10

Symbolic regression, Genetic programming

The plan

- Symbolic Regression (SR)
- Genetic Programming (GP)
- SR + GP

  (in short: SR is an optimization problem and GP is one of the ways how to solve it)

Symbolic regression (SR)

- Linear regression

$$\hat{y} = a_1 + a_2 x + a_3 x_2 + a_4 x_3 + a_5 x_4 + a_6 x_5 + a_7 x_6 + a_8 x_7$$

$$\hat{y} = a_1 + a_2 x + \cancel{a_3 x_2} + \cancel{a_4 x_3} + a_5 x_4 + \cancel{a_6 x_5} + \cancel{a_7 x_6} + a_8 x_7 \quad \leftarrow$$ *(subset selection)*

- Symbolic regression enables working with regression models of any form

$$\hat{y} = 5x_1 \tan\left(x_2 \sqrt{2 + x_3}\right) + 3\sin\left(\frac{x_1 + 10}{x_2 + x_3 + \frac{1}{x_1}}\right)$$

Symbolic regression (SR)

- "Conventional" regression methods optimize parameters of a pre-specified model *(e.g., least squares method in linear regression)*

  + may also optimize feature subset to be included in the model *(i.e., do feature subset selection)*
  - The general structure of the model is fixed

- Symbolic regression avoids imposing prior assumptions about necessary model's form, instead it infers the model's form from the data
- In this approach we have much larger space to search – in fact the search space is infinite (provided that the model complexity isn't artificially limited)

Symbolic regression (SR)

- Any symbolic regression model can be represented as a tree where interior nodes contain mathematical functions and leaf nodes contain variables and constants (terminals)
- SR works with two sets of "symbols" :
  - A set of functions – all allowed functions (with at least one, usually two, input arguments), for example:

    $\{+, -, *, \%, \sin, \cos, \exp, \min, \max\}$,

    where "%" is "protected division" ($1 \% 0 = 0$)

  - A set of terminals – all allowed input variables and constants, for example:

    $\{x_1, x_2, x_3, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

    or $\{x_1, x_2, x_3, \Re\}$

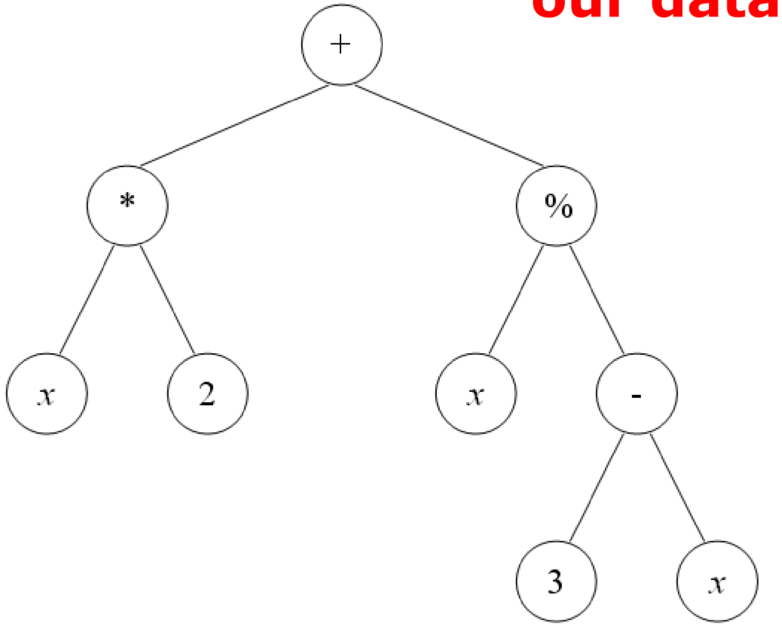
    Fixed random constant

Symbolic regression example

- Functions: $\{+, -, *, \%\}$
  - Always in interior nodes
- Terminals: $\{x, 1, 2, 3, 4, 5\}$
  - Always in leaf nodes
- Example tree:

The main problem stays the same:

how to find the "best" regression model for our data?



If all used functions have two input arguments, then in symbolic regression each non-terminal node will have exactly two child nodes. We can also use functions with one or three input arguments but more arguments are rarely used and usually not needed because such branching typically can be replaced by deeper binary branching.

- The corresponding equation: $\hat{y} = 2x + \dfrac{x}{3-x}$

Genetic programming (GP)

- Genetic programming is an evolutionary approach that extends genetic algorithms for optimization of variable-length non-linear (hierarchical) structures

- GP can also be defined as automated programming
  - GP evolves (optimizes) “computer programs” (sequences of steps) using evolutionary principles so that they perform well on a predefined task

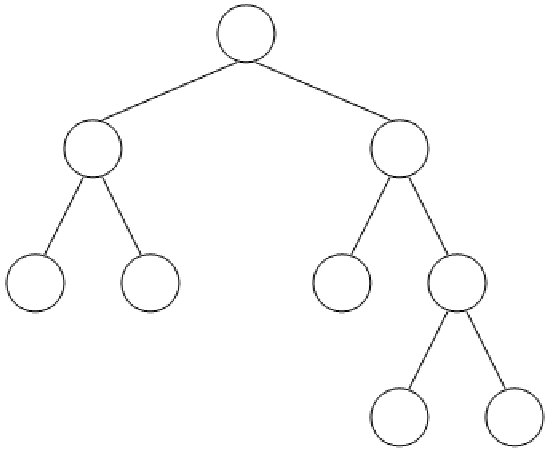
Genetic programming (GP)

- GP is based on Genetic Algorithms
  - Evolution. Population of individuals. Selection, crossover, mutation. Etc.
- In “conventional” genetic algorithms, solutions of an optimization problem are linear and of fixed length

1	0	1	1	1	0

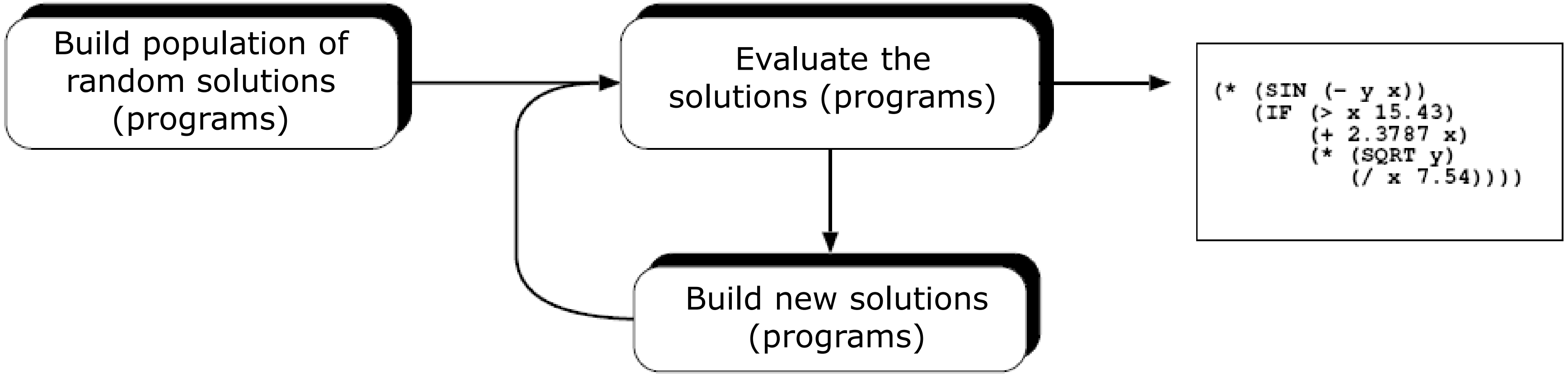
C	B	E	A	F	D

- In genetic programming, solutions usually are hierarchical structures - trees



Genetic programming (GP)

- Initially, GP was implemented for automated evolving of code in the programming language Lisp


Build population of random solutions (programs)
Evaluate the solutions (programs)
Build new solutions (programs)

```
(* (SIN (- y x))
   (IF (> x 15.43)
       (+ 2.3787 x)
       (* (SQRT y)
          (/ x 7.54))))
```

What do we need to be able to use GP

- The most important components to define in order to be able to use Genetic Programming (or other similar algorithms) are:
  - How do we encode the structure that must be optimized (so that it can be manipulated using unified and simple crossover and mutation operators)
  - How do we quantitatively estimate the quality of the structure – its “nearness” to our optimization goal (*development of the fitness function*)

SR + GP

- The general algorithm:
  1. Randomly generate initial generation ("*primordial soup*")
  2. Evaluate all individuals (fitness estimation)
  3. Selection – select individuals for reproduction
  4. Crossover – selected individuals produce offspring
  5. Mutation – small random changes in offspring
  6. With the new generation go to step 2

Before we start, we predefine the sets of functions and terminals.
In my next examples:

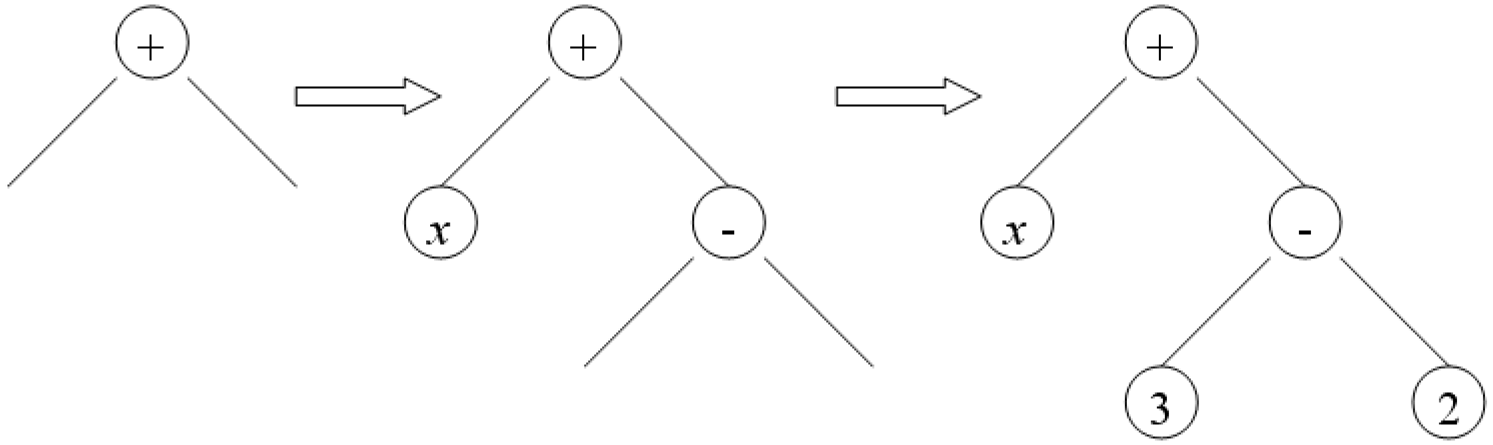
Functions: $\{+, -, *, \%\}$

Terminals: $\{x, 1, 2, 3, 4, 5\}$

And the population will consist of 4 individuals.
We'll look at only one of the possible SR+GP implementations.

SR + GP

- Build the initial generation (primordial soup)
  - Building a random tree (usually with a preset max depth)

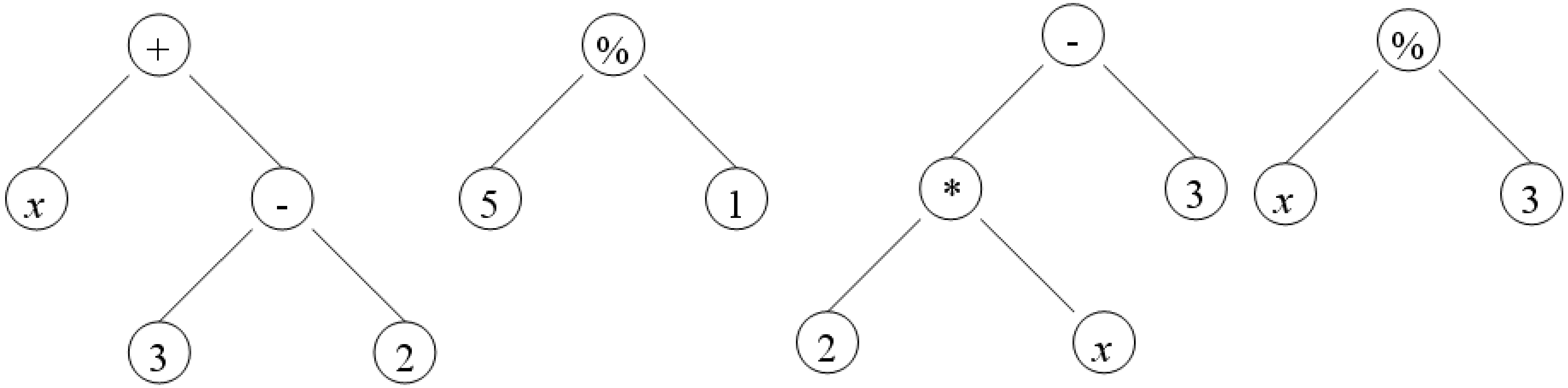


The corresponding equation:

$$\hat{y} = x + (3 - 2)$$

SR + GP

- All individuals of the initial generation are ready



- Fitness evaluation of all individuals

Individual 1	Individual 2	Individual 3	Individual 4
$\hat{y} = x + (3 - 2)$	$\hat{y} = 5 \% 1$	$\hat{y} = 2x - 3$	$\hat{y} = x \% 3$
F = 13	F = 14	F = 21	F = 6

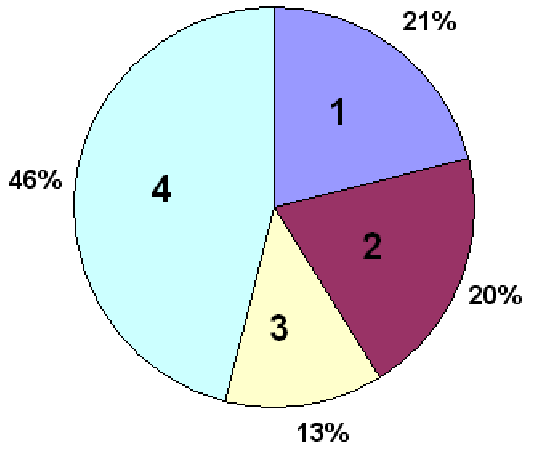
Evaluation of a tree as a regression model, is performed by computing its training error in training data, e.g., as absolute or quadratic error or MDL etc.

SR + GP

Selection

Individual 1	Individual 2	Individual 3	Individual 4
$\hat{y} = x + (3 - 2)$	$\hat{y} = 5 \% 1$	$\hat{y} = 2x - 3$	$\hat{y} = x \% 3$
F = 13	F = 14	F = 21	F = 6

In Genetic Programming we can use the same selection method that we already know from Genetic Algorithms: roulette wheel selection, tournament selection a.o.



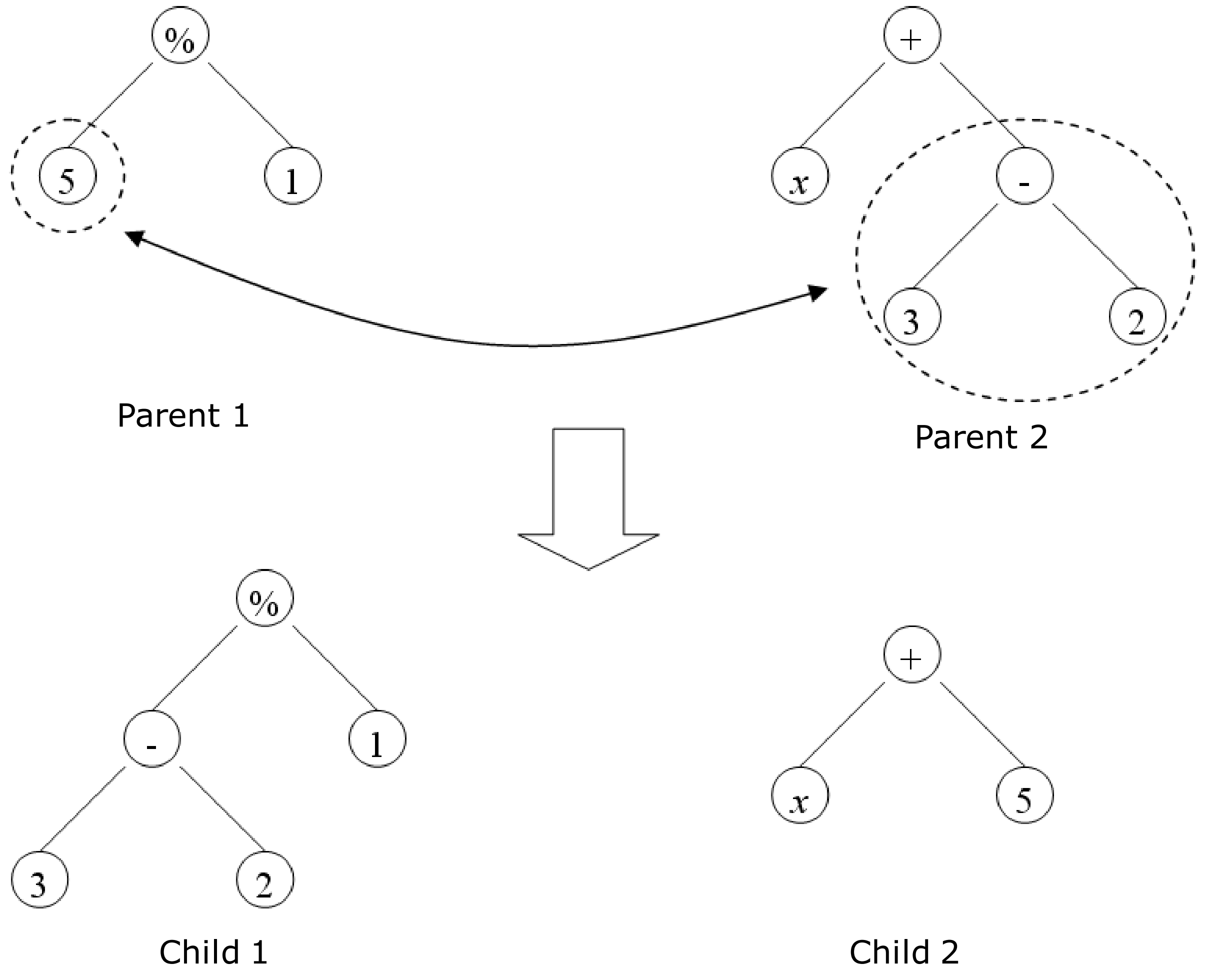
Roulette wheel selection

Example of calculations

	F	1 / F	Probability
Individual 1	13	0.0769	0.21
Individual 2	14	0.0714	0.20
Individual 3	21	0.0476	0.13
Individual 4	6	0.1667	0.46
Sum	54	0.3626	1.00

SR + GP

- Crossover (a random subtree of two selected trees are swapped)

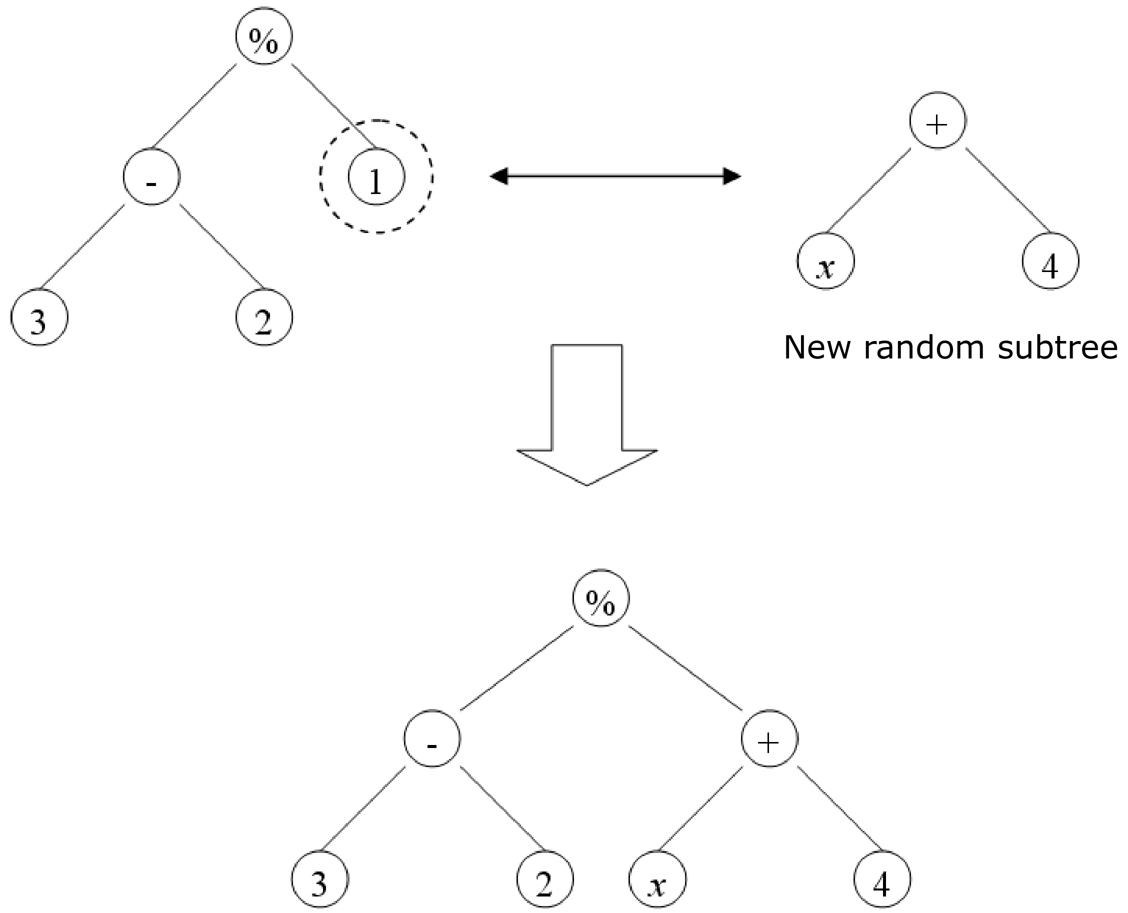


(other methods available)

(may also be implemented so that only one child is generated from two parents)

SR + GP

- Mutation: do small random changes: (1) a randomly chosen small subtree is replaced with a new small randomly generated subtree; (2) one random terminal replaced by another



(other methods available)

SR + GP

- When the new generation of individuals is ready, go back to the evaluation step and repeat the whole process with the new generation
- GP is usually terminated when a predefined number of generations (iterations) are done
- Finally, the best found solution is outputted (*optimality of course is not guaranteed*)

Comments about GP

- Many hyperparameters:
  - The number of individuals in population (a fixed number, usually a few tens or hundreds)
  - Number of generations (tens, hundreds, thousands)
  - Probability of crossover (usually big)
  - Probability of mutation (usually small)
  - Max tree depth (a number of hyperparameters for different purposes)
  - ...
- There are different methods for selection, crossover, and mutation + other stuff like cloning/elitism etc.

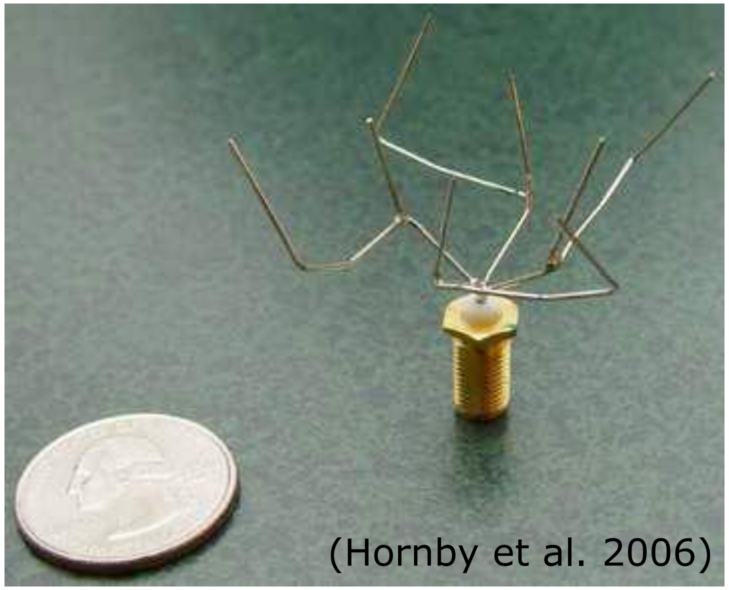
Other methods/approaches

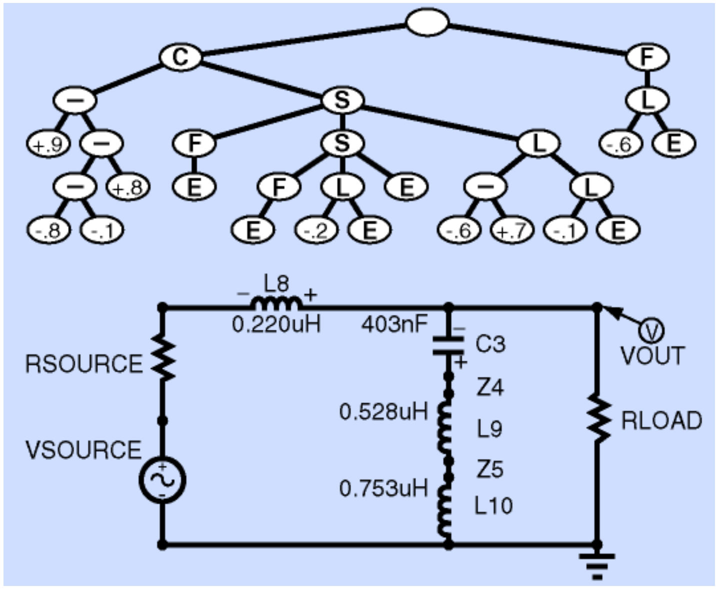
- There are other methods which can be used instead of Genetic Programming:
  - Grammatical Evolution
  - Gene Expression Programming
  - and others

- There is also Linear GP approach which is developed for optimization of structures that are closer to those in the typical programming languages – you optimize a sequence of instructions.
  - But this is not "ordinary" Genetic Algorithm because in Linear GP the length of chromosomes is not fixed.

Some other applications of GP

- Source code generation for various languages
  - There are publications on Lisp/Assembler/C/C++/Java/...
- Image compression
  - A Symbolic Regression model predicts color of next pixel from colors of previous pixels. In the file, store the model and its prediction mistakes.
- Designing of radio antennae
- Automatic design of electronic circuits
- and many other applications


(Hornby et al. 2006)



Literature

- Freely downloadable book:

  Poli R., Langdon W.B., McPhee N.F. A Field Guide to Genetic Programming, 2008

  http://www.gp-field-guide.org.uk/