Lab 12 - Genetic Programming for Evolving N-Parity Functions

CSE, SUSTech

Outline of This Lab

- Teaching Assistant
- What is Genetic Programming?
- What is N-Parity Problem?
- Experimental Setup
- Illustrate the Results!

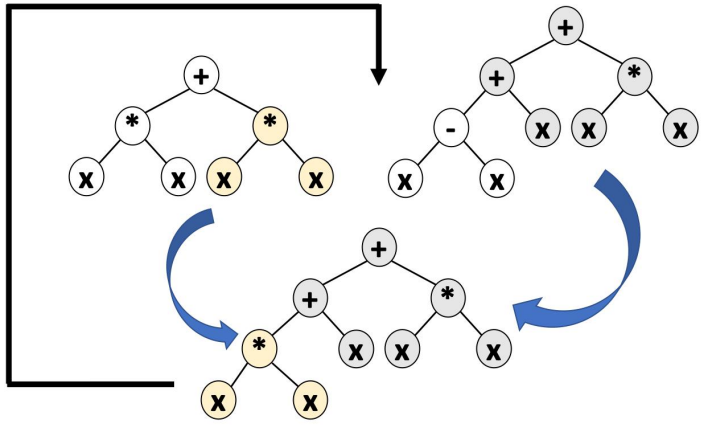
Teaching assistant

- Ms. Honglin Jin (12531321@mail.sustech.edu.cn)

What is Genetic Programming (GP)?
-Introduction

First used by de Garis to indicate the evolution of artificial neural networks, but used by Koza to indicate the application of GP to the evolution of computer programs.

1. Trees (especially Lisp expression trees) are often used to represent individuals.
2. Both crossover and mutation are used.

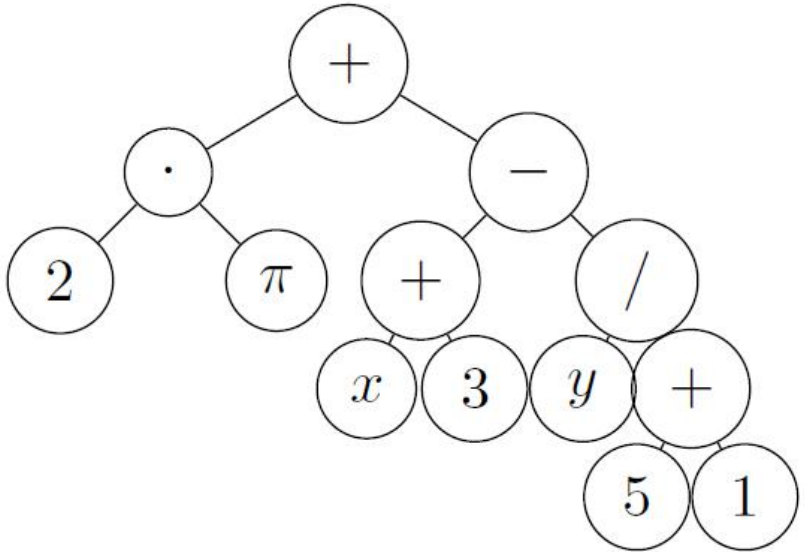


Representation	Tree structures
Recombination	Exchange of subtrees
Mutation	Random change in trees
Parent selection	Fitness proportional
Survivor selection	Generational replacement

Figure 1: Left: illustration of GP. Right: Table 6.4 of [4].

What is Genetic Programming (GP)?
-*Tree Representation: An Example*

- Arithmetic formula: $2 \cdot \pi + ((x + 3) - y/(5+1))$
- Prefix notation
  - ✓Polish notation: +(·(2, π),−(+(x, 3), /(y, +(5, 1))))
  - ✓LISP notation: (+(· 2 π) (−(+ x 3)(/ y (+ 5 1))))
- Parse tree:
  - ✓Nodes (or points) indicate the instructions to execute.
    - ➢Internal nodes: functions.
    - ➢Leaves: terminals.
  - ✓Links indicate the arguments for each instrucption.



What is Genetic Programming (GP)?
-*Preparatory Steps*

Before all, let's see what you need to prepare first . . .

1. The set of primitive functions F. → Define the search space.
   - ✓Example: arithmetic functions and conditional branching operators, or actions of a robot.
2. The set of terminals T.→ Define the search space.
   - ✓Example: independent variables (program's external input), zero-argument functions and numerical constants.
3. The (explicit or implicit) fitness function.→ Define the goal.
4. The algorithm control parameters.
   - ✓Example: population size and maximum size for programs.
5. The termination criterion/criteria.
   - ✓Example: maximum number of generations, or problem-specific success predicate.

What is Genetic Programming (GP)?
-Main Steps of GP

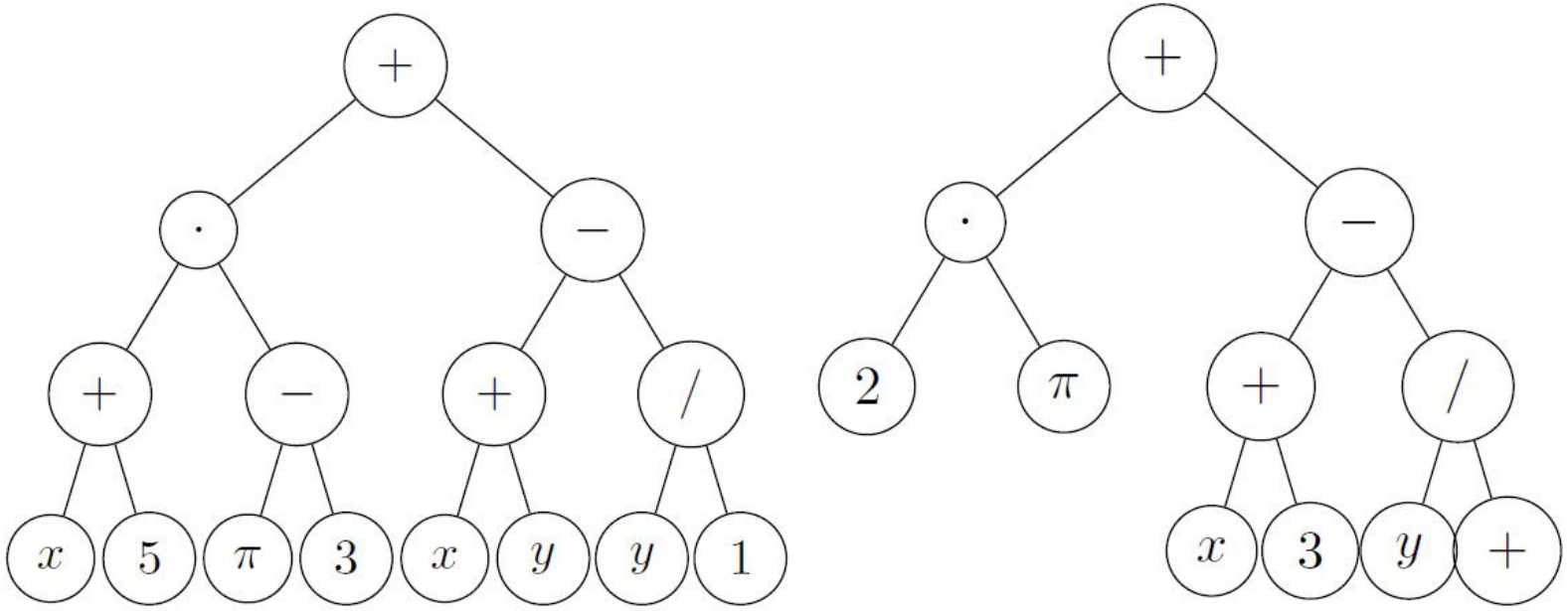
Algorithm 1 Main Steps of GP [7].

1: **Input:** a set of functions F
2: **Input:** a set of terminals T
3: **Input:** a fitness measure
4: **Input:** control parameters
5: Randomly initialise a population of individuals pop
6: $t = 0$
7: **while** termination criterion is not met **do**
8: **for** program $\in pop$ **do** ▶ Evaluation
9: Execute individual program and obtain its fitness
10: **end for**
11: Select one or two individual(s) from pop with a probability based on fitness (with re-selection allowed)
12: Create new individual(s) for the population by applying reproduction, crossover, mutation and architecture-altering operations with specified probabilities
13: **end while**
14: **Return** the best-so-far individual

Now, let's go through initialisation, crossover, mutation and architecture-altering operators one by one.

What is Genetic Programming (GP)?
-*Initialisation Methods I*

- Full method (left): all the branches has depth D_{max}.
- Grow method (right): the branches may have different depths.



What is Genetic Programming (GP)?
-Initialisation Methods II

- Ramped half-and-half (混合法):
  - ✓Half of the population are generated using the full method, while using grow method for generating the others.
  - ✓Sometimes, each individual is generated using either method with equal probability.

Algorithm 2 Ramped half-and-half.

1: **Input:** a set of functions F
2: **Input:** a set of terminals T
3: **Input:** maximum depth D_{max}
4: **for** $i \in \{1, \ldots, \mu\}$ **do**
5: **if** rand < 0.5 **then** ▶ Full method
6: **for** $d \in \{1, \ldots, D_{max} - 1\}$ **do**
7: The contents of nodes at depth d are chosen from F ▶ Select functions
8: **end for**
9: The contents of nodes at depth D_{max} are chosen from T ▶ Select terminals at leaves
10: **else** ▶ Grow method
11: The tree is constructed beginning from the root, with the contents of a node being chosen stochastically from $F \cup T$ if $d < D_{max}$
12: **end if**
13: **end for**

What is Genetic Programming (GP)?
-*Crossover & Mutation*

- Crossover: Create new offspring by recombining randomly chosen parts from two selected parents.
- Mutation: Create one new offspring by randomly mutating a randomly chosen part of one selected parent.

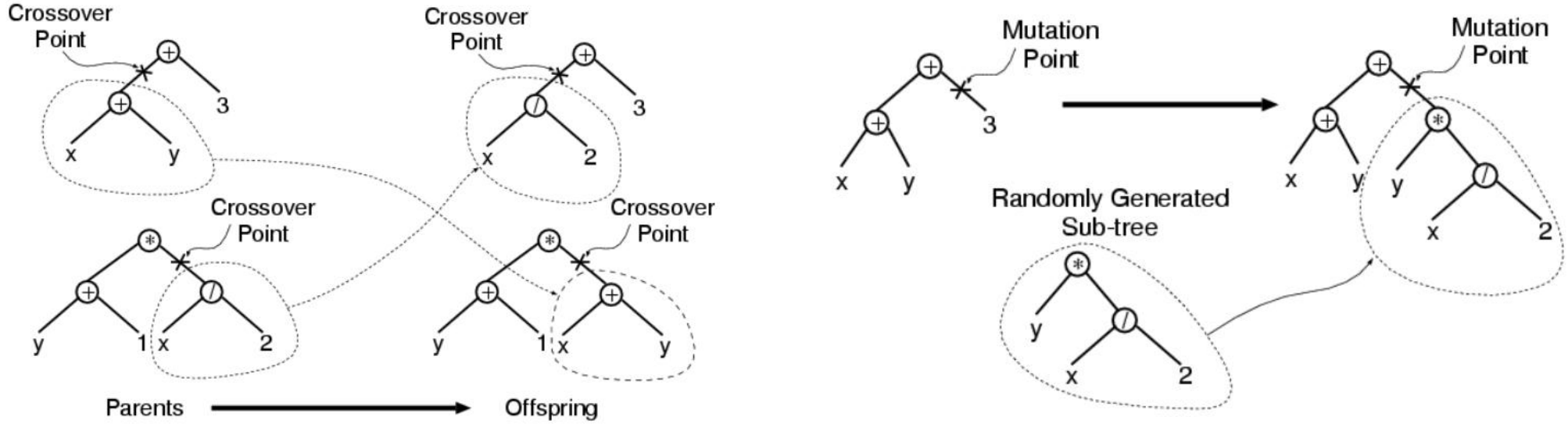


Figure 2: Figures 5.10 and 5.11 of [7].

What is Genetic Programming (GP)?
-Selection I

- Over-selection
  - ✓Often used to deal with the typically large population sizes.
  - ✓Large populations are not unusual in GP.
  - ✓Steps:
    1. Rank the population.
    2. Divide the population into two groups: G_A contains the top α% and G_B contains the (100 − α)%.
    3. Parent selection: 80% come from G_A while the rest come from G_B.
  - ✓How to select α?
    - ➢ α is found empirically.
    - ➢ α depends on the population size.
    - ➢ The selection pressure increases dramatically for larger populations.

      → #individuals from which the majority of parents are chosen stays a low constant value.

What is Genetic Programming (GP)?
-Selection II

- Bloat (膨胀): Average tree sizes tend to grow along the evolution.
  - ✓Also called the code growth or Survival of the fattest.
  - ✓Questions: Why does bloat happen? Is bloat good?
  - ✓Main techniques to control bloat:
    1. Parsimony pressure: Penalty term to reduce the fitness of large trees.
    2. Set a fixed limit on the size or depth of the tree.

       → Reject a tree if it is over-sized (consider as an infeasible solution). → Or set fitness to 0 if over-sized.
    3. Modify search operators.
    4. Multi-objective techniques (fitness and size).

  All sounds like Constraint Handling techniques!

What is N-Parity Problem?

The N-Parity Problem is a classic and difficult binary classification task used to test algorithms, particularly in neural networks and quantum computing. Given an input of N binary bits (0s and 1s), the goal is to determine the parity—whether the total number of ones in the input vector is even or odd.

Experimental Setup

Implement a genetic programming to solve N-Parity problems.

- ✓Initialization: Ramped half-and-half
- ✓Population size: 4000
- ✓Maximum genaration number: 50
- ✓Crossover rate: 0.7
- ✓Mutation rate: 0.3
- ✓Over-selection: 20%

Illustrate The Results!

- Run the implemeted GP and then draw hit histograms for generations 0, 3, 5, 7, 9, and 50 for each N-parity problem with N to be 2, 3, 4, 5, and 6, respectively.
  - ✓x-axis: Hits
  - ✓y-axis: Frequency
- Provide the obtained best individual of the last generation for each N-parity problem with N to be 2, 3, 4, 5, 6, and their corresponding tree structures, respectively.