Outline

- Variable Length Representation, mostly used to solve modeling type of problems.
  - Parse Tree
  - Graph
  - Finite State Machine
- Unlike fixed-length representation, both contents and structure of a variable length representation are evolved.

Data Modeling

- In many scientific and engineering fields, one important task is to find significant and interesting patterns (model) from data that have been collected via experiment or computer simulation.
- The model can then be used to forecast the future for the same kind of data.

Data Modeling - Continued

- This modeling task can be formulated as a search problem.
  - search for a model explaining the data;
  - search for a rule predicting the outcome of new data.
- These models can be represented in many different data structures:
  - Parse tree; Graph; Finite state machines etc
Parse Tree

- Most of the initial Genetic Programming work evolves parse tree models.
- A parse tree represents a LISP program (symbolic expression) that takes inputs and generates outputs.
- The internal nodes of a parse tree contain functions while the leaf nodes contain terminals.
- Both structure and contents of a parse tree evolve.

Closure Requirement

- Each function in the function set $F$ should be able to take any terminal in the terminal set $T$ as an input.

Strongly Typed GP

- Aims: relax the closure requirement of the original GP and enrich the representation.[Montana, 1995]
- Method:
  - Specify the types of terminals and the types of function arguments and return values.
  - Type check all parse trees during parse tree creation and evolution using genetic operation.

Regression Analysis

- It models the relationship between a dependent variable $Y$ and one or more independent variables $X$.
  - A regression model $f$ relates $Y$ to $X$: $Y = f(X)$
- To carry out regression analysis, the form of the function $f$ must be specified. Sometimes the form of this function is based on knowledge about the relationship between $Y$ and $X$ that does not rely on the data. If no such knowledge is available, a convenient form for $f$ is used.
Example

• Given some points in \( \mathbb{R}^2, (x_1, y_1), \ldots, (x_n, y_n) \)
• Find a function \( f(x) \) s.t. \( \forall i = 1, \ldots, n : f(x_i) = y_i \)

Regression using GP

• Each parse tree represents a function \( f \)
• There is no need to specify the form of the regression model \( f \).
• However, a modeler (you) has to provide:
  – a function set: e.g. +, -, \*, /, sin, cos, exp, sqrt
  – a terminal set: e.g. \( x_1 \ldots x_n \), plus constants
• The function and terminal sets have to be sufficient (Koza, 1992) to express a model solution.
• Similar to all regression methods, the objective is minimizing sum squared errors (or similar measures):
  \[ \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \]

Cart Centering Problem

• Objective: find a time-optimal bang-bang control strategy (push force in a positive or negative direction) to center a cart on a one-dimensional frictionless track. (from Koza 1992, page 138)

Cart Centering Problem - Continued

• When the mass of the cart is 2.0 kilograms and the force is 1 newton, the known time-optimal strategy is:
  
  \[
  \begin{cases}
    \text{push positive direction} & \text{if } (-x > v \cdot \text{sign}(v)) \\
    \text{push negative direction} & \text{else}
  \end{cases}
  \]

  – \( x \) and \( v \) are position and velocity of the cart
  – \( \text{sign}(v) \) returns +1 if \( v \) is positive, -1 if \( v \) is negative
GP Modeling

- **Variables:**
  - independent variables: position \((x)\) and velocity \((v)\) of the cart.
- **Function set:** \{\(\ast, -, \ast, \div, \text{abs}, \text{gt}\}\)
  - \(\text{gt}(v1,v2): \text{if } (v1>v2) \text{ return } 1 \text{ else return } -1;\)
- **Terminal set:** \{\(x, v, \text{rnd}\}\)
- **Search space:** how many model candidates?

GP Modeling - Continued

- **Data samples:**
  - 20 randomly generated \((x,v)\) pairs;
- **Fitness function:** minimize
  \[
  \sum_{i=1}^{20} T_i
  \]
  - \(T\) is the time (second) the model takes to center the cart \((x=0)\)
  - Maximum time allowed is 20 seconds.

A GP evolved solution by Koza:

- It behaves exactly the same as
  \[
  GT(-x, v^2 \text{sign}(v))
  \]

Finite State Machines

- A FSM is specified by
  - Inputs \(I\)
  - States \(S\)
  - Outputs \(O\)
  - Transition function \(\delta: S \times I \rightarrow S \times O\)
  - Start state
  - End state

- A FSM can be used for
  - language recognition
  - automatic controller
  - robot controller
Example I

• Task: evolve a FSM that predicts the next bit of an input streams of bit 0 or 1.
• Inputs I: \{0, 1\}
• Outputs O: \{0, 1\}
• States: \{A, B\}
• Fixed-length model

Example II

• A thermostat controller:
• Inputs I: \{hot, okay, cold\}
• Outputs O: \{air-conditioner, do-nothing, furnace\}
• States: \{ready, heating, cooling, just-heated, just-cooled\}
• Fixed length model

FSM Controller

Classroom Exercises I

• Design the function and terminal sets for GP to evolve a search algorithm:
  – Linear Search
  – Binary Search
  – Knuth-Morris-Pratt Search
  – Grover’s Search
Linear Search Algorithm

LinearSearch(value, list)
    if the list is empty, return Λ;
    else
        if the first item of the list has the desired value, return its location;
        else return LinearSearch(value, remainder of the list)

• F = ?
• T = ?
• Fitness function = ?
• Data samples = ?

Classroom Exercises II

• Design the inputs I, states S, outputs O, to evolve a FSM for traffic-light control at a street intersection.