Fine-Grained Population Diversity Estimation for Genetic Programming Based Structure Identification

Stephan M. Winkler
Upper Austria University of Applied Sciences
School of Informatics, Communications and Media
Softwarepark 11
A-4232 Hagenberg, Austria
stephan@heuristiclab.com

Michael Affenzeller
Upper Austria University of Applied Sciences
School of Informatics, Communications and Media
Softwarepark 11
A-4232 Hagenberg, Austria
michael@heuristiclab.com

Stefan Wagner
Upper Austria University of Applied Sciences
School of Informatics, Communications and Media
Softwarepark 11
A-4232 Hagenberg, Austria
stefan@heuristiclab.com

ABSTRACT

We here describe a novel formalism for estimating the structural similarity of formulas that are evolved by a genetic programming (GP) based identification process. This method takes into account several aspects of structure tree comparison that are particularly important in the context of evolutionary system identification; this similarity measure is used for measuring the genetic diversity among GP populations.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics—*Process metrics*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

General Terms

Algorithms, Measurement, Design, Experimentation

Keywords

Genetic Programming, Data Mining, Machine Learning, Population Diversity Analysis, System Identification

1. FINE-GRAINED SIMILARITY ESTIMATION

The standard tree structures representation in GP makes it possible to use more fine grain structural measures that consider nodes, subtrees, and other graph theoretic properties; a comprehensive overview of program tree similarity and diversity measures has been given for instance in [1]. As an alternative, we have designed and implemented a method that systematically collects all pairs of ancestor and descendant nodes in structure trees representing mathematical models and information about the properties of these nodes. Additionally, for each pair we also document the distance (with respect to the level in the model tree) and the index of the ancestor's child tree containing the descendant node. The similarity of two models is then, in analogy to the method described in the previous section, calculated by comparing all pairs of ancestors and descendants in one model to all pairs of the other model and averaging the similarity of the respective best matches.

Figure 1 shows a simple formula and all pairs of ancestors and descendants included in the structure tree representing it; the input indices as well as the level differences ("level delta") are also given. Please note: The pairs given on the right side of Figure 1 are shown intentionally as they symbolize the pairs of nodes with level difference 0, i.e. nodes combined with themselves.

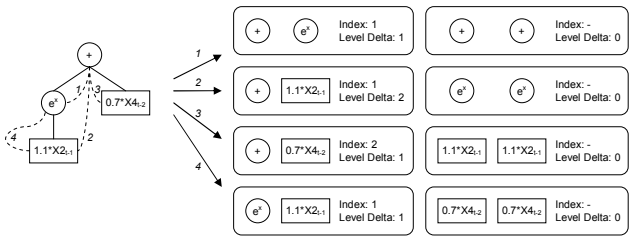

Figure 1: Simple formula structure and all included pairs of ancestors and descendants.

We define a *genetic item* as a 6-tuple storing the following information about the ancestor node a and descendant node d: The type of the node a ($type_a$), the type of the node d ($type_d$), the level delta (δl), the index of the child branch of a that includes d ($index$), the node parameters characterizing a (np_a), and the node parameters characterizing d (np_d). The parameters characterizing nodes are hereby represented by tuples containing the following information: For functions the variant (var), and for terminals the coefficient ($coeff$), the time offset (to), and the variable index (vi).

Now we can define the similarity of two *genetic items* gi_1 and gi_2, $s(gi_1, gi_2)$, as follows:

Most important are the types of the definitions referenced by the nodes; if these are not equal, then the similarity is 0 regardless of all other parameters:

$$\forall_{gi_1,gi_2} : gi_1.type_a \neq gi_2.type_a \Rightarrow s(gi_1, gi_2) = 0 \quad (1)$$

$$\forall_{gi_1,gi_2} : gi_1.type_d \neq gi_2.type_d \Rightarrow s(gi_1, gi_2) = 0 \quad (2)$$

This work was done within the research project L284-N04 "GP-Based Techniques for the Design of Virtual Sensors" sponsored by the Austrian Science Fund (FWF).

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

If the types of the nodes correspond correctly, then the similarity of gi_1 and gi_2 is calculated using the difference contributions $d_1 \dots d_{10}$ of the parameters of gi_1 and gi_2 and coefficients $c_1 \dots c_{10}$ whose use is to be explained later. The differences regarding input index, variant and variable index are not anyhow relativized, their similarity contribution is 1 in the case of equal parameters for both genetic items and 0 otherwise. The differences regarding level difference, coefficient and time offset, on the contrary, are indeed relativized:

- The level difference is divided by the maximum tree height $height_{max}$,
- the difference of coefficients is divided by the range of the referenced terminal definition (in case of uniformly distributed coefficients) or divided by the standard deviation σ (in case coefficients are normal distributed),
- and the difference of the time offsets is divided by the maximum time offset allowed $offset_{max}$.

$$\forall gi_1, gi_2 : (gi_1.type_a = gi_2.type_a) \,\&\, (gi_1.type_d = gi_2.type_d) \Rightarrow$$

$$d_1 = \frac{|gi_1.\delta l - gi_2.\delta l|}{height_{max}} \quad (3)$$

$$d_2 = \begin{cases} 0 & : \; gi_1.index \neq gi_2.index \\ 1 & : \; gi_1.index = gi_2.index \end{cases} \quad (4)$$

$$d_3 = \begin{cases} 0 & : \; gi_1.np_a.var \neq gi_2.np_a.var \\ 1 & : \; gi_1.np_a.var = gi_2.np_a.var \end{cases} \quad (5)$$

$$d_4 = \begin{cases} 0 & : \; gi_1.np_d.var \neq gi_2.np_d.var \\ 1 & : \; gi_1.np_d.var = gi_2.np_d.var \end{cases} \quad (6)$$

$$\delta c_a = |gi_1.np_a.coeff - gi_2.np_a.coeff| \quad (7)$$

$$d_5 = 1 - \begin{cases} if(isUniformTerminal(gi_1.type_a)) : \\ \quad \frac{\delta c_a}{gi_1.type_a.max - gi_1.type_a.min} \\ if(isGaussianTerminal(gi_1.type_a)) : \\ \quad \frac{\delta c_a}{gi_1.type_a.\sigma * 4} \end{cases} \quad (8)$$

$$\delta c_d = |gi_1.np_d.coeff - gi_2.np_d.coeff| \quad (9)$$

$$d_6 = 1 - \begin{cases} if(isUniformTerminal(gi_1.type_d)) : \\ \quad \frac{\delta c_d}{gi_1.type_d.max - gi_1.type_d.min} \\ if(isGaussianTerminal(gi_1.type_d)) : \\ \quad \frac{\delta c_d}{gi_1.type_d.\sigma * 4} \end{cases} \quad (10)$$

$$d_7 = 1 - \frac{|gi_1.np_a.to - gi_2.np_a.to|}{offset_{max}} \quad (11)$$

$$d_8 = 1 - \frac{|gi_1.np_d.to - gi_2.np_d.to|}{offset_{max}} \quad (12)$$

$$d_9 = \begin{cases} 0 & : \; gi_1.np_a.vi \neq gi_2.np_a.vi \\ 1 & : \; gi_1.np_a.vi = gi_2.np_a.vi \end{cases} \quad (13)$$

$$d_{10} = \begin{cases} 0 & : \; gi_1.np_d.vi \neq gi_2.np_d.vi \\ 1 & : \; gi_1.np_d.vi = gi_2.np_d.vi \end{cases} \quad (14)$$

Finally, there are two possibilities how to calculate the structural similarity of gi_1 and gi_2, $sim(gi_1, gi_2)$:

- When using the *additive* calculation, which is the obviously more simple way, $sim(gi_1, gi_2)$ is calculated as the sum of these similarity contributions $d_{1\dots10}$ weighted using the factors $c_{1\dots10}$ and divided by the sum of the weighting factors:

$$sim(gi_1, gi_2) = \frac{\sum_{i=1}^{10} d_i \cdot c_i}{\sum_{i=1}^{10} c_i} \quad (15)$$

- Otherwise, when using the more complicated *multiplicative* calculation method, we first calculate a punishment factor p_i for each d_i (again using weighting factors c_i, $0 \leq c_i \leq$ for all $i \in [1; 10]$) and then get the temporary similarity result sim_{tmp}:

$$\forall_{i \in [1;10]} : p_i = (1 - d_i) \cdot c_i \quad (16)$$

$$sim_{tmp}(gi_1, gi_2) = \prod_{i=1}^{10} (1 - p_i). \quad (17)$$

In the worst case scenario we get $d_i = 0$ for all $i \in [1; 10]$ and therefore the worst possible sim_{tmp} is

$$sim_{worst} = \prod_{i=1}^{10} (1 - ((1 - d_i) \cdot c_i)) = \prod_{i=1}^{10} (1 - c_i). \quad (18)$$

As sim_{worst} is surely greater than 0 we linearly scale the results to the interval $[0; 1]$:

$$sim(gi_1, gi_2) = \frac{sim_{tmp}(gi_1, gi_2) - sim_{worst}}{1 - sim_{worst}}. \quad (19)$$

In fact, we prefer this *multiplicative* similarity calculation method since it allows more specific analysis: By setting a weighting coefficient c_j to a rather high value (i.e., near or even equal to 1.0) the total similarity will become very small for pairs of genetic items that do not correspond with respect to this specific aspect, even if all other aspects would lead to a high similarity result.

For comparing models m_1 and m_2 we collect all pairs of ancestors and descendants (up to a given maximum level difference) in m_1 and m_2 and look for the best matches in the respective opposite model's pool of genetic items. For each genetic item gi_1 in the structure tree of m_1 we elicit that genetic item gi_x in the model structure m_2 with the highest similarity to gi_1; the similarity values $\mathbf{s}$ are collected for all genetic items contained in m_1 and their mean value finally gives us a measurement for the structure based similarity of the models m_1 and m_2, $sim(m_1, m_2)$.

2. GENETIC DIVERSITY

We finally describe the measures which we use to monitor the diversity and population dynamics with respect to the genetic make-up of solution candidates; we hereby use the similarity measures described in Section 1. As we know that both these similarity functions are not symmetric, we can alternatively use the mean value of the two possible similarity calls and so define a symmetric similarity measurement:

$$symmetricAnalysis \Rightarrow$$
$$sim(m_1, m_2) = \frac{sim(m_1, m_2) + sim(m_2, m_1)}{2} \quad (20)$$

In the context of single-population GP we are mainly interested in the similarity among the individuals of the population: For each model m of the population P we calculate the mean and the maximum similarity with all other individuals in the population:

$$meanSim(m, P) = \frac{1}{|P| - 1} \sum_{m2 \in P, m2 \neq m} sim(m, m2) \quad (21)$$

$$maxSim(m, P) = max_{(m2 \in P, m2 \neq m)} sim(m, m2) \quad (22)$$

The mean values of all individuals' similarity values are used for calculating the mean and maximum similarity measures for populations:

$$meanSim(P) = \frac{1}{|P|} \sum_{m \in P} meanSim(m, P) \quad (23)$$

$$maxSim(P) = \frac{1}{|P|} \sum_{m \in P} maxSim(m, P) \quad (24)$$

3. REFERENCES

[1] E. Burke, S. Gustafson, and G. Kendall. A survey and analysis of diversity measures in genetic programming. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 716–723, New York, 2002. Morgan Kaufmann Publishers.