Genetic Programming 1: Introduction

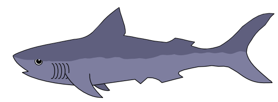
Andrew Kusiak
Intelligent Systems Laboratory
2139 Seamans Center
The University of Iowa
Iowa City, Iowa 52242 – 1527

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

Some parts of 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



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 Programming Overview

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

The University of Iowa Intelligent Systems Laboratory

Genetic Algorithm: 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

The University of Iowa Intelligent Systems Laboratory

Genetic Operators

- Mutation

Before 1 1 1 1 1 1 1

After 1 1 1 0 1 1 1

Mutated gene

Mutation usually happens with probability p_m for each gene

- Crossover

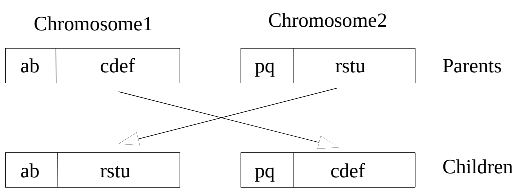
The University of Iowa Intelligent Systems Laboratory

Crossover Types

- Single-point crossover
- Double-point crossover
- Uniform crossover
- Weighted (arithmetic) crossover
- Analytical crossover

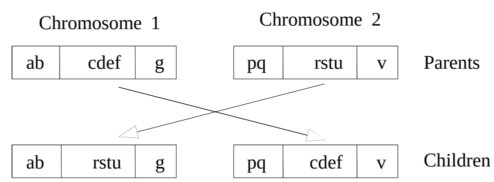
The University of Iowa Intelligent Systems Laboratory

Single-Point Crossover



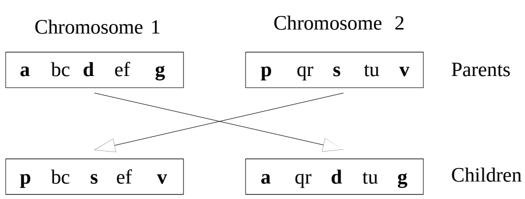
The University of Iowa   Intelligent Systems Laboratory

Double-Point Crossover



The University of Iowa   Intelligent Systems Laboratory

Uniform Crossover



The uniform crossover works on individual locus rather than segments of a chromosome. The probability of selecting a locus for exchange is called the mixing rate.
A mixing rate of 0.5 implies that each locus in the chromosome has an equal chance of being selected for replacement.

The University of Iowa   Intelligent Systems Laboratory

Weighted (Arithmetic) Crossover (a)

- *Weighted (arithmetic) crossover* modifies rather than exchanges genetic material. It works at the chromosome level rather than individual loci. Weight *w* is selected before each crossover and then loci are randomly selected and exchanged.

- The expressions (1) and (2) represent the crossover process.

The University of Iowa   Intelligent Systems Laboratory

Weighted (Arithmetic) Crossover (b)

$c_1 = wp_1 + (1 – w)p_2$ (1)

$c_2 = wp_2 + (1 – w)p_1$ (2)

where:

w = weight in [0, 1]

c_1 = child 1

c_2 = child 2

p_1 = parent 1

p_2 = parent 2

The University of Iowa   Intelligent Systems Laboratory

Analytical Crossover

Analytical crossover works at a chromosome level. It considers the best and the worst fitness of the two selected parents (see (3) - (4))

$c_1 = p_b + s(p_b – p_w)$ (3)

$c_2 = p_b$ (4)

where:

s = scaling factor in [0, 1]

c_1 = child 1

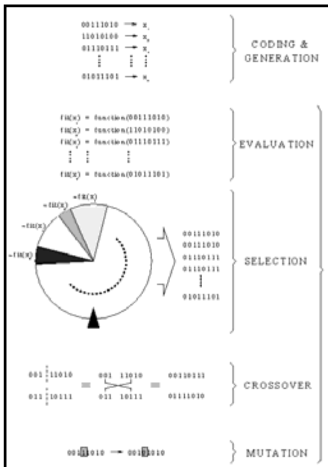
c_2 = child 2

p_b = parent with the best fitness

p_w = parent with the worst fitness

Combining crossovers of different types is allowed

The University of Iowa   Intelligent Systems Laboratory



Genetic Algorithm: Process

Fitness function

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

Intelligent Systems Laboratory

Genetic Algorithm

Pseudo code

```
t = 0
Initialize P(t) to random individuals from the set {1,0}
WHILE termination condition is false
    Select individuals for re-production based on fitness
        Those not selected die
    Apply genetic operators to produce offspring
    Produce P(t+1) by adding offspring population to parent
    population
END
```

The University of Iowa   Intelligent Systems Laboratory

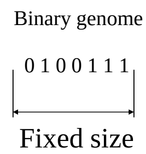
Genetic Algorithm

Drawbacks

- Some solutions are difficult to represent as binary strings.
- Computation time can be excessive.
- Computation not accurately reflect evolution.
- Relatively new concept.

Genetic Algorithm

Representation

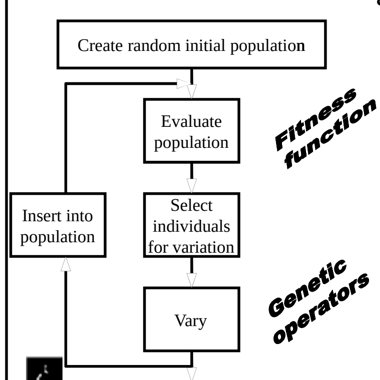


Genetic Programming

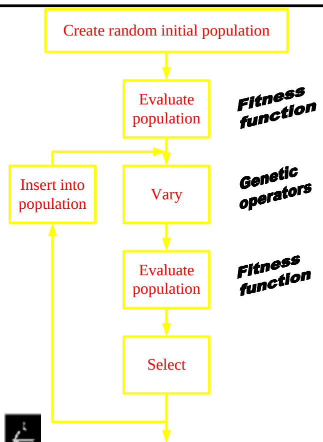
Representation:

No constraints on the representation

Genetic Algorithm: Steps



E.g., Crossover, mutation



Genetic Programming: Steps

The University of Iowa Intelligent Systems Laboratory

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)

The University of Iowa Intelligent Systems Laboratory

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 the primitives used by GP to build a program structures

The University of Iowa Intelligent Systems Laboratory

Example

Genetic Programming: Population

The function set:

$$F = \{AND, OR, NOT\}$$

The terminal set:

$$T = \{d0, d1\}$$

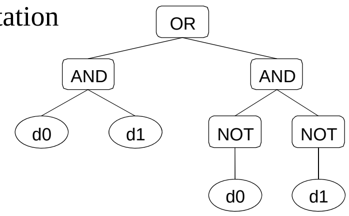
The University of Iowa Intelligent Systems Laboratory

Example

Genetic Programming: Population

The even-2-parity function

- Graph representation



- S-expression; LISP representation

(OR (AND d0 d1) (AND (NOT d0) (NOT d1)))

Genetic Programming: Population

Graph example

Methods of generating initial population:

– **Full**: functions are chosen until the node reaches the maximum depth, 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**

Generating Initial Population

Graph example

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

Genetic Programming: Fitness Function

- Fitness function is a metric
- Fitness function is problem specific
- Fitness function provides feedback to the algorithm which individuals should reproduce
- Fitness function measures how well a program has learned to predict outputs from inputs

Fitness Function

Example

$$f_p = \sum_{\text{All } i} |p_i - o_i|$$

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

The University of Iowa Intelligent Systems Laboratory

Genetic Programming: Selection

Methods of selection:

- fitness-proportional selection
- truncation selection
- ranking selection
- tournament selection

The University of Iowa Intelligent Systems Laboratory

Fitness Proportional Selection

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

$$p_i = f_i / \sum_{\text{All } i} f_i$$

f_i = fitness of individual i

The University of Iowa Intelligent Systems Laboratory

Truncation Selection

Known also as (μ, λ) selection

- μ parents are allowed to breed λ offspring, out of which the fittest μ are used as parents for the next generation
- $(\mu + \lambda)$ selection is also used, where offspring and parents participate in the selection

Ranking Selection

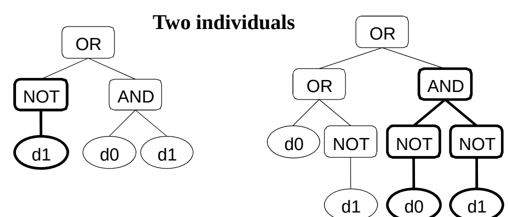
- Selection probability is assigned to an individual as a function of its rank
- Linear and exponential ranking functions are most often used

Tournament Selection

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

Genetic Programming: Genetic Operators

CROSSOVER



(OR (NOT d1) (AND d0 d1))

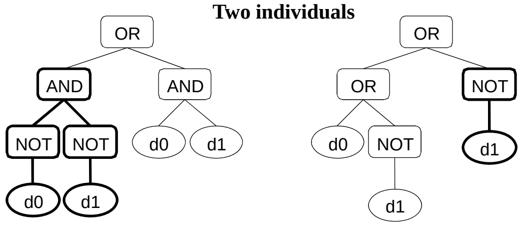
(OR (OR d1 (NOT d0)) (AND (NOT d0) (NOT d1)))

Slide 1

Genetic Programming: Genetic Operators

CROSSOVER

Two individuals



(OR (AND (NOT d0) (NOT d1)) (AND d0 d1))

(OR (OR d1 (NOT d0)) (NOT d1))

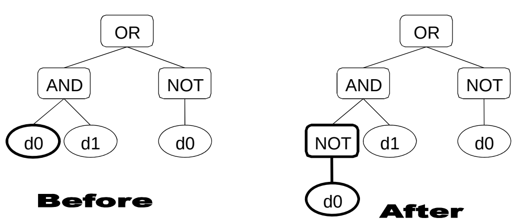
Comment: Generalized Crossovers

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

Slide 2

Genetic Programming: Genetic Operators

MUTATION: within one individual



(OR (AND d0 d1) (NOT d0))

(OR (AND(NOT d0) d1) (NOT d0))

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

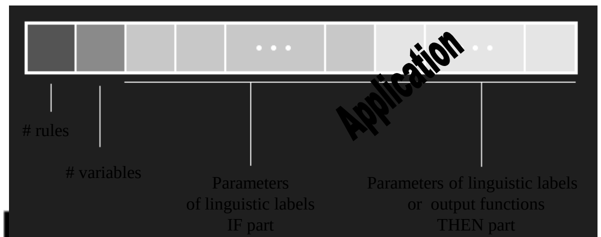
Genetic Programming

- Representation of the problem
  - coding of individuals in the population
- Fitness function
  - evaluation of individuals in their capability to solve the problem

The University of Iowa Intelligent Systems Laboratory

Genetic Programming: Fuzzy Rule-Based System Design

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



The University of Iowa Intelligent Systems Laboratory

References

- Cox, E. (2005), *Fuzzy Modeling and Genetic Algorithms for Data Mining and Exploration*, Morgan Kaufmann, New York, NY.
- Banzhaf, W., P. Nordin, R.E. Keller, and Frank D. Framcone (1998), *Genetic Programming*, Morgan Kaufmann, San Francisco, CA.
- Koza, J. R. *et al.* (2003), *Genetic Programming IV*, Kluwer, Norwell, MA.

The University of Iowa Intelligent Systems Laboratory