Automatic Programming of Agents by Genetic Programming

Lee Spector
Cognitive Science
Hampshire College
Amherst, MA 01002

lspector@hampshire.edu
http://hampshire.edu/lspector
Overview

Genetic Programming

Evolved Agents in Dynamic Environments

New Techniques: PushGP, Pushpop

New Worlds: Traffic, Airlift
Genetic Programming
(Koza, 1992)

Create initial random population of programs

Evaluate fitness of each individual

Termination criterion satisfied?

Y

Designate result

N

Apply genetic operators to produce new population
Mutation of Symbolic Expressions

(+ (* X Y)
  (+ 4 (- Z 23)))

(+ (* X Y)
  (+ 4 (- Z 23)))

(+ (- (+ 2 2) Z)
  (+ 4 (- Z 23)))
Crossover of Symbolic Expressions

Parent 1: \((+ (* X Y) (+ 4 (- Z 23)))\)

Parent 2: \((- (* 17 (+ 2 X)) (* (- (* 2 Z) 1) (+ 14 (/ Y X))))\)

Child 1: \((+ (- (* 2 Z) 1) (+ 4 (- Z 23)))\)

Child 2: \((- (* 17 (+ 2 X)) (* (* X Y) (+ 14 (/ Y X))))\)
Symbolic Regression

Goal: given a data set of (x, y) pairs, produce a program that takes an x value as input and produces the appropriate y value as output.

Function set: {+, -, *, %}

Terminal set: {X, 0.1}

Fitness function: sum the error for X values 0.0, 0.2, ..., 0.9
Target Function: \( y = x^3 - 0.2 \)
Other GP parameters

Maximum number of Generations: ........................................51
Size of Population: ................................................................1000
Maximum depth of new individuals: ..................6
Maximum depth of new subtrees for mutants: .......4
Maximum depth of individuals after crossover: ......17
Fitness-proportionate reproduction fraction: ........0.1
Crossover at any point fraction: .................................0.3
Crossover at function points fraction: ....................0.5
Selection method: FITNESS-PROPORTIONATE
Generation method: RAMPED-HALF-AND-HALF
Randomizer seed: .............................................................1.2
Best Program, Generation 0

\[- (\% (\times 0.1 \\
  (\times X X)) \\\n (- (\% 0.1 0.1) \\
  (\times X X)))) \\
0.1)\]
Best Program, Generation 3

\((- (* X X) (+ 0.1 0.1))\)
Best Program, Generation 5

\[-(\star (\star (\% X 0.1) (\star 0.1 X))(- X (\% 0.1 X))) 0.1)\]
Best Program, Generation 12

\[
\begin{align*}
(+ & (- (- 0.1 \\
& (- 0.1 \\
& (- (* X X) \\
& (+ 0.1 \\
& (- 0.1 \\
& (* 0.1 \\
& 0.1)))))))) \\
(* & X \\
(* & (% 0.1 \\
(% (* (* (- 0.1 0.1) \\
(+ X \\
(- 0.1 0.1))))) \\
X) \\
(+ X (+ (- X 0.1) \\
(* X X)))))) \\
(+ 0.1 (+ 0.1 X)))))) \\
(* & X X))
\end{align*}
\]
Best Program, Generation 22

\[-(-(* \, X \, (* \, X \, X)) \, 0.1) \, 0.1)\]
# Wumpus World

<table>
<thead>
<tr>
<th></th>
<th>Breeze</th>
<th>Breeze</th>
<th>Breeze</th>
<th>Breeze</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>O</strong></td>
<td>Pit</td>
<td>Breeze</td>
<td>Breeze</td>
<td><strong>O</strong></td>
</tr>
<tr>
<td><strong>O</strong></td>
<td>Pit</td>
<td>Breeze</td>
<td>Breeze</td>
<td><strong>O</strong></td>
</tr>
<tr>
<td>Breeze</td>
<td>Breeze</td>
<td>Breeze</td>
<td>Breeze</td>
<td></td>
</tr>
<tr>
<td>Breeze</td>
<td><strong>O</strong></td>
<td>Breeze</td>
<td>Breeze</td>
<td><strong>$</strong></td>
</tr>
<tr>
<td>Breeze</td>
<td>Stench</td>
<td>Stench</td>
<td>Stench</td>
<td>Breeze</td>
</tr>
<tr>
<td><strong>Agent</strong></td>
<td>Stench</td>
<td>Breeze</td>
<td><strong>O</strong></td>
<td>Pit</td>
</tr>
</tbody>
</table>
Wumpus World Problem

Goal: to guide an agent through a complex and dangerous virtual world (Russell and Norvig, 1995).

Function set: and, or, not, sequence, if-zero, if-less-or-equal, +, -, *, sensors, constants, [read, write]
Evolving Agents with Culture

Evolutionary Time
Memory/Culture in Wumpus World

Total runs: 1709 (population size 1000, 21 generations/run)

<table>
<thead>
<tr>
<th>Condition</th>
<th>Computational Effort</th>
</tr>
</thead>
<tbody>
<tr>
<td>No memory</td>
<td>1,710,000</td>
</tr>
<tr>
<td>Memory</td>
<td>2,100,000</td>
</tr>
<tr>
<td>Culture</td>
<td>1,386,000</td>
</tr>
</tbody>
</table>
Evolving Learning Agents

Ontogenetic Programming:
Ontogeny via self-modification.

![Graph showing fitness over generations for different types of agents](image-url)
Evolving Teamwork and Coordination


Team composition: homogeneous, heterogeneous, segregated

Sensing: absolute, deictic

<table>
<thead>
<tr>
<th>Sensing</th>
<th>Restricted Breeding</th>
<th>Free Breeding</th>
<th>Clones</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>Best</td>
<td>Average</td>
</tr>
<tr>
<td>Deictic</td>
<td>1.65</td>
<td>0.13</td>
<td>2.03</td>
</tr>
<tr>
<td>Name–Based</td>
<td>1.33</td>
<td>0.03</td>
<td>1.79</td>
</tr>
<tr>
<td>None</td>
<td>2.20</td>
<td>0.49</td>
<td>2.23</td>
</tr>
</tbody>
</table>
Goal: Scale up GP/agents techniques for human-competitive performance in complex, dynamic environments.

Evolve agents that may use:
- multiple data types
- subroutines (any architecture)
- recursion
- evolved control structures
- evolved evolutionary mechanisms

Push supports all of this using simple, mostly standard GP techniques.
Modularity and Scaling

Table 5: Results of PushGP runs on even-parity problems with the instruction set in Table 3.

<table>
<thead>
<tr>
<th>Arity</th>
<th>Effort</th>
<th>% Random Solutions</th>
<th>% Using DO or DO*</th>
<th>Effort Relative To Koza Without ADFs</th>
<th>Effort Relative To Koza With ADFs</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>80,000</td>
<td>49%</td>
<td>20%</td>
<td>1.2X</td>
<td>1.5X</td>
</tr>
<tr>
<td>4</td>
<td>96,000</td>
<td>23%</td>
<td>62%</td>
<td>0.25X</td>
<td>0.55X</td>
</tr>
<tr>
<td>5</td>
<td>352,000</td>
<td>3%</td>
<td>56%</td>
<td>0.54X</td>
<td>0.76X</td>
</tr>
<tr>
<td>6</td>
<td>160,000</td>
<td>4%</td>
<td>63%</td>
<td>0.002X</td>
<td>0.12X</td>
</tr>
</tbody>
</table>
Autoconstructive Evolution

Individuals make their own children.

The machinery of reproduction and diversification (and thereby the machinery of evolution) evolves.

Radical self-adaptation.
Figure 1: Error of the best program and diversity of successful parents over the course of a Pushpop run on a symbolic regression problem with the target function \( y = 5x^2 + x - 2 \). Diversity of two individuals was calculated as the sum, over all unique expressions in either of the individuals, of the difference between the number of occurrences of the expression in the two individuals. The graphed diversity measure is the sum of the diversities of all pairs of individuals in a randomly selected set of 128 successful parents from each generation. When less than 128 parents were successful (in some generations before reproductive competence, which occurred here at generation 32) the graph repeats the value from the previous generation. This run used a population size of 2048, a tournament size of 32, 16 fitness cases (0–15), and a maximum program size of 64 points.
Traffic

[Image of a traffic grid with roads labeled as Road 0, Road 1, Road 2, and Road 3]
Traffic

So far:

- Hampshire simulator.
- Simple road networks/conditions.
- Simple metrics (average wait).
- Simple evolved agents.

Soon:

- BBN simulator.
- Complex road networks.
- Alternative metrics.
- Agents using EAMs.
Virtual Quidditch
Virtual Quidditch

Richly heterogeneous: player roles, balls themselves are active/intelligent.

Richly 3-dimensional: flying game, full use of the third dimension.

Extensible: rules not uniquely determined by the Rowling books; physics based on magic spells so the sky is the limit!

Beyond human experience: unlike soccer, few intuitions about strategy to bias methods.

Like real-time, only faster: model some aspects of real-time but design for rapid fitness tests.
Airlift

(See MIT/BBN report)
Quantum Computing

\[ \begin{array}{c}
\text{0} \\
\text{1} \\
\text{2}
\end{array} \begin{array}{c}
\text{U}(\theta) \\
\text{H} \\
\text{X}(\theta)
\end{array} \begin{array}{c}
\text{F} \\
\text{H} \\
\text{M} \end{array} \begin{array}{c}
\text{H} \\
\text{M0} \\
\text{M1}
\end{array} \]

**Figure 3.** Hand-tuned version of evolved AND/OR; \( \theta = 0.74909, \ldots \)

**Table 1.** Error probabilities (to 5 digits) for hand-tuned simplified AND/OR algorithm.

<table>
<thead>
<tr>
<th>Orbit</th>
<th>( p_e )</th>
<th>Orbit</th>
<th>( p_e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0</td>
<td>0.00560</td>
<td>0 1 0 1</td>
<td>0.28731</td>
</tr>
<tr>
<td>0 0 0 1</td>
<td>0.28731</td>
<td>1 1 0 1</td>
<td>0.21269</td>
</tr>
<tr>
<td>0 0 1 1</td>
<td>0.21269</td>
<td>1 1 1 1</td>
<td>0.00560</td>
</tr>
</tbody>
</table>
Agenda

Integrate PushGP/Pushpop with MIT/BBN and/or additional agent simulators.

Evolve agents; compare evolved/hand-crafted agent designs and performance.

Integrate MIT/BBN Elementary Adaptive Modules (EAMs) into PushGP/Pushpop.

Assess utility of components made available to evolution including EAMs.
Multi-Type, Self-Adaptive Genetic Programming for Complex Applications

New Ideas

- Richly heterogeneous data can be flexibly integrated in programs produced by stack-based genetic programming.

- Explicit code manipulation allows for automatic emergence of modules and evolved program architecture.

- Self-adaptive construction of evolutionary mechanisms enhances fit to problem environments.

Impact

- Evolved agents for heterogeneous, dynamic environments.

- Broader range of applications for automatic programming technologies.

- Automatic programming with less configuration by users.

Schedule

<table>
<thead>
<tr>
<th>Feb 01</th>
<th>Feb 02</th>
<th>Feb 03</th>
<th>Feb 04</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port new GP systems to Beowulf cluster</td>
<td>Integration with agent environments</td>
<td>Alternative building-blocks</td>
<td></td>
</tr>
<tr>
<td>Benchmarking</td>
<td>Application/analysis</td>
<td>Analysis of evolved agents</td>
<td></td>
</tr>
</tbody>
</table>

Hampshire College: Lee Spector