Identifying Structural Mechanisms in Standard Genetic Programming

Jason M. Daida and Adam M. Hilss

Center for the Study of Complex Systems and the Space Physics Research Laboratory
The University of Michigan, 2455 Hayward Avenue, Ann Arbor, Michigan 48109-2143

Abstract. This paper presents a hypothesis about an undiscovered class of mechanisms that exist in standard GP. Rather than being intentionally designed, these mechanisms would be an unintended consequence of using trees as information structures. A model is described that predicts outcomes in GP that would arise solely from such mechanisms. Comparisons with empirical results from GP lend support to the existence of these mechanisms.

1 Introduction

Aside from crossover and selection, are there other mechanisms that might determine a broad variety of phenomena in standard GP?

At first glance, the answer might be "probably not." GP uses Darwinian natural selection and genetic crossover as metaphors for computation. Its design lineage stems from genetic algorithms (GA). Its processes are more similar with other algorithms in genetic and evolutionary computation than they are different. The question suggests the unusual possibility that there are additional mechanisms at work beyond that which has been intentionally engineered. In other words, given a standard GP system that has the mechanisms of crossover and selection and not much else, the question insinuates that other fundamental mechanisms might exist.

Even upon closer scrutiny, the answer might still be "probably not," although there has been previous work that suggests that there are other considerations that need to be made. The line of investigation by Soule and Foster (e.g., [1]) implied that something else is at work, while Langdon et al. has hypothesized that the causes for what Soule and Foster have found was a consequence of a random walk on the distribution of tree shapes (see [2, 3]).

However, according to this paper, the answer would be "yes, additional mechanisms do exist." We offer a hypothesis that identifies several of these mechanisms, all of which exist at a layer of abstraction below that of the biological metaphors that have given rise to GP. We further argue that not only do these mechanisms exist, but that these mechanisms can profoundly determine the extent to which GP can search.

This paper is organized as follows. Section 2 qualitatively describes our hypothesis. Section 3 describes a model that is a consequence of this hypothesis. Section 4 details the consequences of applying the model to GP search space and offers predictions of GP behavior in this search space. Section 5 compares the predictions with empirical data. Section 6 concludes.

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 1639–1651, 2003.

2 A Hypothesis

2.1 Trees as Information Structure

Our hypothesis involves the existence of mechanisms that are, on one hand, pervasive and are fundamental to standard GP and many of its derivatives. On the other hand, our hypothesis requires the discovery of mechanisms that are unknown. This discovery would need to take place in an engineered system in which every single aspect of that system has been open to inspection, and has been so for over a decade. How is it possible that mechanisms exist if we never put them there in the first place?

The answer, surprisingly, stems from the choice of using trees as an information structure. In evolutionary computation, the issue of trees would typically fall under the category of *problem representation* (e.g., [4]). One of the basic considerations in evolutionary computation is the matter of choosing a representation that is suited for solving a particular problem [5]. The manner in which one represents an appropriate solution—say, a vector of real values—in turn drives the type of operators, the type of objective function, even the type of evolutionary algorithm to be used [6]. Representations, and their corresponding evolutionary algorithms, are graded according to their worth in solving a particular problem. In the post-NFL era, one takes an egalitarian view of representation and evolutionary algorithms: there is no single representation or evolutionary algorithm that is universally superior to the rest when it comes to optimization [7].

However, the matter of trees as information structure is distinct from the matter of trees as an issue of problem representation. When one does an analysis in problem representation, one implicitly assumes that information from a problem's domain is a factor throughout that analysis. When one does an analysis in information structure, that assumption is not a given. Instead, it is common to treat trees as mathematical entities apart from the information such trees would carry. Consequently, it is not unusual to treat information structure as a level of abstraction below that of problem representation (as in [8]).

Whereas there are several problem representations that have been of interest in evolutionary computation (i.e., binary strings, real-valued vectors, finite-state representations, and parse trees), there are primarily two information structures that are fundamental to computer science: linear lists and trees. Each type of information structure has its own mathematical heritage. Each type of information structure also has a set of techniques for representing and manipulating such a structure within a computer ([8], p. 232).

Trees as information structures represent an extensive body of literature that has been, for the most part, separate from work in GP.[1] Trees were likely first formalized as a mathematical entity in [13]. The term *tree* and the beginnings of an extensive treatment of trees as a mathematical entity likely started with [14]. Trees overtly represented information in computer memory in some of the earliest machine-language

[1] Langdon has used results from [9-10] in his analyses of GP performance. His incorporation of the mathematical literature seems to be exception and not the rule in GP theory. In evolutionary computation at large, incorporation of the literature has been mostly with respect to finding alternative representations for use in various evolutionary algorithms (e.g., [11–12]).

computers for the purpose of algebraic formula manipulation ([8], 459). Because tree structures were well suited for formula manipulation, these structures were used in some of the 1950's work in automatic programming [15].

2.2 Visualizing Tree Structures

To appreciate the role of tree structure in forming solutions in GP, it helps to visualize the kinds of structures that occur. Visualizing structure, particularly for typical trees encountered in GP, can be done by mapping *m*-ary trees (binary trees for this paper) to a circular grid. For binary trees, this grid can be derived by starting with a full binary tree, which we designate as *C*.

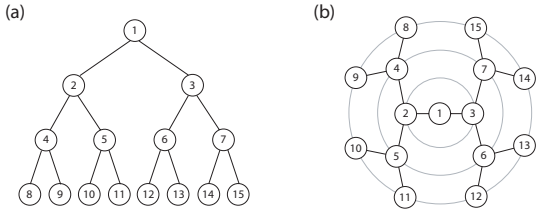

Fig. 1. Mapping a full binary tree to a circular grid. (a) Full binary tree of depth 3. (b) Corresponding circular grid

The nodes of this labeled, full binary tree *C* can be mapped to a circular grid in polar coordinates (r, θ) by first assigning depth levels to a set of concentric rings. The exact mapping of depth levels to ring radii is arbitrary, so long as depth level 0 remains at the center and increasing depths are assigned to rings of increasing radius. For convenience, we let depth level *n* be mapped to a ring of radius *n*. Figure 1 gives an example of a full binary tree of depth 3 that has been labeled in this manner.

An arbitrary binary tree *A* can be mapped to labels of a full binary tree by showing that a set of labels corresponding to *A* is a subset of labels corresponding to *C*. (Note: a formal description of our visualization method is found in [16].)

Figure 2 shows an example of a full binary tree of depth 10 (2047 nodes) and an example of binary tree of depth 26 (461 nodes) from GP. From the standpoint of typical measures of tree structure (i.e., *number of nodes* and *depth level*), the GP tree is unremarkable. The GP binary tree was generated under circumstances that were also unremarkable and is a solution from a symbolic regression problem that has been documented elsewhere in the literature [17–20].

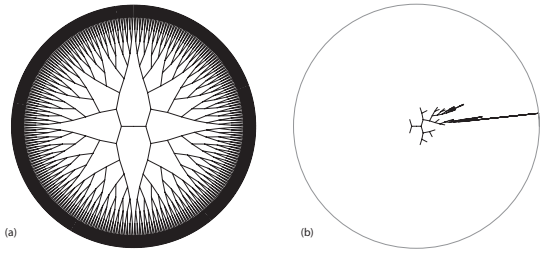

Fig. 2. Two examples of binary trees. (a) Full binary tree, depth 10. (b) Structure of a GP-generated solution (using arity-2 functions), depth 26

Figure 3 shows the individual in Fig. 2b in the context of 11 other individuals that have been taken from the same population of 500. Visualization of this population (not shown) depicts much similarity between tree shapes in a population.

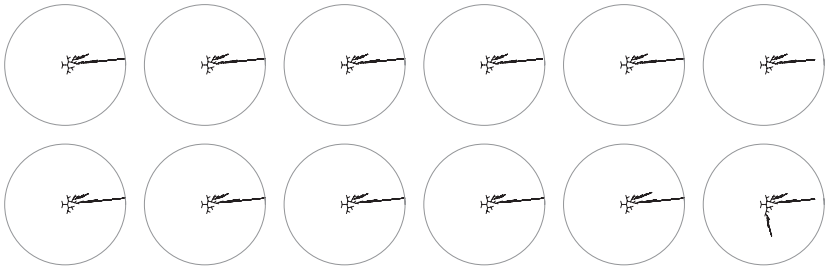

Fig. 3. Sample of a population. These are 12 representatives out of a population of 500. The individual that is shown in Fig. 2b is the second from the top left corner. The remaining 488 individuals look similar to these

In the following section, we pose a hypothesis that may account for the patterns hinted at in these figures.

2.3 Trees and Structural Mechanisms: A Hypothesis

We hypothesize that there exists a class of structural mechanisms that "piggyback" on the operators that generate variation in GP (e.g., crossover and mutation). These structural mechanisms incrementally modify the shape of the tree structures during iteration. In so doing, these mechanisms incrementally modify the probabilities of choosing a subtree that are distributed across a particular (*m*-ary) tree structure. In standard GP, probabilities of choosing a subtree are distributed across nodes as either uniform or piecewise uniform. (In many GP systems, there is a user-determined bias — sometimes called *internal node bias* – that weights the probability of selecting an internal node against that of terminal node. The conditional probability of choosing for either type of node is uniform within that type.) As a result, for (disjoint) subtree partitions at a given depth, the probabilities of choosing a subtree at that depth are biased toward those subtrees with more terminal nodes. This would generally hold true even under internal node bias, since subtrees with more terminal nodes often have more internal nodes. Consequently, this shift in probability, together with mechanisms that add structure, can feedback positively for the growth of deeper subtrees. This type of random growth would subsequently result in a non-uniform distribution in the space of allowable tree structures. In other words, the trees are inherently sparse.

We can illustrate various aspects of this hypothesis by starting out with an example of "piggybacking" in crossover and mutation. Figure 4 shows how a structurally equivalent operation can result from two different operators in GP. For crossover: Fig. 4a depicts Parent 1 binary tree with an arrow indicating the crossover point; Fig. 4b, Parent 2 binary tree with an arrow indicating the crossover point; Fig. 4c, Child binary tree; Fig. 4d, the oriented equivalent with bold indicating the added structure relative to Parent 1. For mutation: Fig. 4e depicts Parent 1 binary tree with an arrow indicating the point at which mutation starts; Fig. 4f, Child plane tree; Fig. 4g, the oriented equivalent with bold indicating the added structure relative to Parent 1. Both 4d and 4g show structural equivalents of adding a three-node binary

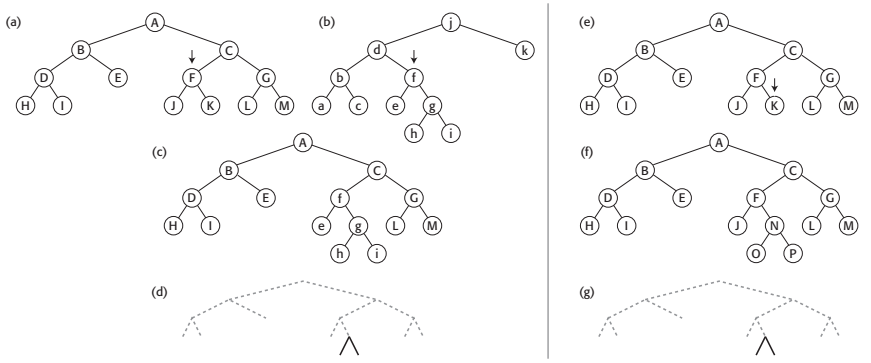

Fig. 4. Crossover and mutation. Crossover and mutation can result in a structurally equivalent operation that adds a common (oriented) subtree

subtree to Parent 1. The degree and content of each node is arbitrary, as is the location of crossover and mutation points, so long as their orientation remains fixed.

We would consider structurally equivalent operators—like the one that has been discussed in Fig. 4 – as belonging to the class of mechanisms that we have hypothesized. Namely, here is an example of a mechanism that "piggybacks" on the operators like crossover and mutation.

Such operators can explain the preferential growth of some parts of a tree over others. Figure 5 indicates two subtrees that are labeled 1 and 2 at their respective roots. Assume that the probability of picking any node is uniform. Further assume, for illustrative purposes only, that whatever the variational operator, the net effect is to increase a subtree by a three-node binary tree. In Fig. 5a, the probability of picking a node from either subtree is equal ($p = 1/3$). In Fig. 5b, the probability of picking a node in subtree 2 is 3 times greater than in picking a node from subtree 1 ($p = 3/5$ *v.* $p = 1/5$). In Fig. 5c, the probability of picking a node in subtree 2 is 5 times greater ($p = 5/7$ *v.* $p = 1/7$). In Fig. 5d, the probability of picking a node in subtree 2 is 7 times greater ($p = 7/9$ *v.* $p = 1/9$). The longer a subtree is not selected in this process of incremental growth, the less likely it is for this subtree to grow.

We can extend this thought experiment by continuing this iteration. To simplify matters, we can presume to replace a randomly selected terminal node with a three-node binary subtree. Figure 6 shows a result of this.

This iterative process results in a tree that compares well to the visualization of actual trees in Figs. 2 and 3. However, this thought experiment still falls short of having a verifiable model. Towards this end, then, the next section describes our model.

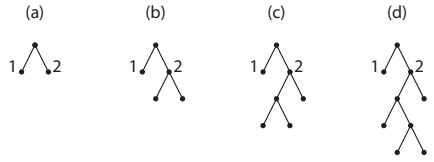

Fig. 5. Preferential growth between subtrees

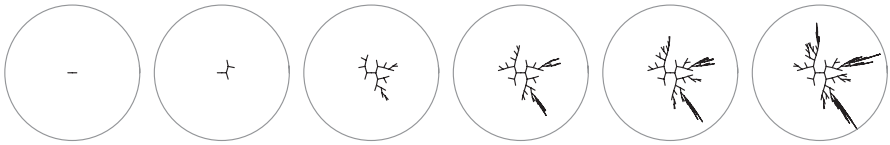

Fig. 6. Growth of a binary plane tree. This figure depicts six "snapshots" during the growth of a binary tree. The outer gray ring is for reference only and corresponds to depth 15

3 The Lattice-Aggregate Model

The lattice-aggregate model formalizes the hypothesis set forth in the previous section. It uses set-theoretic operations on ordered sets of integers to describe the iterative, stochastic growth of *m*-ary trees as encountered in processes like GP. As in Sect. 2, we simplify our discussion by considering binary trees only.

3.1 Basic Description

Let the root node be defined at depth $d = 0$. Assuming that tree T has depth $d > 0$, the set A of nodes of T can be decomposed into two mutually exclusive, non-empty sets J and K such that

- Set J corresponds to the internal nodes of T
- Set K corresponds to the leaves of T

We define a set B to correspond to a subtree of T. Note that set B is a function of k, whereupon $k \in A$. The smallest possible subtree, a leaf, can be described as

$$B^1 = \{k\},\ k \in \Im^+,\ k \in A\,. \tag{1}$$

The next smallest subtree, a parent and two nodes, can be described as

$$B^3 = \{k,\ 2k,\ 2k + 1\}. \tag{2}$$

For the purposes of modeling GP behaviors[2] for depths 0 – 26, we arbitrarily define B^5, B^7, B^9, and B^{11} to correspond to 5-, 7-, 9-, and 11-node subtrees, respectively.[3]

$$B^5 = \{k,\ 2k,\ 2k + 1,\ 4k + 3,\ 4k + 4\}. \tag{3}$$

$$B^7 = B^5 \cup \{8k + 5,\ 8k + 6\}. \tag{4}$$

$$B^9 = B^7 \cup \{16k + 11,\ 16k + 12\}. \tag{5}$$

$$B^{11} = B^9 \cup \{32k + 23,\ 32k + 24\}. \tag{6}$$

(Note that the particular selection of elements for B^5, B^7, B^9, and B^{11} seems somewhat arbitrary. These subtrees represent a minimal set of subtrees that can account for

[2] Depth 26 was chosen largely on the basis of comparison with known empirical data sets.

[3] For depths larger than 26, we would need to consider larger and deeper subtrees, since the opportunity to pick a deeper subtree exists for a larger and deeper tree.

much of the behavior that has been observed in GP at depths 1–26. A fuller account for how these subtrees were chosen has been deferred to a future paper.)

Now let $k \in$ K. We can then represent the growth of a tree by B^5 as

$$A' = A \cup B^5(k) = \text{A} \cup \{2k + 4k + 3\}. \quad (7)$$

Likewise, we can do the same for B^7, B^9, and B^{11}.

Consequently, we can represent a stochastic model of tree growth for depths 0 – 26 as a recursive operation upon integer sets, namely:

$$A' = A \cup B^{i}(k), \quad (8)$$

where i is a discrete random variable with sample space $S_{\text{B}} = \{5, 7, 9, 11\}$ and k is a discrete, uniformly distributed random variable with sample space $S_{\text{B}} =$ K. It can be demonstrated that an appropriate probability distribution function corresponding to i entails the following relationship[4]:

$$P(\text{i} = 5) = 2\,P(i = 7) = 4\,P(i = 9) = 8\,P(i = 11). \quad (9)$$

Example. Given $A = \{1, 2, 3, 6, 7, 12, 13, 14, 15\}$. Set A decomposes into $J = \{1, 3, 6, 7\}$ and K = {2, 12, 13, 14, 15}. Assuming that the second element of K and that B^5 have been chosen, A' = {1, 2, 3, 6, 7, 12, 13, 14, 15, 24, 25, 50, 51}.

3.2 Variations

There are several additional variations that need to be considered in the modeling of tree growth in GP. The first set of variations assists in identifying the upper and lower density bounds of tree growth, while the second set of variations address methods of population initialization.

The density of a set A can be defined as follows:

$$\text{density} \equiv \frac{\text{N(A)}}{2^{\log_2(\max(\text{A})+1)} - 1}, \quad (10)$$

where N(A) is the number of elements in A and max(A) identifies the maximum value in A. This definition corresponds to a ratio that is the number of nodes of a tree that is normalized by the number of nodes in a full tree of identical depth.

To identify the upper density bound, equation (8) can be restated as

$$A' = A \cup B^3(k). \quad (11)$$

Equation 11 corresponds to the case in which tree growth is entirely determined by three-node subtrees. Note that if k was deterministic (instead of stochastic) such that all $k \in K$ is selected for replacement by B^3, the resulting tree would approach being full.

To identify a lower density bound, equation (8) can be restated as

[4] This assumes that the comparison is with standard GP, in which the probability of choosing an internal node for crossover is uniform. These probabilities were determined through the following coarse approximation: given a full binary tree, what is the probability of selecting a subtree of depth j relative to the probability of selecting a subtree of depth j –1?

$$A' = A \cup B^n(\boldsymbol{k}), \tag{12}$$

where B^n is the least dense set of those sets B that are used in modeling growth. It is assumed that the density for sets B are determined at $k = 1$. For the proposed model for depths 0 – 26, the set that is least dense is B^{11}.

It is possible to modify equation (8) to account for varying methods of population initialization. While such modifications have been done to model Koza's ramped half-and-half for depths 2–6, the exposition of these modifications have been left to a future paper.

4 Model Predictions

4.1 Determination of Search Space Boundaries

The lattice-aggregate model was subsequently used to derive boundaries in the size-shape search space of trees from depths 0 – 26. This derivation consisted of four steps, namely:

- Used Monte Carlo methods to sample the proposed lattice-aggregate model corresponding to equation (11).
- Extracted depth and number of nodes from each sample.
- Computed the cumulative distribution of the numbers of nodes per tree at a given depth d. Repeat for $d = \{0, 1, 2, \ldots 26\}$.
- Determined isopleths in size-shape space that correspond to contours of constant distribution.

This process is shown in Fig. 7 for 50,000 samples. Isopleths were generated for 99%, 75%, median, 25%, and 1% distributions. Given the relatively steep fall-offs in distribution, the 99% and the 1% isopleths do approximate boundaries that specify where trees do or do not occur in this search space.

A similar procedure was applied to determine isopleths for equations (11) and (12). Again, given relatively steep fall-offs in distribution, the 99% isopleth for equation (11) approximated the uppermost bound of search, while the 1% isopleth for equation (12) approximated the lowermost bound of search.

4.2 Model Results

Figure 8 summarizes the isopleths for equations (8), (11), and (12). The isopleths suggest the existence of at least four distinct regions for depths 0 – 26. The regions are as follows:

- *Region I.* This is the region where most solutions in standard GP would occur (for binary trees). Full mixing of various size / shape subtrees in the derivation of solutions occurs here. The width of Region I is driven largely by population initialization.
- *Region II*. These are the regions (II_a and II_b) where increasingly fewer individuals would appear the further these individuals are from Region I. Only partial mixing of size / shape subtrees occurs here, with mixing becoming non-existent towards the boundaries furthest away from Region I. Region II_a is delineated by the boundaries that are approximately located by the 99% isopleth for equation (11)

and the 99% isopleth for equation (8). The transitions between Regions II and III are pronounced.

- *Region III*. These are the regions (III_a and III_b) where even fewer individuals would typically appear. Region III_a is delineated by the boundaries that are approximately located by the 99% isopleth for equation (11) and the limit for full trees. Region III_b is delineated by the boundaries that are approximately located by the 1% isopleth for equation (12) and the limit for minimal trees.
- *Region IV*. These are the regions (IV_a and IV_b) that are precluded from binary trees.

5 Empirical Comparison

We compared the map of predicted regions against results obtained from common test problems. In particular, we considered *quintic*, *sextic*, *6-input multiplexer*, and *11-input multiplexer*. Furthermore, we placed the outcomes from these problems in the context of both our theory and in the prevailing theory [3, 25]. The setup and implementation of *quintic* and *sextic* are from [21]; the *multiplexer* problems that use arity-2 functions, [22].

The GP parameters used were Koza's [21]: population 4,000; tournament selection $M = 7$; maximum generations 50. The maximum depth was set to 512. We used a modified version of lilgp [23] as was used in [17]. Most of the modifications were for bug fixes and for the replacement of the random number generator with the Mersenne Twister [24]. We configured lilgp to run as a single thread. Internal node bias was set at default (0.9).

Figure 9 shows the results for these test problems. Each dot represents a best-of-trial individual out of a

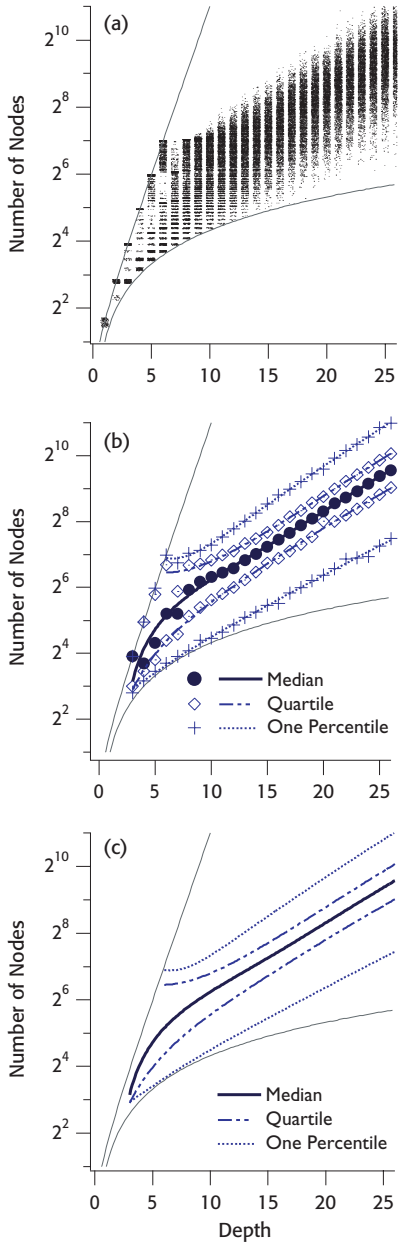

Fig. 7. Derivation of isopleths (Region I). Top plot shows the Monte Carlo results. Middle shows the numerical values of constant distribution and fitted isopleths. Bottom shows just the fitted isopleths

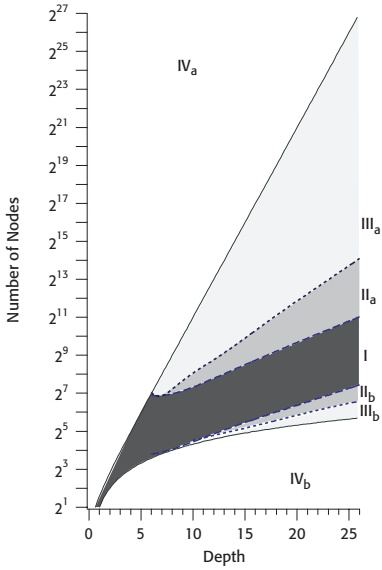

Fig. 8. Predicted regions

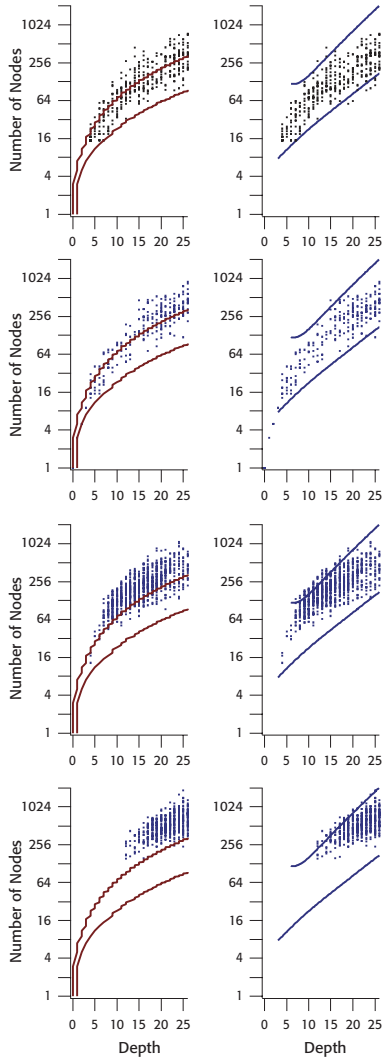

Fig. 9. Empirical comparison

population of 4,000. Each graph represents the ensemble performance of 1,000 trials (i.e., a sampling of four million individuals total per graph). The first row corresponds to the *quintic* problem; the second row, *sextic*; the third, *6-input multiplexer*; and the fourth, *11-input multiplexer*. The data in the left- and right-side graphs of each row are the same. What differ are the boundaries that have been superimposed. On the left side are the 5% and 95% boundaries as indicated by the prevailing theory [3, 25]. On the right side are the Region I boundaries as predicted by our model. Note that samples that occur at depths greater than 26 are not shown, since the map of predicted regions has been generated using an approximation that is not valid for depths greater than 26 (i.e., B^5, B^7,…, B^{11}). At depths greater than 26, a better approximation would need to be used and is left for future work.

Most of the data shown in Fig. 9 do fall within the Region I boundaries. In particular, for *quintic*, 97% of the depicted samples fall within the Region I boundaries in comparison to 61% for the prevailing theory. For *sextic*, about 95% fall within the Region I boundaries, as in comparison to 43% for the prevailing theory. For *6-input multiplexer*, Region I accounts for 93%, in comparison to the prevailing theory (15%).

For *11-input multiplexer*, Region I accounts for 92%, in comparison to the prevailing theory (0.62%).

Another comparison with empirical results is given in this proceedings [27]. This work introduces a test called the *Lid* problem, which is anticipated by our hypothesis and model. Indeed, if structure alone accounts for significant GP behavior, it should be possible to devise a structure-only problem in GP that demonstrates this. Other empirical comparisons can be found in [28].

6 Conclusions

This paper described a hypothesis of an undiscovered class of mechanisms that exist in standard GP. Rather than being intentionally designed, these structural mechanisms have been hypothesized to be an unintended consequence of using trees as information structures. To test this hypothesis, a model has been described that predicts outcomes in GP that would arise solely from such mechanisms.

There are several other conclusions that can be made.

- *The model is a logical progression in the consideration of tree structure in GP.* Although the idea of undiscovered mechanisms may seem farfetched, they are consistent with the idea of random, diffusion processes in GP. The difference, however, between our hypothesis and the prevailing theory [3, 25] is the inclusion of iterative growth in this otherwise random process. So instead of a random walk on the distribution of tree shapes, the random walk itself determines tree shape distribution because GP trees grow iteratively. This seemingly minor distinction is analogous to the distinction in physics between purely diffusive processes and diffusion-limited aggregation [26].
- *The hypothesis and model have suggested a method of visualizing tree structures in GP*. The method was briefly mentioned in this paper and in [28]. A detailed account is given in this proceedings [16].
- *The empirical results support our hypothesis and model for the existence of structural mechanisms in GP.* The model predicts a region where most outcomes of GP would be found (i.e., Region I). This would occur because Region I is where all of the available subtree shapes could be used to form a solution. Because the derivation for this region is based only on structure and not programmatic content, this region should be applicable to a broad variety of GP problems that use just arity-2 functions. The empirical results given in this paper do support the existence of Region I, with that region accounting for better than 90% of all solution outcomes of the experiment.
- *The hypothesized mechanisms provide a simple explanation of the evolution of size and shape in standard GP.* Arguably, this explanation is simpler than that provided under the prevailing theory. The prevailing theory implies that selection pressure on content accounts for the discrepancies. While the results suggest that selection pressure on content is still a factor within Region I, such pressure remains within the context of a consistent structural account for size and shape.
- *The existence of the other predicted regions is not supported by the empirical results*. Although the empirical results do provide support for our hypothesis and model, the results fall short of what is needed to demonstrate the existence of those regions. The empirical results subsequently fall short of validating the exis-

tence of structural mechanisms. However, as it has turned out, obtaining such evidence is nontrivial. Another paper [27] more directly addresses this issue than has been possible here.

Acknowledgments. We thank the following individuals and organizations for their help: CSCS U-M / Santa Fe Institute Fall Workshops, L.M. Sander, S. Stanhope, J. Polito 2, P. Litvak, S. Yalcin. D. Maclean, W. Worzel, M. Samples, and UMACERS teams Magic, Meta-Edge, Royal, Borges, and Binomial-3. We appreciate the critiques extended to us from the anonymous reviewers, E. Korkmaz, K. Seo, and N. McPhee. The first author also acknowledges I. Kristo and S. Daida.

References

[1] T. Soule, et al., "Code Growth in Genetic Programming," in *GP 1996*, J.R. Koza, et al., Eds. Cambridge: The MIT Press, 1996, pp. 215–223.

[2] W.B. Langdon, et al., "The Evolution of Size and Shape," in *Advances in Genetic Programming 3*, L. Spector, et al. Eds. Cambridge: The MIT Press, 1999, pp. 163–190.

[3] W.B. Langdon and R. Poli, *Foundations of Genetic Programming*. Berlin: Springer-Verlag, 2002.

[4] P.J. Angeline, "Parse Trees," in *Handbook of Evolutionary Computation*, T. Bäck, et al., Eds. Bristol: Institute of Physics Publishing, 1997, pp. C1.6:1–C1.6:3.

[5] K. Deb, "Introduction [to Issues in Representations]," in *Handbook of Evolutionary Computation*, T. Bäck, et al., Eds. Bristol: Institute of Physics Publishing, 1997, pp. C1.1:1–C1.1:4.

[6] D.B. Fogel and P.J. Angeline, "Guidelines for a Suitable Encoding," in *Handbook of Evolutionary Computation*, T. Bäck, et al., Eds. Bristol: Institute of Physics Publishing, 1997, pp. C1.7:1–C1.7:2.

[7] D.H. Wolpert and W.G. Macready, "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67–82, 1997.

[8] D.E. Knuth, *The Art of Computer Programming: Volume 1: Fundamental Algorithms*, vol. 1, Third ed. Reading: Addison–Wesley, 1997.

[9] L. Alonso and R. Schott, *Random Generation of Trees*. Boston: Kluwer Academic Publishers, 1995.

[10] P. Flajolet and A. Odlyzko, "The Average Height of Binary Trees and Other Simple Trees," *Journal of Computer and System Sciences*, vol. 25, pp. 171–213, 1982.

[11] J. Knowles and D. Corne, "A New Evolutionary Approach to the Degree Constrained Minimum Spanning Tree Problem," *IEEE Transactions on Evolutionary Computation*, vol. 4, pp. 125–134, 2000.

[12] G.R. Raidl, "A Hybrid GP Approach for Numerically Robust Symbolic Regression," in *GP 1998*, J. R. Koza, et al., Eds. San Francisco: Morgan Kaufmann Publishers, 1998, pp. 323–328.

[13] G. Kirchoff, *Annalen der Physik und Chemie*, vol. 72, pp. 497–508, 1847.

[14] A. Cayley, *Collected Mathematical Papers of A. Cayley*, vol. 3, 1857.

[15] H. G. Kahrimanian, presented at Symposium on Automatic Programming, Washington, D.C., 1954.

[16] J.M. Daida and A. Hilss,"Visualizing Tree Structures in Genetic Programming," *GECCO 2003.*

[17] J.M. Daida, et al., "Analysis of Single-Node (Building) Blocks in Genetic Programming," in *Advances in Genetic Programming 3*, L. Spector, et al., Eds. Cambridge: The MIT Press, 1999, pp. 217–241.

[18] J.M. Daida, et al., "What Makes a Problem GP-Hard? Analysis of a Tunably Difficult Problem in GP," in *GECCO 1999*, W. Banzhaf, et al., Eds. San Francisco: Morgan Kaufmann Publishers, 1999, pp. 982–989.

[19] O.A. Chaudhri, et al., "Characterizing a Tunably Difficult Problem in Genetic Programming," in *GECCO 2000*, L. D. Whitley, et al. Eds. San Francisco: Morgan Kaufmann Publishers, 2000, pp. 395–402.

[20] JM. Daida, et al., "What Makes a Problem GP-Hard? Analysis of a Tunably Difficult Problem in Genetic Programming," *Journal of Genetic Programming and Evolvable Hardware*, vol. 2, pp. 165–191, 2001.

[21] J. R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge: The MIT Press, 1994.

[22] W.B. Langdon, "Quadratic Bloat in Genetic Programming," in *GECCO 2000*, L. D. Whitley, et al., Eds. San Francisco: Morgan Kaufmann Publishers, 2000, pp. 451–458.

[23] D. Zongker and W. Punch, "lilgp," v. 1.02 ed. Lansing: Michigan State University Genetic Algorithms Research and Applications Group, 1995.

[24] M. Matsumoto and T. Nishimura, "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudorandom Number Generator," *ACM Transactions on Modeling and Computer Simulation*, vol. 8, pp. 3–30, 1998.

[25] W.B. Langdon, "Size Fair and Homologous Tree Crossovers for Tree Genetic Programming," *Journal of Genetic Programming and Evolvable Machines*, vol. 1, pp. 95–119, 2000.

[26] T. A. Witten and L. M. Sander, "Diffusion-Limited Aggregation: A Kinetic Critical Phenomenon," *Physics Review Letters*, vol. 47, pp. 1400–1403, 1981.

[27] J.M. Daida, et al., “What Makes a Problem GP-Hard? Validating a Hypothesis of Structural Causes,” *GECCO 2003*.

[28] J.M. Daida, “Limits to Expression in Genetic Programming: Lattice-Aggregate Modeling,” *CEC 2002*, Piscataway: IEEE Press, pp. 273–278.