Genetic Programming 1: 
Introduction

Andrew Kusiak
Intelligent Systems Laboratory
2139 Seamans Center
The University of Iowa
Iowa City, Iowa 52242 – 1527

andrew-kusiak@uiowa.edu
http://www.icaen.uiowa.edu/~ankusiak

This presentation is based on the material provided by Professor
W. Pedrycz, ECE Department, University of Alberta, Canada)

Introduction

• Learning
• Search Strategies
• Genetic Algorithms
• Genetic Programming

Reasoning Process: Typology

- Abduction: Create a model (hypothesis) for a specific data set to explain that data set
- Induction: Extend model (hypothesis) for representative data sets to make a general assertion or explanation
- Deduction: Apply a model to generate a hypothesis for detecting and classifying (explaining) the existence of a target

Reasoning Process: Typology

- Abduction: Data mining (Discovery of models)
- Induction: Data fusion (Detection of models)
- Deduction: Analysis

E. Waltz 1998

Types of Learning

- Supervised: Training examples with known inputs and outputs
- Unsupervised: No outputs are specified
- Reinforcement: Falls between the previous two types; A notion for the output quality is fed back to the learning algorithm.

Search Strategies in Learning Systems

- Blind (uninformed) search, e.g., tree search
  - breadth-first-search strategy
  - depth-first-search strategy
- Hill climbing,
  e.g., simulated annealing
- Beam search: limited number of solutions at each vertex
Genetic Algorithms: Basic Terms

- **POPULATION** – a set of individuals which evolve according to rules of selection and genetic operators
- **FITNESS** – a measure of ‘goodness’ assigned to each individual
- **SELECTION** – a process of choosing high fitness individuals
- **GENETIC OPERATORS** – used to perturb high fitness individuals

Genetic Algorithms: Process

- Coding – population size
- Evaluation
- Selection
- Crossover – probability of crossover
- Mutation – probability of mutation
- Number of generations

Genetic Algorithm Representation

- Binary genome
- Fixed size

Genetic Programming

- Representation
  - No constraints on the representation

Genetic Algorithm: Steps

- Create random initial population
- Evaluate population
- Insert into population
- Select individuals for variation
- Vary
- E.g., Crossovers, mutation, reproduction
Genetic Programming: Steps

Create random initial population
Evaluate population
Vary
Evaluate population
Select
Insert into population

GP vs GA

Create random initial population
Evaluate population
Vary
Evaluate population
Insert into population
Select
Insert into population
Select individuals for variation
Vary

Genetic Programming: Population

Population: a set of individuals composed recursively from two sets:
- Set of $N_{func}$ functions:
  - arithmetic operations (+, -, *, …)
  - mathematical functions (sin, cos, exp, …)
  - Boolean operations (AND, OR, NOT)
  - subroutines or other domain-specific functions
- Set of $N_{term}$ terminals:
  - variable atoms (inputs, sensor information, state variables)
  - constant atoms (numbers)
  - functions without arguments

Comment: Analogy to Machine Learning

- A feature (input) in the training data set becomes part of the terminal set in GP.

Thus, the features of the learning domain become some of the primitives used by GP to build program structures.

Example

Genetic Programming: Population

The function set:

\[ F = \{ \text{AND}, \text{OR}, \text{NOT} \} \]

The terminal set:

\[ T = \{ d0, d1 \} \]

Example

Genetic Programming: Population

The even-2-parity function
- Graph representation

\[ \text{or} \quad \text{AND} \]
\[ \text{NOT} \]
\[ d0 \]
\[ d1 \]

- S-expression; LISP representation

\[ \text{OR} \quad \text{AND} \quad \text{NOT} \quad \text{NOT} \quad \text{NOT} \]
\[ \text{AND} \quad \text{AND} \quad \text{d0} \quad \text{d0} \quad \text{d1} \quad \text{d1} \]

Genetic Programming:
Population Initialization

Methods of generating initial population:
- **Full**: functions are chosen until the node reaches the maximum depth, then terminals are selected. Therefore each branch of the tree has the same depth = max depth.
- **Grow**: nodes are randomly selected from the function and terminal sets.
- **Ramped half-and-half**: supports greater population diversity. The population is divided equally among individual trees of depth 2, 3, 4, 5, and 6 (max depth). Half of the trees in each group are initialized with full and half with grow method.

Generating Initial Population

Ramped half-and-half method: enhances population diversity

Assume the maximum depth parameter = 6

- The population is divided equally among individuals to be initialized with trees having depths 2, 3, 4, 5, and 6.
- For each depth group half of the trees are initialized with full method and half with grow method.

Generating Initial Population

Ramped half-and-half method: enhances population diversity

Assume the maximum depth parameter = 6

- The population is divided equally among individuals to be initialized with trees having depths 2, 3, 4, 5, and 6.
- For each depth group half of the trees are initialized with full method and half with grow method.

Fitness Function

\[ f_p = \sum_{\text{All}} |p_i - o_i| \]

- \( p_i \) = (predicted) output from the GP program
- \( o_i \) = (actual) output from the training set

Genetic Programming:
Fitness

Types of fitness functions:
- raw fitness: not transformed
- standardized fitness: zero fitness value is always assigned to the fittest individual
- normalized fitness: all values are between 0 and 1

Genetic Programming:
Selection

Methods of selection:
- fitness-proportional selection
- truncation selection
- ranking selection
- tournament selection
Fitness Proportional Selection

Probability of the individual $i$ to be given a chance passing offspring to the next generation $p_i$

$$p_i = \frac{f_i}{\sum f_i}$$

$f_i$ = fitness of individual $i$

Truncation Selection

Known also as $(\mu, \lambda)$ selection

$\mu$ parents are allowed to breed $\lambda$ offspring, out of which the fittest $\mu$ are used as parents for the next generation

$(\mu + \lambda)$ selection is also used, where offspring and parents participate in the selection

Comment

- For the selection scheme with $\mu < \lambda$ (# offspring $\lambda$) # parents $\mu$ the genetic algorithm becomes evolutionary strategy algorithm (another type of evolutionary computation algorithm)
- The selection in EA is over-production selection, not mating selection as in GA. It is closer to what Darwin called “natural selection”

Ranking Selection

- Selection probability is assigned to an individual as a function of its rank
- Linear and exponential ranking functions are most often used

Tournament Selection

- Selection based on a competition within a subset of the population
- Tournament = the number of individuals is selected randomly

Genetic Programming: Genetic Operators

CROSSOVER

Two individuals

$$\text{(OR NOT d1) (AND d1 d1)}$$

$$\text{(OR NOT d1) (AND NOT d1 NOT d1)}$$

$$\text{(OR OR NOT d1) (AND NOT d1 NOT d1)}$$
Genetic Programming:
Genetic Operators

**CROSSOVER**

Two individuals

- **OR**
- **AND**
- **NOT**
- **d0**
- **d1**

Comment: Generalized Crossovers

- "Intelligent" crossover: Selection of a crossover point that is less destructive to the offspring
- Crossover operator that learns
- Heuristic guided crossover
- Context sensitive crossover

Genetic Programming:
Genetic Operators

**MUTATION:** within one individual

- **OR**
- **AND**
- **NOT**
- **d0**
- **d1**

Genetic Programming:
Parameters

- Major parameters
  - population size
  - number of generations
- Minor parameters
  - probability of crossover
  - selection of crossover points
  - size of S-expressions
  - probability of mutation
- Different ways of executing the runs
  - initial population
  - selection method
  - elitist strategy

Genetic Programming: Fuzzy Rule-Based System Design

- **Coding**
  - each individual represents a single model, an array of floating-point numbers

- **Parameters**
  - # rules
  - # variables
  - Parameters of linguistic labels
  - Parameters of output functions
  - THEN part