Operator-Based Distance for GP

Steven Gustafson
University of Nottingham, UK

Leonardo Vanneschi
University of Milano-Bicocca, Italy
Why Distance?

- analyse search space
- fitness distance correlation(s)
- diversity measures
- methods using dissimilarity
- predict convergence
- estimate similarity
Why Operator-Based Distance?

- operators define neighbourhood
- search *only* traverses neighbourhood
- non-operator-based measures inaccuracies
Why *NOT* Operator-Based?

- difficult to design
- expensive to compute
- specific for operator(s)
- accuracy gain not always clear
Aim of this Research

- Assess feasibility of operator-based distance

- Define approximations schemes that are
  - more specific than standard distance measures
  - less complex than "true" operator distance
Assumptions

- GP using syntax trees
- *only* operator is subtree crossover
- Edit distance
  - number of node additions/deletions/changes to make two trees equal
Standard Edit Distance

- complexity between two trees:
  \( O(k) \) (k nodes in trees)

- pair-wise distance in population
  \( O(M^2 k) \) (where M is population size)

- for a metric space:
  \( M(M-1)/2 \) (comparisons)

- \( O(Mk) \) with preprocessing
Further Assumptions

• Subtree crossover replaces a subtree in the parent with a donor’s subtree

• The "true" distance between two trees is the algorithmic distance, according to operators (and other algorithm properties)
Some Notation

- P is population with M trees
- T1 is parent tree
- T2 is the tree to transform T1 into
- T1/T2 is the "difference" between trees
- T1/T2 → (st1, st2)
  where st2 must replace st1 in T1 to make T1 equal to T2
OpBD. Overview

- if T1 is in P, then the distance value \textit{depends} on finding st2 in P

- if st2 is not in P, then the distance value \textit{depends} on creating st2 with other operations

- distances greater than 1 require \textit{simulating} possible future operations and populations
OpBD Problems

- distance based on simulation of future populations and operations is not exact
- accuracy is lost with these simulations
- can we find a balance?
Probabilistic OpBD

• provide confidence bound with distance?
  • complicated

• only consider distances of 0 and 1?
  • reflective of generational and steady-state
  • treat "distance" as a probability!
  • incorporate other algorithm and representation properties to increase accuracy
Subtree Crossover Distance

distance \((T_1,T_2,V,P)\)

\[\text{end}\\]

begin
\[(st_1, st_2) = \frac{T_1}{T_2};\]
\[p_{s1} = \text{probSelecting} \ (st_1, T_1);\]
\[p_{s2} = \text{probCreating} \ (st_2, P);\]
\[\text{return} \ (p_{s1} \ast p_{s2});\]
\[\text{end}\]
SXO-OpBD Complexity

- T1/T2 is linear time in size of T1 and T2
- \textit{probSelecting} is linear time of size of T1
- \textit{probCreating} is $O(Mk)$
- \textit{pair-wise distance} for P is in $O(M^3 k^3)$
- \textit{preprocessing} P can reduce complexity
Complexity Reduction

• incorporate algorithm features

• reduce complexity but maintain accuracy

• only consider subtrees in solutions that are likely to be selected (highly fit)

• linear time in M, once for pair-wise distances of population
More Complexity Reduction

- only consider subtrees *likely* to be selected by subtree crossover (according to size)

- consider *fit solutions* and *likely* subtrees

- tune these two approximations for "*appropriate*" levels
The GP System

- steady-state
- $M = 20$
- primitives are empty
- two-node "join" function
- tournament selection, size 3
- new solution replaces worst in $P$
- 500 generations (operator applications)
The Problem

- based on Tree-String - *only structure*

- generate a tree shape to make an instance

- fitness is the absolute *difference* between number of nodes at each depth between instance tree shape and candidate solution

- 30 random instances,
  30 GP runs on each instance
Fig. 1. An instance is defined by a randomly generated tree shape (a) and can be solved perfectly by a candidate tree shape that has the same number of nodes at each depth (b).
Distance and Complexity

\[ D_{SC}(T_1, T_2, P) = \frac{\text{occurrences}(s_{T_2}, P)}{\text{#subtrees}(P)} \]

occurrences of missing subtree st2 in P over the number of subtrees in the population

roughly, the likelihood of selecting st2
Pair-wise Variations

Fig. 2. A scatter plot showing the correlation between the edit distance and the operator-based distance, average pair-wise distance in each population.
Complexity Variations

Fig. 3. A scatter plot showing the correlation between the edit distance and the operator-based distance complexities, or the average number of nodes visiting during the calculation of each pair-wise distance measure.
Memory Requirements

Fig. 4. The number of unique subtrees in each generation for all runs. Initially, with a population of full trees of depth 3, there are only three unique subtrees.
Conclusions

- addressed the *gap* between distance measures and "*true algorithmic distance*"

- formulated *specific operator-based distance* for GP syntax trees and subtree crossover

- considered *complexity reduction methods*

- demonstrated some properties of an operator-based distance and edit distance
Future Work

- applying operator-based distance to other problems
- tuning the complexity reduction methods
- evaluating accuracy
- incorporating operator-based distance into
  - *fitness distance correlation* research (Leonardo)
  - *diversity measure and method* research (Steven)