# COMP 4752
# Computational Intelligence


**Lecture 19**
Genetic Programming

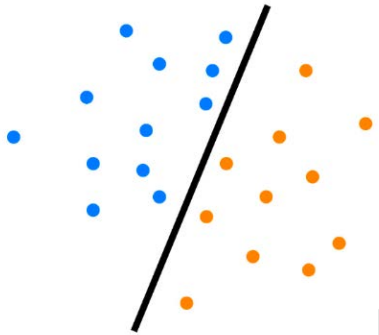
# Motivation for GP

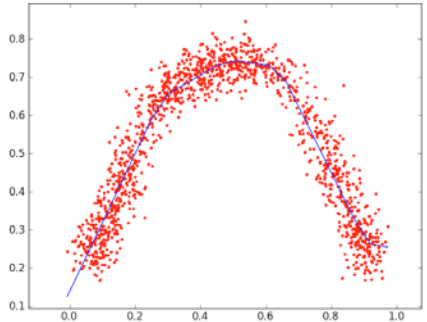
- Automatic Programming
  - Computers learn to program themselves
- Machine Learning
  - Study of computer algorithms that improve automatically through experience
  - Classification / Regression, Function Approximation
- Genetic Programming
  - Induce a population of computer programs that improve automatically as they experience the data on which they are trained

# Machine Learning

- Training, testing, and cross validation
- Supervised Learning
  - Each training instance is given correct output
- Unsupervised Learning
  - No correct inputs given, it discovers patterns
- Reinforcement Learning
  - General reward signal used for quality

# Machine Learning






- Classification
  - Supervised, discrete output
- Regression
  - Supervised, continuous output
- Clustering
  - Unsupervised
- Dimensionality Reduction / Function Approx.
  - Mapping high-dimensional inputs into a lower-dimensional space

# GP Overview

- Developed in USA in 1990's
- John Khoza
- Typically applied to machine learning tasks
- Attributed Features:
  - Competes with neural nets and alike
  - Needs huge populations, and is slow
  - Non-linear chromosomes (trees / graphs)

# GP Technical Summary

- Representation: Tree structures
- Recombination: Exchange of subtrees
- Mutation: Random change in trees
- Parent Selection: Fitness proportional
- Survivor Selection: Generational replacement

# Example: Credit Scoring

- Distinguish good from bad loan applicants
- Model needed that matches known data

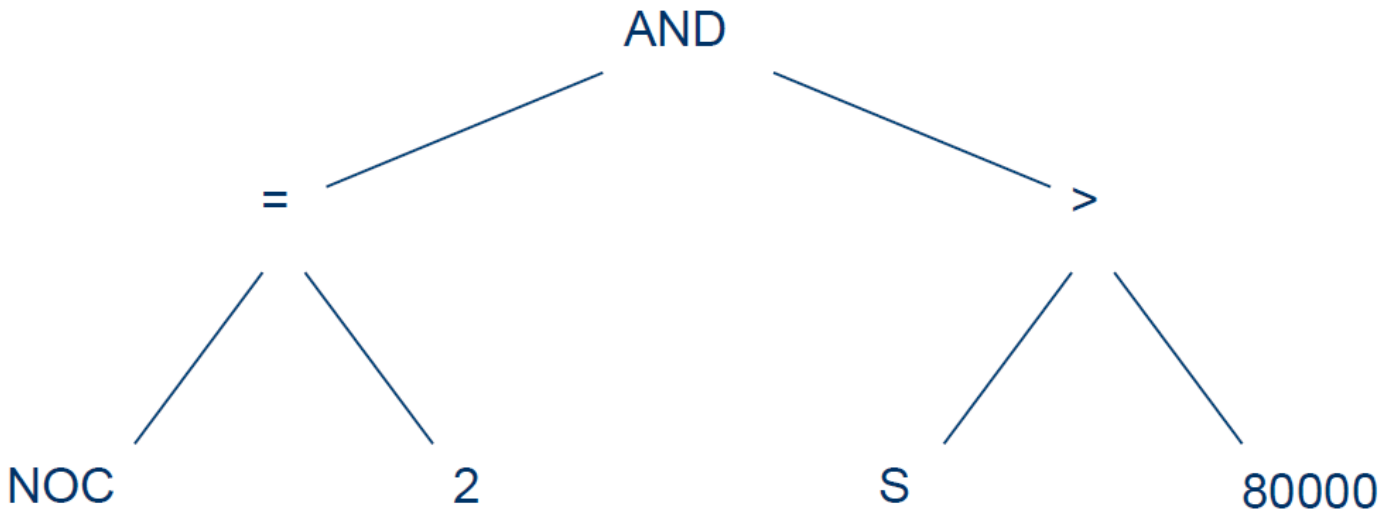
| ID | No of children | Salary | Marital status | OK? |
|---|---|---|---|---|
| ID-1 | 2 | 45000 | Married | 0 |
| ID-2 | 0 | 30000 | Single | 1 |
| ID-3 | 1 | 40000 | Divorced | 1 |
| … | | | | |

# Example: Credit Scoring

- A possible model:
  - IF (NOC=2) AND (S>80000) THEN good ELSE bad
- In general
  - IF formula THEN good ELSE bad
- Only unknown is the right formula, hence
- Our search space (phenotypes) is the set of formulas
- Fitness: percentage of well classified cases of the model it stands for
- Representation of formulas: Parse Trees

# Parse Tree Example

- IF (NOC=2) AND (S>80000) THEN good ELSE bad



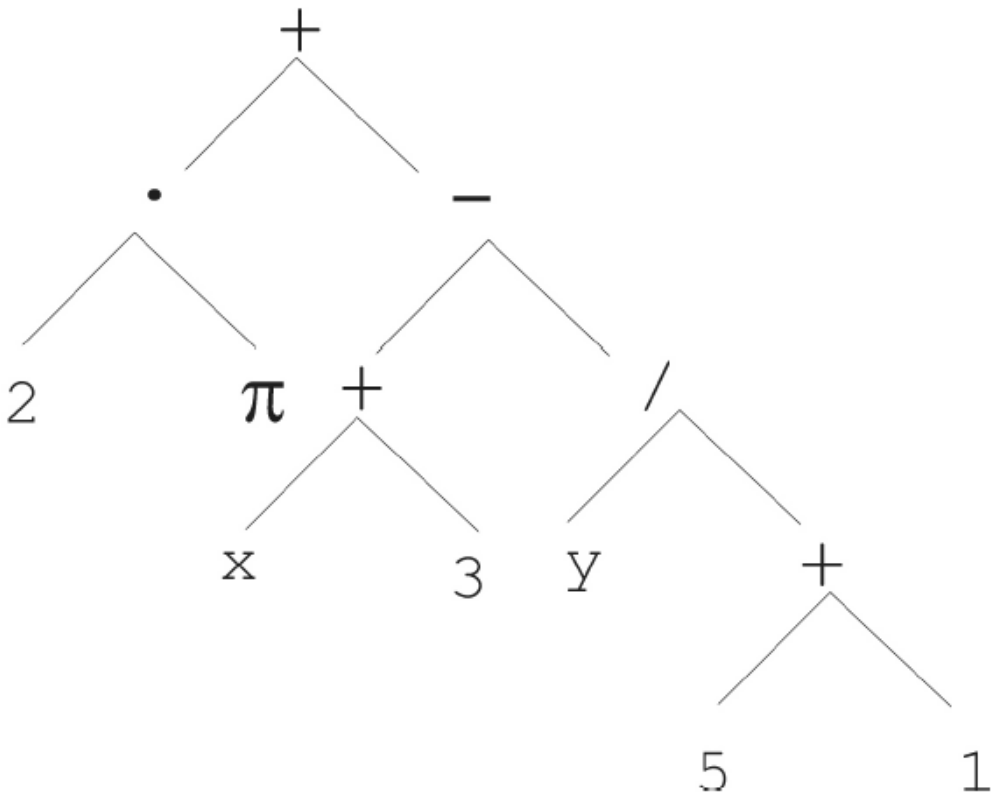
# Tree-Based GP Representation

- Trees are a universal form

$$2 \cdot \pi + \left( (x + 3) - \frac{y}{5 + 1} \right)$$

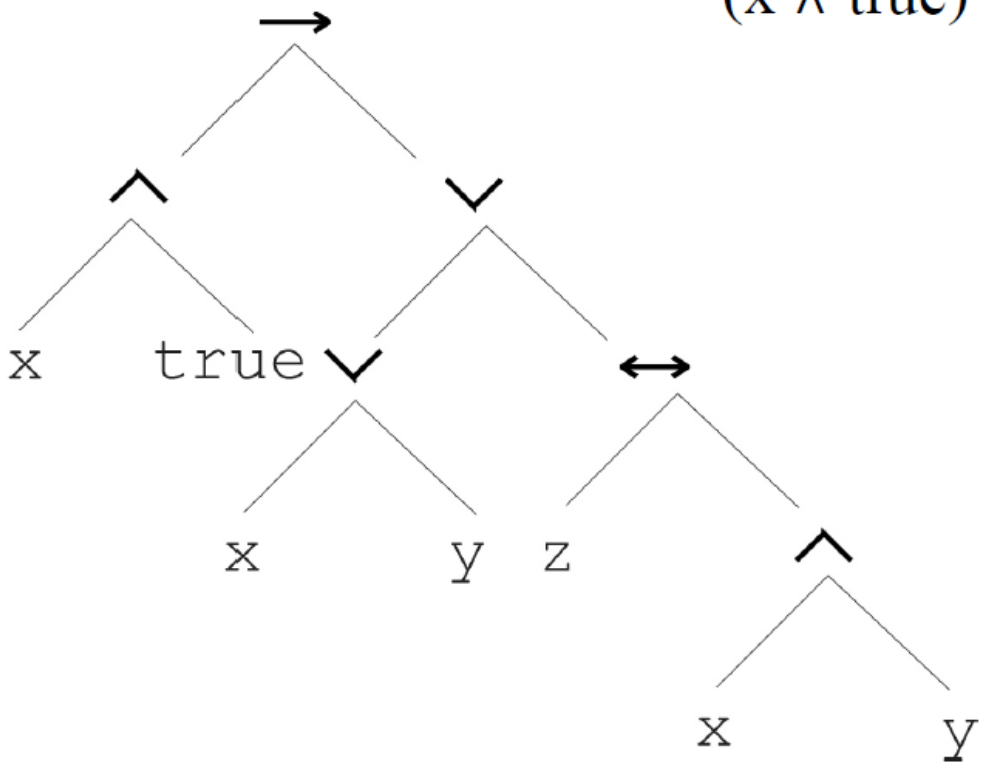
```
i =1;
while (i < 20)
{
        i = i +1
}
```

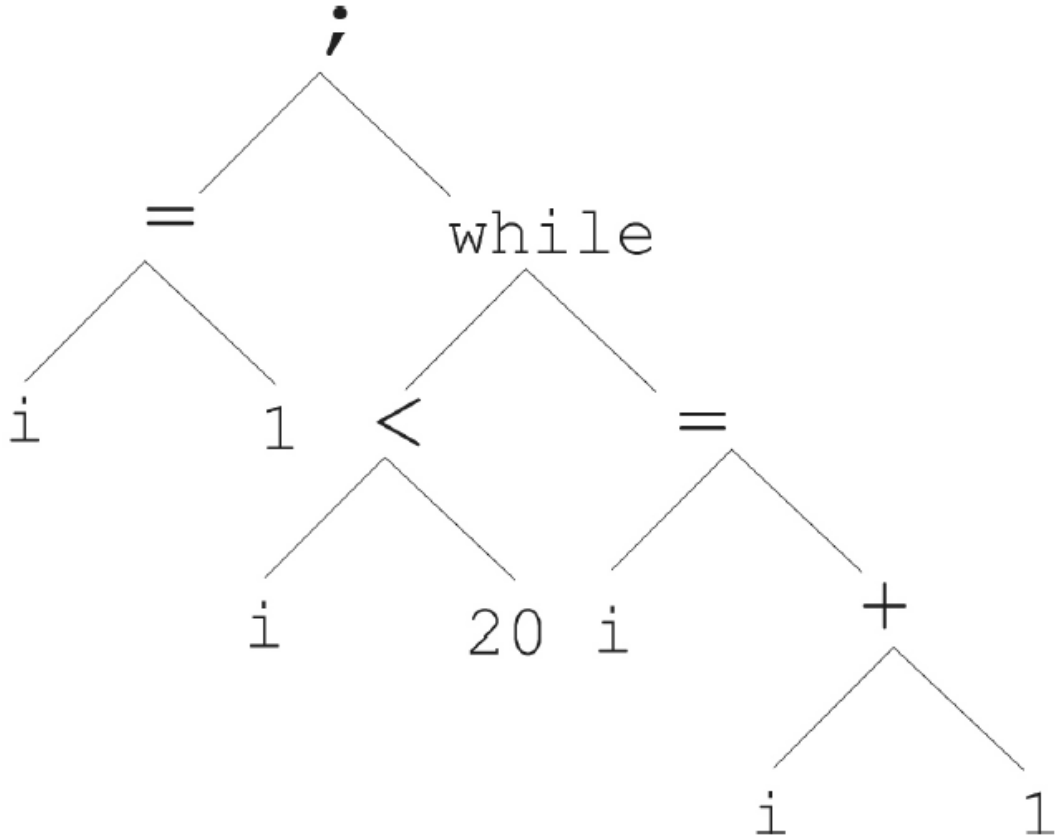
$$(x \wedge \text{true}) \rightarrow ((x \vee y) \vee (z \leftrightarrow (x \wedge y)))$$




$$2 \cdot \pi + \left( (x + 3) - \frac{y}{5 + 1} \right)$$

$$(x \wedge \text{true}) \rightarrow ((x \vee y) \vee (z \leftrightarrow (x \wedge y)))$$






```
i =1;
while (i < 20)
{
        i = i +1
}
```

# GP Representation

- In GA, chromosomes are linear structures (vectors, integer strings, permutations)
- Tree shaped chromosomes are non-linear
- In GA, size of chromosomes is fixed
- Tress in GP may vary in depth and width

# GP Terminals and Functions

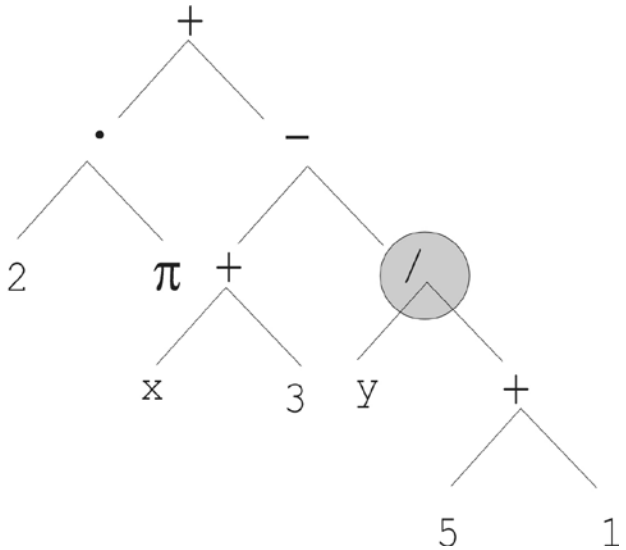
- Symbolic expressions defined by
  - Terminal set T
  - Function set F
- Terminals provide a value to the system
  - Comprised of the inputs to the GP program, the constants, and the zero-arg functions
- Functions process a value already in system
  - Comprised of the statements, operators, and functions available to the BP system (Boolean, arith, conditional, control, loop, etc)

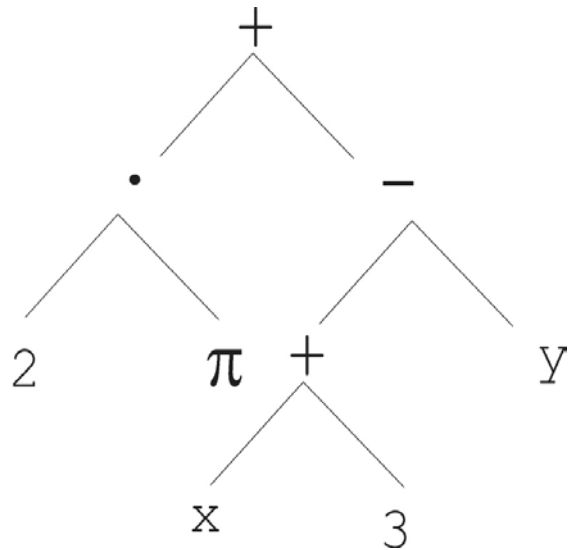
# Initialization of GP Trees

- Maximal initial depth of trees $D_{max}$ set
- Full method (each branch has depth = $D_{max}$)
  - Nodes at $d<D_{max}$ randomly chosen from function set F
  - Nodes at $d=D_{max}$ randomly chosen from terminal set T
- Grow method (each branch has depth <= $D_{max}$)
  - Nodes at $d<D_{max}$ randomly chosen from F U T
  - Nodes at $d=D_{max}$ randomly chosen from T
- Common GP init: Half-and Half, where grow and null each deliver half the population

# Mutation Operator

- Common: Replace randomly chosen subtree by randomly generated tree



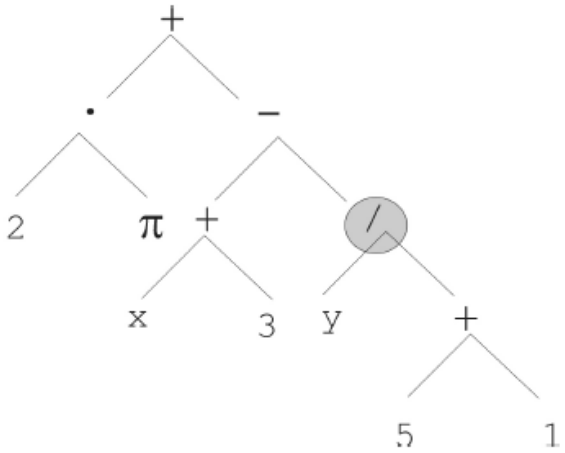
# Mutation Operator

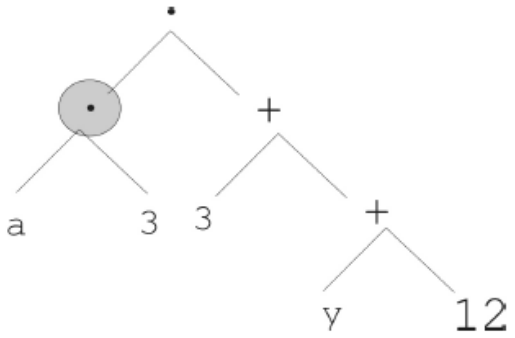
- Mutation has two parameters:
  - Probability $p_m$ to choose mutate vs. recombination
  - Probability to choose a the given subtree to replace
- Remarkably, pm is advised to be 0, or very very small like 0.05 (Banzahf et al ‘98)
- Size of the child can exceed size of the parent

# Recombination

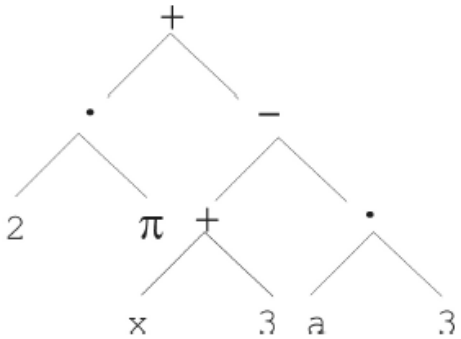
- Common: exchange two randomly chosen subtrees among the parents
- Recombination has two parameters:
  - Probability pc to choose recombination vs. mutation
  - Probability to choose an internal point within each parent as the crossover point
- The size of offspring can exceed that of parents

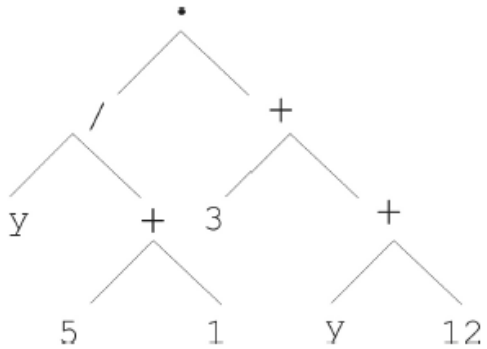
parents




offspring



# Fitness Evaluation

- The measure to choose parents
- Fitness: How well a program has learned to predict the outputs from given inputs
- Designed to give continuous feedback about how well a program performs on the training set
- Error-Based fitness functions, squared errors, or square root errors

# Fitness Examples

- Number of matching pixels in image matching
- Number of walls hit for a robot controlled by GP to learn obstacle avoidance
- Number of correctly classified examples
- Deviation between prediction and reality
- Money won by agent in betting game
- Amount of food found and eaten in life sim

# Generational GP

1. INITIALIZE population w/ random individuals
2. REPEAT UNTIL (termination condition)
3. EVALUTE population / individual fitness
4. SELECT parents with high fitness
5. COMBINE parents to form offspring
6. MUTATE resulting offspring
7. NEXT POP = select from [pop,offspring,parents]

# Steady-State GP

1. Initialize the population
2. while (termination condition not met)
3. Randomly choose subset of pop for a tournament
4. Evaluate fitness of each individual in tournament
5. Select winners of tournament via selection algorithm
6. Apply genetic operators to winners of tournament
7. Replace losers of tournament with offspring
8. Return the best individual from the population

# GP Example: Symbolic Regression

- Target Function: $y = f(x) = \frac{x^2}{2}$
- GP Configutation:
  - Terminal set: Variable x, int constants [-5,5]
  - Function set: +, -, *, % (protected /)
  - Fitness: RMSE over 10 training cases
  - Parameters: pop size, initialization, crossover rate, mutation rate, selection methods

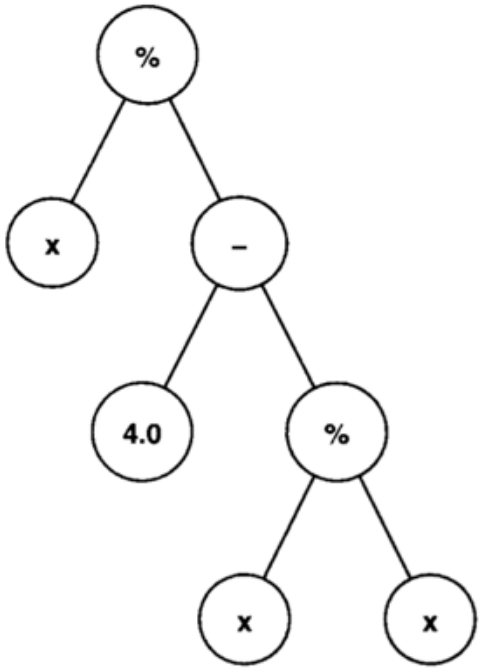
# GP Example: Symbolic Regression

- Training Set

| | Input | Output |
|---|---|---|
| Fitness Case 1 | 0.000 | 0.000 |
| Fitness Case 2 | 0.100 | 0.005 |
| Fitness Case 3 | 0.200 | 0.020 |
| Fitness Case 4 | 0.300 | 0.045 |
| Fitness Case 5 | 0.400 | 0.080 |
| Fitness Case 6 | 0.500 | 0.125 |
| Fitness Case 7 | 0.600 | 0.180 |
| Fitness Case 8 | 0.700 | 0.245 |
| Fitness Case 9 | 0.800 | 0.320 |
| Fitness Case 10 | 0.900 | 0.405 |

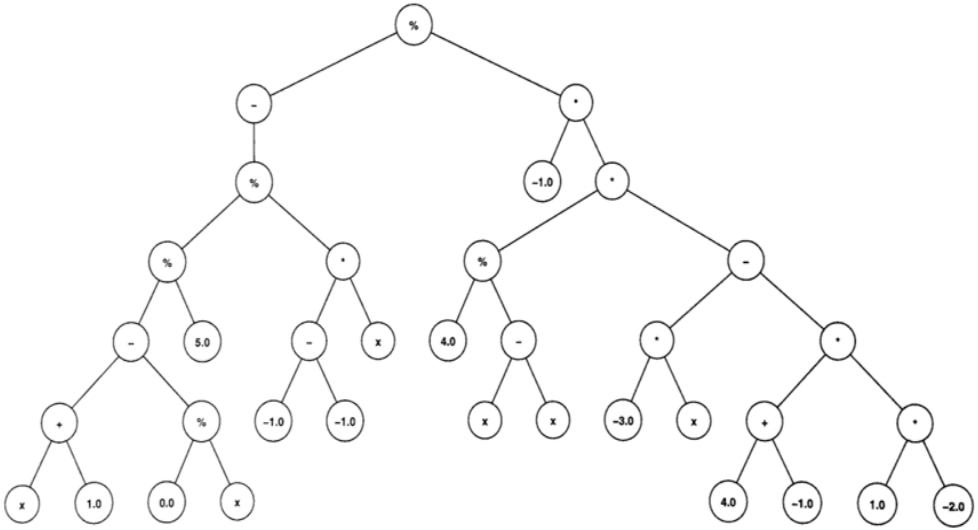
# GP Example: Symbolic Regression

- Best individual from generation 0



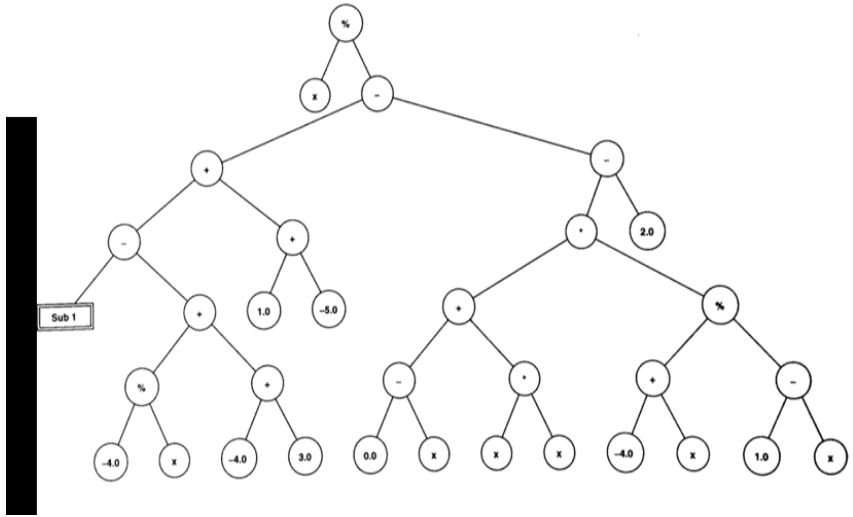
# GP Example: Symbolic Regression

- Best individual from generation 1



# GP Example: Symbolic Regression

- Best individual from generation 2



# GP Example: Symbolic Regression

- Best individual from generation 3

$$y = f(x) = \frac{x^2}{2}$$