kang. An evolutionary approach for modeling the equivalent circuit for electrochemical impedance spectroscopy. In Ruhul Amin Sarker, Robert G. Reynolds, Hussein Aly Abbass Amein, Kay Chen Tan, Robert Ian McKay, Daryl Leslie Essam, and Tom Gedeon, editors, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'03)*, volume 3, pages 1819–1825, Canberra, Australia, December 8–13, 2003. Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/CEC.2003.1299893. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/Cao_2003_Aeafmtecfeis.html.
119. David Kristoffersson. A software framework for genetic programming. Bachelor's thesis, Skellefteå, Sweden: Luleå University of Technology (LTU), Skellefteå Campus, Spring 2005. URL <http://epubl.ltu.se/1404-5494/2005/28/LTU-HIP-EX-0528-SE.pdf>. 2005:28 HIP. ISRN: LTU-HIP-EX-05/28-SE.
120. Thomas Weise and Kurt Geihs. Dgpf – an adaptable framework for distributed multi-objective search algorithms applied to the genetic programming of sensor networks. In Bogdan Filipič and Jurij Šilc, editors, *Proceedings of the Second International Conference on Bioinspired Optimization Methods and their Applications (BIOMA'06)*, Informacijska Družba (Information Society), pages 157–166, Ljubljana, Slovenia: Jožef Stefan International Postgraduate School, October 9–10, 2006. Ljubljana, Slovenia: Jožef Stefan Institute.
121. John R. Woodward. Evolving turing complete representations. In Ruhul Amin Sarker, Robert G. Reynolds, Hussein Aly Abbass Amein, Kay Chen Tan, Robert Ian McKay, Daryl Leslie Essam, and Tom Gedeon, editors, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'03)*, volume 2, pages 830–837, Canberra, Australia, December 8–13, 2003. Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/CEC.2003.1299753. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/Woodward_2003_Etcr.html.

Bibliography XIX



122. Astro Teller. Turing completeness in the language of genetic programming with indexed memory. In Zbigniew Michalewicz, James David Schaffer, Hans-Paul Schwefel, David B. Fogel, and Hiroaki Kitano, editors, *Proceedings of the First IEEE Conference on Evolutionary Computation (CEC'94), 1994 IEEE World Congress on Computation Intelligence (WCCI'94)*, pages 136–141, Orlando, FL, USA: Walt Disney World Dolphin Hotel, June 27–29, 1994. Piscataway, NJ, USA: IEEE Computer Society, Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/ICEC.1994.350027. URL <http://www.cs.cmu.edu/afs/cs/usr/astro/public/papers/Turing.ps>.
123. Astro Teller. Genetic programming, indexed memory, the halting problem, and other curiosities. In *Proceedings of the Seventh Florida Artificial Intelligence Research Symposium (FLAIRS'94)*, pages 270–274, Pensacola Beach, FL, USA, May 5–7, 1994. Los Alamitos, CA, USA: IEEE Computer Society Press. URL <http://www.cs.cmu.edu/afs/cs/usr/astro/public/papers/Curiosities.ps>.
124. David Jackson. Parsing and translation of expressions by genetic programming. In Hans-Georg Beyer, Una-May O'Reilly, Dirk V. Arnold, Wolfgang Banzhaf, Christian Blum, Eric W. Bonabeau, Erick Cantú-Paz, Dipankar Dasgupta, Kalyanmoy Deb, James A. Foster, Edwin D. de Jong, Hod Lipson, Xavier Llorà, Spiros Mancoridis, Martin Pelikan, Günther R. Raidl, Terence Soule, Jean-Paul Watson, and Eckart Zitzler, editors, *Proceedings of Genetic and Evolutionary Computation Conference (GECCO'05)*, pages 1681–1688, Washington, DC, USA: Loews L'Enfant Plaza Hotel, June 25–27, 2005. New York, NY, USA: ACM Press. doi: 10.1145/1068009.1068291.
125. Anna Derezińska. Advanced mutation operators applicable in c# programs. In Krzysztof Sacha, editor, *Software Engineering Techniques: Design for Quality – IFIP Working Conference on Software Engineering Techniques (SET'06)*, pages 283–288, Warsaw, Poland: Warsaw University, October 17–20, 2006. Berlin/Heidelberg: Springer-Verlag.
126. Carl Friedrich Gauß. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium*. Hamburg, Germany: Hamburgi Sumtibus Frid. Perthes et I. H. Besser, 1809. URL <http://books.google.de/books?id=ORU0AAAAQAAJ>.
127. Carl Friedrich Gauß. *Theoria Combinationis Observationum Erroribus Minimis Obnoxiae*, volume 5 (Classis Mathematicae) of *Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores*. Göttingen, Germany: Societas Regia Scientiarum Gottingensis and Göttingen, Germany: Gottingae: Henricum Dieterich, 1821. URL <http://books.google.de/books?id=ZQ80AAAAQAAJ>.
128. Carl Friedrich Gauß. *Supplementum Theoria Combinationis Observationum Erroribus Minimis Obnoxiae*, volume 6 (Classis Mathematicae) of *Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores*. Göttingen, Germany: Societas Regia Scientiarum Gottingensis and Göttingen, Germany: Gottingae: Henricum Dieterich, 1826.

Bibliography XX



129. Adrien-Marie Legendre. *Nouvelles Méthodes pour la Détermination des Orbites des Comètes*. Paris, France: Firmin Didot, Librairie pour les Mathématiques, la Marine, l'Architecture, et les Éditions stéréotypes, 1805. URL <http://books.google.de/books?id=PRc0AAAAQAAJ>.
130. Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2:164–168, 1944.
131. Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441, June 1963. ISSN 0036-1399. doi: 10.1137/0111030.
132. Edgar Anderson. The irises of the gaspè peninsula. *Bulletin of the American Iris Society*, 59:2–5, 1935.
133. James C. Bezdek, James M. Keller, Raghu Krishnapuram, Ludmila I. Kuncheva, and Nikhil R. Pal. Will the real iris data please stand up? *IEEE Transactions on Fuzzy Systems (TFS)*, 7(3):368–369, June 1999. doi: 10.1109/91.771092.
134. David J. Montana. Strongly typed genetic programming. BBN Technical Report #7866, Cambridge, MA, USA: Bolt Beranek and Newman Inc. (BBN), May 7, 1993. URL <http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/ftp.io.com/papers/stgp.ps.Z>.
135. David J. Montana. Strongly typed genetic programming. BBN Technical Report #7866, Cambridge, MA, USA: Bolt Beranek and Newman Inc. (BBN), March 25, 1994. URL <http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/ftp.io.com/papers/stgp2.ps.Z>.
136. Thomas D. Haynes, Dale A. Schoenfeld, and Roger L. Wainwright. Type inheritance in strongly typed genetic programming. In Peter John Angelino and Kenneth E. Kinnear, Jr, editors, *Advances in Genetic Programming II*, Bradford Books, chapter 18, pages 359–376. Cambridge, MA, USA: MIT Press, October 26, 1996. URL <http://www.mcs.utulsa.edu/~rogerw/papers/Haynes-hier.pdf>.
137. Thomas D. Haynes, Roger L. Wainwright, Sandip Sen, and Dale A. Schoenfeld. Strongly typed genetic programming in evolving cooperation strategies. In Larry J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA'95)*, pages 271–278, Pittsburgh, PA, USA: University of Pittsburgh, July 15–19, 1995. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://www.mcs.utulsa.edu/~rogerw/papers/Haynes-icga95.pdf>.
138. William Benjamin Langdon and Riccardo Poli. Fitness causes bloat: Mutation. In Wolfgang Banzhaf, Riccardo Poli, Marc Schoenauer, and Terence Claus Fogarty, editors, *Proceedings of the First European Workshop on Genetic Programming (EuroGP'98)*, volume 1391/1998 of *Lecture Notes in Computer Science (LNCS)*, pages 37–48, Paris, France, April 14–15, 1998. Berlin, Germany: Springer-Verlag GmbH. URL <http://citeseer.ist.psu.edu/langdon98fitness.html>.

Bibliography XXI



139. Sean Luke and Liviu Panait. A comparison of bloat control methods for genetic programming. *Evolutionary Computation*, 14(3):309–344, Fall 2006. doi: 10.1162/evco.2006.14.3.309. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.159.1580>.
140. Riccardo Poli. A simple but theoretically-motivated method to control bloat in genetic programming. In Conor Ryan, Terence Soule, Maarten Keijzer, Edward P. K. Tsang, Riccardo Poli, and Ernesto Jorge Fernandes Costa, editors, *Proceedings of the 6th European Conference on Genetic Programming (EuroGP'03)*, volume 2610/2003 of *Lecture Notes in Computer Science (LNCS)*, pages 204–217, Wivenhoe Park, Colchester, Essex, UK: University of Essex, Departments of Mathematical and Biological Sciences, April 14–16, 2003. Berlin, Germany: Springer-Verlag GmbH. URL <http://cswww.essex.ac.uk/staff/poli/papers/eurogp2003.pdf>.
141. Peter John Angeline. Subtree crossover causes bloat. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David Edward Goldberg, Hitoshi Iba, and Rick L. Riolo, editors, *Proceedings of the Third Annual Genetic Programming Conference (GP'98)*, pages 745–752, Madison, WI, USA: University of Wisconsin, July 22–25, 1998. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
142. Sean Luke. Code growth is not caused by introns. In L. Darrell Whitley, editor, *Late Breaking Papers at Genetic and Evolutionary Computation Conference (GECCO'00 LBP)*, pages 228–235, Las Vegas, NV, USA: Riviera Hotel and Casino, July 10–12, 2000. Menlo Park, CA, USA: AAAI Press. URL <http://www.cs.gmu.edu/~sean/papers/intronpaper.pdf>.
143. Walter Alden Tackett. *Recombination, Selection, and the Genetic Construction of Computer Programs*. PhD thesis, Los Angeles, CA, USA: University of Southern California (USC), Department of Electrical Engineering Systems, April 17, 1994. URL <http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/ftp.io.com/papers/watphd.tar.Z>. CENG Technical Report 94-13.
144. Peter W. H. Smith and Kim Harries. Code growth, explicitly defined introns, and alternative selection schemes. *Evolutionary Computation*, 6(4):339–360, Winter 1998. doi: 10.1162/evco.1998.6.4.339. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.24.2202>.
145. Tobias Bickle and Lothar Thiele. Genetic programming and redundancy. In J. Hopf, editor, *Genetic Algorithms within the Framework of Evolutionary Computation, Workshop at KI-94*, pages 33–38, September 1994. URL <ftp://ftp.tik.ee.ethz.ch/pub/people/thiele/paper/B1Th94.ps>.
146. Tobias Bickle. *Theory of Evolutionary Algorithms and Application to System Synthesis*. PhD thesis, Zürich, Switzerland: Eidgenössische Technische Hochschule (ETH) Zürich, November 1996. URL <http://www.tik.ee.ethz.ch/~tec/publications/bli97a/diss.ps.gz>.

Bibliography XXII



147. Peter Nordin and Wolfgang Banzhaf. Complexity compression and evolution. In Larry J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA'95)*, pages 310–317, Pittsburgh, PA, USA: University of Pittsburgh, July 15–19, 1995. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://web.cs.mun.ca/~banzhaf/papers/icga95-1.pdf>.
148. Nicholas Freitag McPhee and Justin Darwin Miller. Accurate replication in genetic programming. In Larry J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA'95)*, pages 303–309, Pittsburgh, PA, USA: University of Pittsburgh, July 15–19, 1995. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.mrs.umn.edu/~mcpheee/Research/Accurate_replication.ps.
149. Justinian P. Rosca. Generality versus size in genetic programming. In John R. Koza, David Edward Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Proceedings of the First Annual Conference of Genetic Programming (GP'96)*, Bradford Books, pages 381–387, Stanford, CA, USA: Stanford University, July 28–31, 1996. Cambridge, MA, USA: MIT Press. URL <ftp://ftp.cs.rochester.edu/pub/u/roscas/gp/96.gp.ps.gz>.
150. Wolfgang Banzhaf, Peter Nordin, Robert E. Keller, and Frank D. Francone. *Genetic Programming: An Introduction – On the Automatic Evolution of Computer Programs and Its Applications*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. and Heidelberg, Germany: dpunkt.verlag, November 30, 1997. ISBN 1-558-60510-X, 3-920993-58-6, and 978-1-558-60510-7. URL <http://books.google.de/books?id=1697qefFdtIC>.
151. William Benjamin Langdon, Terence Soule, Riccardo Poli, and James A. Foster. The evolution of size and shape. In Lee Spector, William Benjamin Langdon, Una-May O'Reilly, and Peter John Angeline, editors, *Advances in Genetic Programming III*, Bradford Books, chapter 8, pages 163–190. Cambridge, MA, USA: MIT Press, July 16, 1996. URL <http://www.cs.bham.ac.uk/~wbl/aigp3/ch08.ps.gz>.
152. Terence Soule and James A. Foster. Removal bias: a new cause of code growth in tree based evolutionary programming. In Patrick K. Simpson, editor, *The 1998 IEEE International Conference on Evolutionary Computation (CEC'98), 1998 IEEE World Congress on Computation Intelligence (WCCI'98)*, pages 781–186, Anchorage, AK, USA: Egan Civic & Convention Center and Anchorage Hilton Hotel, May 4–9, 1998. Piscataway, NJ, USA: IEEE Computer Society, Piscataway, NJ, USA: IEEE Computer Society. doi: 10.1109/ICEC.1998.700151. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.3659>.
153. Stefan Bleuler, Martin Brack, Lothar Thiele, and Eckart Zitzler. Multiobjective genetic programming: Reducing bloat using spea2. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'01)*, pages 536–543, Gangnam-gu, Seoul, Korea: COEX, World Trade Center, May 27–30, 2001. Piscataway, NJ, USA: IEEE Computer Society. URL <ftp://ftp.tik.ee.ethz.ch/pub/people/zitzler/BBTZ2001b.ps.gz>.

Bibliography XXIII

154. Edwin D. de Jong, Richard A. Watson, and Jordan B. Pollack. Reducing bloat and promoting diversity using multi-objective methods. In Lee Spector, Erik D. Goodman, Annie S. Wu, William Benjamin Langdon, Hans-Michael Voigt, Mitsuo Gen, Sandip Sen, Marco Dorigo, Shahram Pezeshk, Max H. Garzon, and Edmund K. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'01)*, pages 11–18, San Francisco, CA, USA: Holiday Inn Golden Gateway Hotel, July 7–11, 2001. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/jong_2001_gecco.html.
155. Riccardo Poli, William Benjamin Langdon, and Nicholas Freitag McPhee. *A Field Guide to Genetic Programming*. London, UK: Lulu Enterprises UK Ltd, March 2008. ISBN 1-4092-0073-6 and 978-1-4092-0073-4. URL http://www.lulu.com/items/volume_63/2167000/2167025/2/print/book.pdf. With contributions by John R. Koza.
156. Wolfgang Banzhaf. Genetic programming for pedestrians. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms (ICGA'93)*, pages 17–21, Urbana-Champaign, IL, USA: University of Illinois, July 17–21, 1993. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/banzhaf_mrl.html. Also: MRL Technical Report 93-03.
157. Markus F. Brämeier and Wolfgang Banzhaf. A comparison of linear genetic programming and neural networks in medical data mining. *IEEE Transactions on Evolutionary Computation (IEEE-EC)*, 5(1):17–26, 2001. doi: 10.1109/4235.910462. URL http://web.cs.mun.ca/~banzhaf/papers/ieee_taec.pdf.
158. Timothy Perkis. Stack based genetic programming. In Zbigniew Michalewicz, James David Schaffer, Hans-Paul Schwefel, David B. Fogel, and Hiroaki Kitano, editors, *Proceedings of the First IEEE Conference on Evolutionary Computation (CEC'94), 1994 IEEE World Congress on Computation Intelligence (WCCI'94)*, volume 1, pages 148–153, Orlando, FL, USA: Walt Disney World Dolphin Hotel, June 27–29, 1994. Piscataway, NJ, USA: IEEE Computer Society, Piscataway, NJ, USA: IEEE Computer Society. URL ftp://coast.cs.purdue.edu/pub/doc/genetic_algorithms/GP/Stack-Genetic-Programming.ps.gz.
159. Stan Openshaw and Ian Turton. Building new spatial interaction models using genetic programming. In Terence Claus Fogarty, editor, *Proceedings of the Workshop on Artificial Intelligence and Simulation of Behaviour, International Workshop on Evolutionary Computing, Selected Papers (AISB'94)*, volume 865/1994 of *Lecture Notes in Computer Science (LNCS)*, Leeds, UK, April 11–13, 1994. Chichester, West Sussex, UK: Society for the Study of Artificial Intelligence and the Simulation of Behaviour (SSAISB), Berlin, Germany: Springer-Verlag GmbH. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/openshaw_1994_bnsim.html.

Bibliography XXIV



160. Ronald L. Crepeau. Genetic evolution of machine language software. In *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, pages 121–134, Tahoe City, CA, USA, July 9, 1995. San Mateo, CA, USA: Morgan Kaufmann. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/crepeau_1995_GEMS.html.
161. Kwong Sak Leung, Kin Hong Lee, and Sin Man Cheang. Genetic parallel programming – evolving linear machine codes on a multiple-alu processor. In S. Yaacob, Meenakshi Nagarajan, and A. Chekima, editors, *Proceedings of International Conference on Artificial Intelligence in Engineering and Technology (ICAIET'02)*, pages 207–213, Kota Kinabalu, Sabah, Malaysia: University of Malaysia Sabah, June 17–18, 2002.
162. Sin Man Cheang, Kwong Sak Leung, and Kin Hong Lee. Genetic parallel programming: Design and implementation. *Evolutionary Computation*, 14(2):129–156, Summer 2006. doi: 10.1162/evco.2006.14.2.129.
163. James A. Foster. Review: Discipulus: A commercial genetic programming system. *Genetic Programming and Evolvable Machines*, 2(2):201–203, June 2001. doi: 10.1023/A:1011516717456.
164. F. Corno, G. Cumani, M. Sonza Reorda, and Giovanni Squillero. Efficient machine-code test-program induction. In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Hitoshi Iba, Paul Marrow, and Mark Shackleton, editors, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'02), 2002 IEEE World Congress on Computation Intelligence (WCCI'02)*, volume 1-2, pages 1486–1491, Honolulu, HI, USA: Hilton Hawaiian Village Hotel (Beach Resort & Spa), May 12–17, 2002. Piscataway, NJ, USA: IEEE Computer Society, Los Alamitos, CA, USA: IEEE Computer Society Press. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/corno_2002_emctpi.html.
165. Giovanni Squillero. Microgp – an evolutionary assembly program generator. *Genetic Programming and Evolvable Machines*, 6(3):247–263, September 2005. doi: 10.1007/s10710-005-2985-x.
166. Peter Nordin, Frank D. Francone, and Wolfgang Banzhaf. Explicitly defined introns and destructive crossover in genetic programming. In *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, pages 6–22, Tahoe City, CA, USA, July 9, 1995. San Mateo, CA, USA: Morgan Kaufmann. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/nordin_1995_introns.html.
167. Peter Nordin. A compiling genetic programming system that directly manipulates the machine code. In Kenneth E. Kinnear, Jr, editor, *Advances in Genetic Programming I*, Bradford Books, chapter 14, pages 311–331. Cambridge, MA, USA: MIT Press, April 7, 1994. Machine code GP Sun Spark and i868.
168. Peter Nordin. *Evolutionary Program Induction of Binary Machine Code and its Applications*. PhD thesis, Dortmund, North Rhine-Westphalia, Germany: Universität Dortmund, Fachbereich Informatik.

Bibliography XXV



169. Peter Nordin and Wolfgang Banzhaf. Evolving turing-complete programs for a register machine with self-modifying code. In Larry J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA'95)*, pages 318–325, Pittsburgh, PA, USA: University of Pittsburgh, July 15–19, 1995. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/Nordin_1995_tcp.html.
170. Peter Nordin, Wolfgang Banzhaf, and Frank D. Francone. Efficient evolution of machine code for cisc architectures using instruction blocks and homologous crossover. In Lee Spector, William Benjamin Langdon, Una-May O'Reilly, and Peter John Angeline, editors, *Advances in Genetic Programming III*, Bradford Books, chapter 12, pages 275–299. Cambridge, MA, USA: MIT Press, July 16, 1996. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/nordin_1999_aigp3.html.
171. Peter Nordin, Frank Hoffmann, Frank D. Francone, and Markus F. Brämeier. Aim-gp and parallelism. In Peter John Angeline, Zbigniew Michalewicz, Marc Schoenauer, Xin Yao, and Ali M. S. Zalzala, editors, *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'99)*, volume 1-3, pages 1059–1066, Washington, DC, USA: Mayflower Hotel, July 6–9, 1999. Piscataway, NJ, USA: IEEE Computer Society. URL http://www.cs.mun.ca/~banzhaf/papers/cec_parallel.pdf.
172. Eduard Lukschndl, Magnus Holmlund, Eric Moden, Mats G. Nordahl, and Peter Nordin. Induction of java bytecode with genetic programming. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference (GP'98 LBP)*, pages 135–142, Madison, WI, USA: University of Wisconsin, June 22–25, 1998. Stanford, CA, USA: Stanford University Bookstore, Stanford University.
173. Eduard Lukschndl, Henrik Borgvall, Lars Nohle, Mats G. Nordahl, and Peter Nordin. Evolving routing algorithms with the jbgp-system. In Riccardo Poli, Hans-Michael Voigt, Stefano Cagnoni, David Wolfe Corne, George D. Smith, and Terence Claus Fogarty, editors, *Proceedings of the First European Workshops on Evolutionary Image Analysis, Signal Processing and Telecommunications: EvolASP'99, EuroEcTel'99 (EvoWorkshops'99)*, volume 1596/1999 of *Lecture Notes in Computer Science (LNCS)*, pages 193–202, Göteborg, Sweden, May 26–27, 1999. Berlin, Germany: Springer-Verlag GmbH. doi: 10.1007/10704703_16.
174. Eduard Lukschndl, Henrik Borgvall, Lars Nohle, Mats G. Nordahl, and Peter Nordin. Distributed java bytecode genetic programming with telecom applications. In Riccardo Poli, Wolfgang Banzhaf, William Benjamin Langdon, Julian Francis Miller, Peter Nordin, and Terence Claus Fogarty, editors, *Proceedings of the Third European Conference on Genetic Programming (EuroGP'00)*, volume 1802/2000 of *Lecture Notes in Computer Science (LNCS)*, pages 316–325, Edinburgh, Scotland, UK, April 15–16, 2000. France: EvoGP, The Genetic Programming Working Group of EvoNET, The Network of Excellence in Evolutionary Computing, Berlin, Germany: Springer-Verlag GmbH.

Bibliography XXVI



175. Eduard Lukschndl, Peter Nordin, and Mats G. Nordahl. Using the java method evolver for load balancing in communication networks. In L. Darrell Whitley, editor, *Late Breaking Papers at Genetic and Evolutionary Computation Conference (GECCO'00 LBP)*, pages 236–239, Las Vegas, NV, USA: Riviera Hotel and Casino, July 10–12, 2000. Menlo Park, CA, USA: AAAI Press.
176. Brad Harvey, James A. Foster, and Deborah Frincke. Towards byte code genetic programming. In Wolfgang Banzhaf, Jason M. Daida, Ágoston E. Eiben, Max H. Garzon, Vasant Honavar, Mark J. Jakela, and Robert Elliott Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)*, volume 2, page 1234, Orlando, FL, USA, July 13–17, 1999. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL http://www.cs.bham.ac.uk/~wbl/biblio/gp-html/harvey_1999_TBCGP.html.
177. Brad Harvey, James A. Foster, and Deborah Frincke. Byte code genetic programming. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference (GP'98 LBP)*, Madison, WI, USA: University of Wisconsin, June 22–25, 1998. Stanford, CA, USA: Stanford University Bookstore, Stanford University. URL <http://www.csds.uidaho.edu/deb/jvm.pdf>.
178. Stefan Klahold, Steffen Frank, Robert E. Keller, and Wolfgang Banzhaf. Exploring the possibilities and restrictions of genetic programming in java bytecode. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference (GP'98 LBP)*, pages 120–124, Madison, WI, USA: University of Wisconsin, June 22–25, 1998. Stanford, CA, USA: Stanford University Bookstore, Stanford University.
179. Riccardo Poli. Parallel distributed genetic programming. Technical Report CSRP-96-15, Birmingham, UK: University of Birmingham, School of Computer Science, September 1996. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.7666>.
180. Riccardo Poli. Evolution of recursive transition networks for natural language recognition with parallel distributed genetic programming. Technical Report CSRP-96-19, .
181. Riccardo Poli. Some steps towards a form of parallel distributed genetic programming. . URL <http://cswww.essex.ac.uk/staff/poli/papers/Poli-WSC1-1996.pdf>. Originally published as technical report, CSRP-96-14, University of Birmingham, School of Computer Science, 1996.

Bibliography XXVII

182. Riccardo Poli. Evolution of recursive transition networks for natural language recognition with parallel distributed genetic programming. In David Wolfe Corne and Jonathan L. Shapiro, editors, *Proceedings of the Workshop on Artificial Intelligence and Simulation of Behaviour, International Workshop on Evolutionary Computing, Selected Papers (AISB'97)*, volume 1305/1997 of *Lecture Notes in Computer Science (LNCS)*, pages 163–177, Manchester, UK, April 7–8, 1997. Berlin, Germany: Springer-Verlag GmbH. URL <http://cswww.essex.ac.uk/staff/rpoli/papers/Poli-AISB-1997.pdf>.
183. Riccardo Poli. Discovery of symbolic, neuro-symbolic and neural networks with parallel distributed genetic programming. In George D. Smith, Nigel C. Steele, and Rudolf F. Albrecht, editors, *Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms (ICANNGA'97)*, SpringerComputerScience, Vienna, Austria, April 1–4, 1997. Vienna, Austria: Springer Verlag GmbH. URL <http://cswww.essex.ac.uk/staff/rpoli/papers/Poli-ICANNGA1997.pdf>.
184. Yuval Davidor. Epistasis variance: A viewpoint on ga-hardness. In Bruce M. Spatz and Gregory J. E. Rawlins, editors, *Proceedings of the First Workshop on Foundations of Genetic Algorithms (FOGA'90)*, pages 23–35, Bloomington, IN, USA: Indiana University, Bloomington Campus, July 15–18, 1990. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
185. Lee Altenberg. Nk fitness landscapes. In Thomas Bäck, David B. Fogel, and Zbigniew Michalewicz, editors, *Handbook of Evolutionary Computation*, Computational Intelligence Library, chapter B2.7.2. New York, NY, USA: Oxford University Press, Inc., Dirac House, Temple Back, Bristol, UK: Institute of Physics Publishing Ltd. (IOP), and Boca Raton, FL, USA: CRC Press, Inc., January 1, 1997. URL <http://www.cmi.univ-mrs.fr/~pardoux/LeeNKFL.pdf>.
186. Bart Naudts and Alain Verschoren. Epistasis on finite and infinite spaces. In *8th International Conference on Systems Research, Informatics and Cybernetics (InterSymp'96)*, pages 19–23, Baden-Baden, Baden-Württemberg, Germany, August 14–18, 1996. Tecumseh, ON, Canada: International Institute for Advanced Studies in Systems Research and Cybernetic (IIAS). URL <http://citeserx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.6455>.
187. Colin R. Reeves and Christine C. Wright. Epistasis in genetic algorithms: An experimental design perspective. In Larry J. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA'95)*, pages 217–224, Pittsburgh, PA, USA: University of Pittsburgh, July 15–19, 1995. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. URL <http://citeserx.ist.psu.edu/viewdoc/summary?doi=10.1.1.39.29>.
188. Thomas Weise. *Global Optimization Algorithms – Theory and Application*. Germany: it-weise.de (self-published), 2009. URL <http://www.it-weise.de/projects/book.pdf>.

Bibliography XXVIII



189. William Bateson. *Mendel's Principles of Heredity*. Kessinger Publishing's® Rare Reprints. Cambridge, UK: Cambridge University Press, 1909. ISBN 1428648194 and 9781428648197. URL <http://books.google.de/books?id=WWZfDQ1jn8gC>.
190. Sir Ronald Aylmer Fisher. The correlations between relatives on the supposition of mendelian inheritance. *Philosophical Transactions of the Royal Society of Edinburgh*, 52:399–433, October 1, 1918. URL <http://www.library.adelaide.edu.au/digitised/fisher/9.pdf>.
191. Patrick C. Phillips. The language of gene interaction. *Genetics*, 149(3):1167–1171, July 1998. URL <http://www.genetics.org/cgi/reprint/149/3/1167.pdf>.
192. Jay L. Lush. Progeny test and individual performance as indicators of an animal's breeding value. *Journal of Dairy Science (JDS)*, 18(1):1–19, January 1935. URL <http://jds.fass.org/cgi/reprint/18/1/1>.
193. Thomas Weise and Ke Tang. Evolving distributed algorithms with genetic programming. *IEEE Transactions on Evolutionary Computation (IEEE-EC)*, 16(2):242–265, April 2012. doi: 10.1109/TEVC.2011.2112666.
194. Thomas Weise. *Evolving Distributed Algorithms with Genetic Programming*. PhD thesis, Kassel, Hesse, Germany: University of Kassel, Fachbereich 16: Elektrotechnik/Informatik, Distributed Systems Group, May 4, 2009. Won the Dissertation Award of The Association of German Engineers (Verein Deutscher Ingenieure, VDI).
195. Thomas Weise and Michael Zapf. Evolving distributed algorithms with genetic programming: Election. In Lihong Xu, Erik D. Goodman, and Yongsheng Ding, editors, *Proceedings of the First ACM/SIGEVO Summit on Genetic and Evolutionary Computation (GEC'09)*, pages 577–584, Shanghai, China: Hua-Ting Hotel & Towers, June 12–14, 2009. New York, NY, USA: ACM Press. doi: 10.1145/1543834.1543913.
196. Thomas Weise. Traditional and rule-based genetic programming, December 22, 2009.
197. Thomas Weise. Internal cooperation/brainstorming session presentation – evolving distributed algorithms with genetic programming, December 15, 2008.