Chapter 2
Identification of Novel Genetic Models of Glaucoma Using the "EMERGENT" Genetic Programming-Based Artificial Intelligence System

Jason H. Moore, Casey S. Greene and Douglas P. Hill

2.1 Introduction

Primary open-angle glaucoma (POAG) is a common eye disease that is characterized by an increase in intraocular pressure that, if untreated, can lead to a decrease in vision or even blindness due to progressive nerve damage. Family studies have shown that POAG has a heritable component with siblings of those affected having 5-10 times the risk of randomly selected people from the same population (Wang et al. 2010). As recently reviewed (Cooke Bailey et al. 2013), at least five genomic regions have been associated with POAG in large genome-wide association studies (GWAS). While these initial genetic studies provide some clues they do not come close to explaining the variability in risk due to genetic variation. A significant limitation of these genetic studies is that they consider each genetic variant or polymorphisms individually ignoring both genomic and ecological contexts that are likely to influence how a particular region of the genome impacts risk through a complex hierarchy of biological systems. The goal of the present study is to more fully explore the genotype-phenotype relationship in a genome-wide genetic study of POAG that embraces, rather than ignores, the complexity of the disease. It is our working hypothesis that alternatives to the one genetic variant at a time framework implemented using parametric statistical approaches such as logistic regression will reveal interesting new associations that will spark future work in this area leading to new predictive models and perhaps increased biological understanding that will open the door to new treatments.

J. H. Moore (✉) · C. S. Greene · D. P. Hill
The Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, 19104-6021, USA
e-mail: jhmoore@upenn.edu

C. S. Greene
e-mail: casey.s.greene@dartmouth.edu

D. P. Hill
e-mail: douglas.hill@Dartmouth.edu

R. Riolo et al. (eds.), *Genetic Programming Theory and Practice XII,*
Genetic and Evolutionary Computation, DOI 10.1007/978-3-319-16030-6_2

The availability of big data in disciplines such as human genetics has rekindled an interest in artificial intelligence (AI) and machine learning methods that are capable of identifying complex relationships among measured features such as genetic variation. This need for more powerful modeling approaches coupled with the availability of inexpensive high-performance computing and improved human-computer interaction (HCI) technology means that the timing is perfect to employ these methods for looking at large-scale human genetics data. A key feature of AI research is the desire to create a computational system that can reason or make decisions as a human would. This is in significant contrast to the current GWAS strategy that carries out each genetic analysis without any input from a human or their abundant expert knowledge base. The goal of the present study is to utilize an AI approach to the genetic analysis of POAG that explores complex relationships in the data in a manner that is much more consistent with how humans would approach manual data analysis given effectively infinite time.

We introduce here the Exploratory Modeling for Extracting Relationships using Genetic and Evolutionary Navigation Techniques (EMERGENT) algorithm as an AI approach to the genetic analysis of common human diseases. At the heart of EMERGENT is a symbolic discriminant analysis (SDA) approach (Moore et al. 2002; Moore et al. 2007) that performs classification using models constructed from a list of possible mathematical functions and a list of features or attributes. This base classification method is appealing because it makes no assumptions about the functional form of the model. This is in contrast to methods such as logistic regression that first assume a particular model to which all data are fitted. It is the base assumption of EMERGENT that human genetics data is sufficiently complex that we do not know what etiological models for diseases like POAG should look like beyond what has been learned from rare Mendelian diseases like cystic fibrosis where it is much easier to pin down the genetic cause. At this level, EMERGENT is like any other machine learning method that takes features as inputs and that produces a classifier for prediction. The goal of course is to optimize the selection of functions and features to maximize the classification accuracy of the model and to do so in a manner that more closely mimics human problem-solving. It is this last goal that distinguishes EMERGENT from other machine learning methods like decision trees or neural networks.

The framework EMERGENT uses for representing models and for exploring an effectively infinite model space is based on genetic programming (GP). Genetic programming is an automated computational discovery tool that is inspired by Darwinian evolution by natural selection (Banzhaf et al. 1998; Koza 1992). The goal of GP is to "evolve" computer programs to solve complex problems. This is accomplished by first generating or initializing a population of random computer programs that are composed of the basic building blocks needed to solve or approximate a solution to the problem. Genetic programming and its many variations have been applied successfully in a wide range of different problem domains including bioinformatics (Fogel and Corne 2003) and genetic analysis more specifically (Moore et al. 2007; Moore et al. 2008b; Ritchie et al. 2003). This is an attractive approach to the genetic analysis problem because it is inherently flexible, stochastic, parallel and easily

adapted to exploit expert knowledge of the type that humans would employ in their own modeling strategies. Genetic programming falls within the scope of AI as a computational intelligence method.

The key to EMERGENT as an AI approach is two-fold. First, as described above, GP provides a flexible way to represent solutions such as symbolic discriminant functions as computer programs. Second, GP provides a stochastic and parallel search based on the principles of evolution by natural selection. These first two characteristics are at the heart of the EMERGENT algorithm and are important for model discovery. Finally, while the algorithm is discovering good models, we want the system to learn how to generate good models. This final characteristic is a meta-layer that is inspired by how humans solve problems. That is, not only can we as humans solve a complex problem but we can learn at the same time strategies that make solving the problem faster and easier. To accomplish this we have implemented a type of GP called computational evolution that learns how to generate new models while it is learning what a good model is. Computational evolution has been previously reviewed (Banzhaf et al. 2006) and has been previously employed in GP systems such as PushGP (Spector and Robinson 2002). The key to our own implementation of computational evolution within the EMERGENT framework is the ability of the system to learn to use different sources of expert knowledge to help guide the search for new models. This is intended to mimic how humans draw on past experience to solve a problem.

In the next sections we outline our previous work with the EMERGENT system, the details of the EMERGENT algorithm as employed here and then its application to a genome-wide genetic study of POAG. Our results suggest both interesting and novel genetic associations for POAG that have not been previously reported.

2.2 History of the EMERGENT Framework

Development of the EMERGENT framework (formally described generically as a Computational Evolution System or CES) described in detail below has proceeded in multiple steps. Moore et al. developed the GP-based symbolic discrimination analysis (SDA) method for flexible classification of disease using genomics data (Moore et al. 2002). This was later extended to developing models of genetic variation predictive of common disease endpoints (Moore et al. 2007). This approach was extended to include some of the features of a computational evolution system (Banzhaf et al. 2006). We developed a hierarchical, spatially-explicit GP-based system that allows for the evolution of arbitrarily complex solutions and solution operators, and includes population memory via archives, feedback loops between archives and solutions, and environmental sensing (Greene et al. 2009a, b; Moore et al. 2008a; Moore et al. 2008c; Payne et al. 2010). Analyses of this system have demonstrated its ability to identify complex disease-causing genetic architectures in simulated data, and to recognize and exploit useful sources of expert knowledge. Specifically, we have shown that statistical expert knowledge, in the form of ReliefF scores (Moore and

White 2007), can be incorporated via environmental sensing (Greene et al. 2009b) and population initialization (Payne et al. 2010) to improve system performance. In addition, we recently showed that biological expert knowledge in the form of protein-protein interactions could be used to guide EMERGENT toward valid gene-gene interaction models (Pattin et al. 2010). We also showed how visualization of EMERGENT results could improve the modeling process (Moore et al. 2011). More recently, we have demonstrated how Pareto optimization (Moore et al. 2012) and measures of interestingness (Moore et al. 2013) can be used to improve the search for novel models. Here, we demonstrate how the EMERGENT framework derived from the studies summarized above can be used to look at the genetics of POAG in a novel manner.

2.3 Methods

In this section, we first present a summary of our EMERGENT AI framework for open-ended genetic analysis of complex human diseases. We then discuss our application of this approach to POAG.

2.3.1 Overview of the EMERGENT Framework

In Fig. 2.1, we provide a graphical overview of EMERGENT, which is both hierarchically organized and spatially explicit. The bottom level of the hierarchy consists of a lattice of solutions (Fig. 2.1D), which compete with one another within spatially-localized, overlapping neighborhoods. The second layer of the hierarchy contains a lattice of arbitrarily complex solution operators (Fig. 2.1C), which operate on the solutions in the lower layer. The third layer of the hierarchy contains a lattice of mutation operators (Fig. 2.1B), which modify the solution operators in the second layer, and the highest layer of the hierarchy governs the rate at which the mutation operators are modified (Fig. 2.1A). EMERGENT includes a source of expert knowledge (Fig. 2.1E) that can be used with the solution operators and as part of Pareto optimization (Fig. 2.1F). EMERGENT also possesses an attribute archive, which stores the frequencies with which attributes are used. The solution operators can then exploit these data to bias the construction of solutions toward frequently utilized attributes. We did not use the attribute archive in the present study.

2.3.2 Solution Representation, Fitness Evaluation, Selection, and Pareto Optimization

Each solution represents a classifier, which takes a set of SNPs as input and produces an output that can be used to assign diseased or healthy status. These solutions are

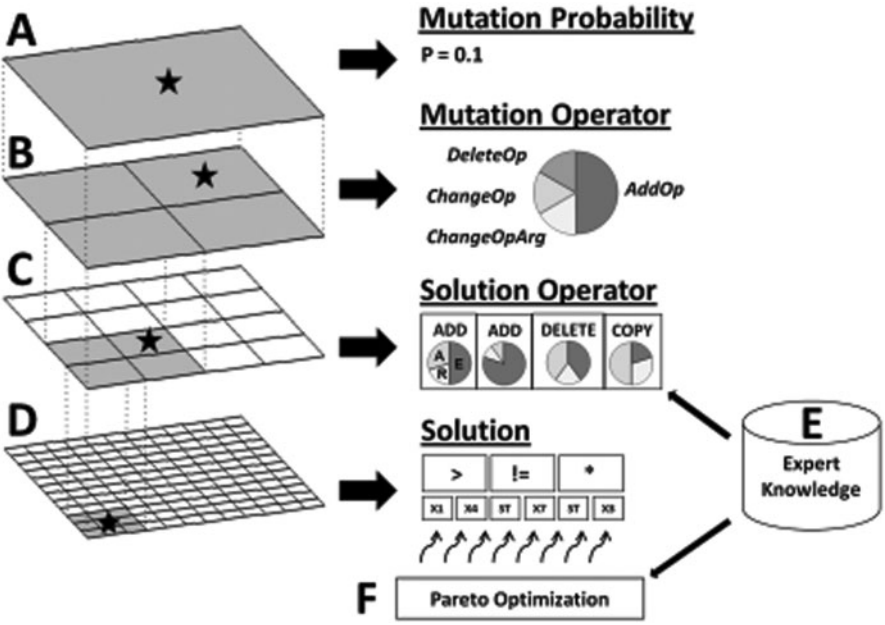

Fig. 2.1 Visual overview of our computational evolution system for discovering symbolic discriminant functions that differentiate disease subjects from healthy subjects using information about single nucleotide polymorphisms (SNPs). The hierarchical structure is shown on the left while some specific examples at each level are shown in the middle. At the lowest level (*D*) is a grid of solutions. Each solution consists of a list of functions and their arguments (e.g. $X1$ is an attribute or SNP) that are evaluated using a stack (denoted by ST in the solution). The next level up (*C*) is a grid of solution operators that each consists of some combination of the ADD, DELETE and COPY functions each with their respective set of probabilities that define whether attributes are added, deleted or copied randomly, using an attribute archive (memory) or just randomly. In this implementation of EMERGENT, we use pre-processed expert knowledge (*E*) with Pareto optimization (*F*) to help reduce overfitting. The top two levels of the hierarchy (*A* and *B*) exist to generate variability in the operators that modify the solutions. This system allows operators of arbitrary complexity to modify solutions. A 12×12 grid is shown here as an example. A 36×36 grid was used in the present study

represented as stacks, where each element in the stack consists of a function and two operands (Fig. 2.1). The function set contains $+, -, *, /, \%, <, <=, >, >=, ==, !=$, where $\%$ denotes protected modulus. Operands are either SNPs, constants, or the output of another element in the stack.

Each solution produces a discrete output S_i when applied to an individual i. Symbolic discriminant analysis (Moore et al. 2002) is then used to map this output to a classification rule, as follows. The solution is independently applied to the set of diseased and healthy individuals to obtain two separate distributions of outputs, $S_{diseased}$ and $S_{healthy}$, respectively. A classification threshold S_0 is then calculated as the arithmetic mean of the medians of these two distributions. Each of the possible relationships between S_0 and S_i($<, <=, >=, >$) is tested across all individuals

and the one with best overall accuracy is chosen to classify whether individuals are healthy or diseased.

Solution accuracy is assessed through a comparison of predicted and actual clinical endpoints. Specifically, the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) are used to calculate accuracy as

$$A = (1/2)(TP/(TP + FN) + TN/(TN + FP)$$

Solution length can be assessed in several ways. The number of elements in the classifier is the most straightforward. Since many solutions leave results on the stack that do not contribute to the classification, we can define "number of relevant elements" as only those contributing to the result. Finally we can count the number of unique features (i.e. genetic variants) in the relevant elements. We have chosen to use this as the measure of length in the present study as it makes the resulting solutions easier to analyze.

The population is organized on a two-dimensional lattice with periodic boundary conditions. Each solution occupies a single lattice site, and competes with the solutions occupying the eight spatially adjacent sites. In all previous EMERGENT implementations election has been both synchronous and elitist, such that the solution of highest fitness within a given neighborhood was always selected to repopulate the focal site of that neighborhood. In the present study, selection proceeds in two stages modeled after Pareto domination tournaments and fitness sharing described by Horn et al. (1994). As described in detail (Moore et al. 2013), we used classification accuracy, the number of features or attributes in the model (i.e. complexity) and interaction information (Moore et al. 2006) as the axes in the Pareto optimization. Here, the sum of the interaction information for all pairs of attributes in a model is the measure of interestingness (described in more detail below). First all dominated solutions and solutions evaluating to a constant are removed from competition. A solution is dominated if there exists any solution that is equal to it or better for all Pareto criteria, and better in at least one criterion. If no solution survives this stage, one of the nine is chosen with equal probability. If more than one solution survives, each is assigned a probability and a roulette wheel selection is made. In order to prevent convergence on solutions of a single length, higher selection probability is assigned to a solution if there are relatively fewer solutions of that length in the lattice. For the present results we made the probability inversely proportional to the square of the number of existing solutions of the same length. Reproduction is either sexual or asexual, as dictated by the evolvable solution operators that reside in the next layer of the hierarchy.

The population is initialized by randomly generating solutions with one to 15 elements subject to the constraint that they produce a valid output that is not constant for all input. The functions are selected at random with uniform probability from the function set. A seed may be specified for the random number generator to make results deterministic and reproducible. Optionally, some of the initial population may be replaced by solutions selected from a database. This selection is also deterministic.

2.3.3 Solution Operators

EMERGENT allows for the evolution of arbitrarily complex variation operators used to modify solutions. This is achieved by initializing the solution operator lattice (Fig. 2.1C) with a set of basic building blocks which can be recombined in any way to form composite operators throughout the execution of the program. The action of some of these operators is influenced by any of several types of expert knowledge (EK) that EMERGENT recognizes. In this study we have used two types of EK, Association EK and Attribute EK. Association EK is used to help EMERGENT to more quickly find solutions using specific combinations of attributes or SNPs. Here, we used a measure of interaction information as the expert knowledge. Adding and altering attributes is based on a lookup table that is constructed from the strength of interactions between pairs of attributes. Because 486,726 attributes have over 1.1×2^{11} pairs, it was impossible to pre-compute and store all pairs in the memory available. We pre-computed all pairs but stored only the 1440 most strongly interacting pairs, a tiny fraction of the total. These are the building blocks and the way they are influenced by Association EK. Pre-computer ReliefF scores were used as Attribute EK. Both of these pre-processing steps are described in additional detail below.

1. ADD: Inserts a randomly generated element into the solution at a randomly selected position.
   a. Attribute EK: The selection of attributes in the new element is biased toward those favored in the Attribute EK file.
   b. Association EK: The existing attribute just upstream of the new element is taken into account. The selection of new attributes is biased toward others in the same cluster.
2. ALTER: Modifies either the function or an argument of a randomly selected element.
   a. Attribute EK: If an attribute argument is chosen, its selection is biased toward those favored in the Attribute EK file.
   b. Association EK: If an attribute argument is chosen, the nearest upstream attribute is taken into account as in the ADD operator.
3. COPY: Within a randomly selected neighboring solution, randomly selects an element and inserts it into a randomly selected position in the focal solution.
   a. Attribute EK: The element chosen to copy has an attribute among the most favored in the Attribute EK file.
   b. Association EK: No effect.
4. DELETE: Removes an element from a randomly selected position.
   a. Attribute EK: The element chosen for deletion has an attribute among the least favored in the Attribute EK file.
   b. Association EK: No effect.
5. REPLACE: Within a randomly selected neighboring solution, randomly selects a source position. In the focal solution, REPLACE randomly selects a destination position. Replaces everything between the destination position and the end (root)

of the focal solution with everything between the source position and the end of the source solution.

a. Attribute EK: The source position is chosen to have an attribute among the most favored in the Attribute EK file.
b. Association EK: No Effect.

The solution operators reside on a periodic, toroidal lattice of coarser granularity than the solution lattice (Fig. 2.1C). Each site is occupied by a single solution operator, which is assigned to operate on a 3×3 sub-grid of solutions. These operators compete with one another in a manner similar to the competition among solutions, and their selection probability is determined by the fitness changes they evoke in the solutions they control. For this purpose we assigned the fitness of a solution as we have done in previous studies: balanced accuracy with a small penalty for number of elements. We did not adapt a Pareto tournament to the selection of solution operators although this would be an interesting extension to explore in a future study.

2.3.4 Mutation Operators

The third level of the hierarchy contains the mutation operators, which are used to modify the solution operators (Fig. 2.1B). These reside on a toroidal lattice of even coarser granularity, and are assigned to modify a subset of the solution operators. The mutation operators are represented as three-element vectors, where each element corresponds to the probability with which a specific mutation operator is used. These three mutation operators work as follows. The first (DeleteOp) deletes an element of a solution operator; the second (AddOp) adds an element to a solution operator, and the third (ChangeOp) mutates an existing element in a solution operator. The probabilities with which these mutation operators are used undergo mutation at a rate specified in the highest level of the hierarchy (Fig. 2.1A).

2.3.5 Primary Open Angle Glaucoma (POAG) Data

The data used in this study came from the Glaucoma Gene Environment Initiative (GLAUGEN) study (Cornelis et al. 2010) that included approximately 1272 subjects with POAG and 1057 healthy controls. A total of 657,366 single-nucleotide polymorphisms (SNPs) were measured across the human genome in these subjects. The data were obtained through application to the dbGaP resource (Mailman et al. 2007) at the National Institutes of Health (accession phs000308.v1.p1). Filtering out the SNPs with very little genetic variation (minor allele frequency < 0.05) or those that were missing 10 % or more of their values left 486,726 SNPs for analysis. Missing values were imputed using a frequency-based method. The goal of the modeling exercise is to identify the optimal subset of SNPs along with the optimal mathematical model that is predictive of the binary class (POAG vs healthy).

2.3.6 Pre-Processing, Experimental Design and Post-Processing

The goal of this study was to apply EMERGENT to the genetic analysis of POAG. We first pre-processed the data by estimating the interaction information for all pairs of SNPs as described previously (Fan et al. 2011; Hu et al. 2011; Moore et al. 2006). We considered pairs of SNPs that have higher interaction information more interesting (Association EK). We also pre-processed the data by running the ReliefF machine learning algorithm as reviewed by Moore et al. (2010). These pre-processed measures of interestingness were used as expert knowledge (Association EK and Attribute EK, respectively) in the CES solution modifiers. Also, the interaction information measure was used as an additional axis in the three-way Pareto optimization.

Each EMERGENT run was conducted with a 36×36 grid of solutions for 2000 generations. The EMERGENT system was implemented in a hierarchical framework inspired by the age-layered population structure algorithm or ALPS (Hornby 2006). Here, we implemented a depth 11 binary tree where each node represents an EMERGENT run with the leaves of the tree representing the initial runs. Each higher node run is initialized with Pareto optimal solutions from the lower nodes. We reported the best models discovered at depths 9, 10 and 11. The interaction information among the top features was used to build a network for statistical interpretation (Hu et al. 2011; Hu et al. 2013).

2.4 Results

The EMERGENT algorithm was run a total of 1024 independent times at the lowest level of the tree shown at the bottom of Fig. 2.2. Symbolic discriminant models of genetic variation from the final Pareto front constructed from sets of two runs were used to seed a new set of 512 runs. This process was repeated according to the tree shown resulting in in a final Pareto optimal set of models at the top of the tree. An advantage of using Pareto optimization is that multiple measures of model quality or interestingness can be used in the model discovery process and for final model selection. We decided to focus on picking final models that had the highest accuracy for classification of POAG, had a moderate number of features (i.e. SNPs) to guard against overfitting and that maximized the consistency of model discovery at levels 9, 10 and 11 of the tree of EMERGENT runs. We found that models with six features were highly consistent across multiple final runs. The top of Fig. 2.2 summarizes simplified versions of the symbolic discriminant models that met our selection criteria. The accuracies of these best models ranged from 0.611 to 0.615. Note that the function $X1 = X6$ and $X8 \; != X18$ showed up in all four best models from level nine and in the best models from the subsequent higher-level runs. In addition, $X3 \; != X16$ and $X12 > X7$ showed up consistently. As shown in Fig. 2.2, each of these two-feature functions formed one of three atomistic modules that came together with an overall function is interpreted as the sum of at least two of the three simplified functions. The consistency of these models each derived from hundreds

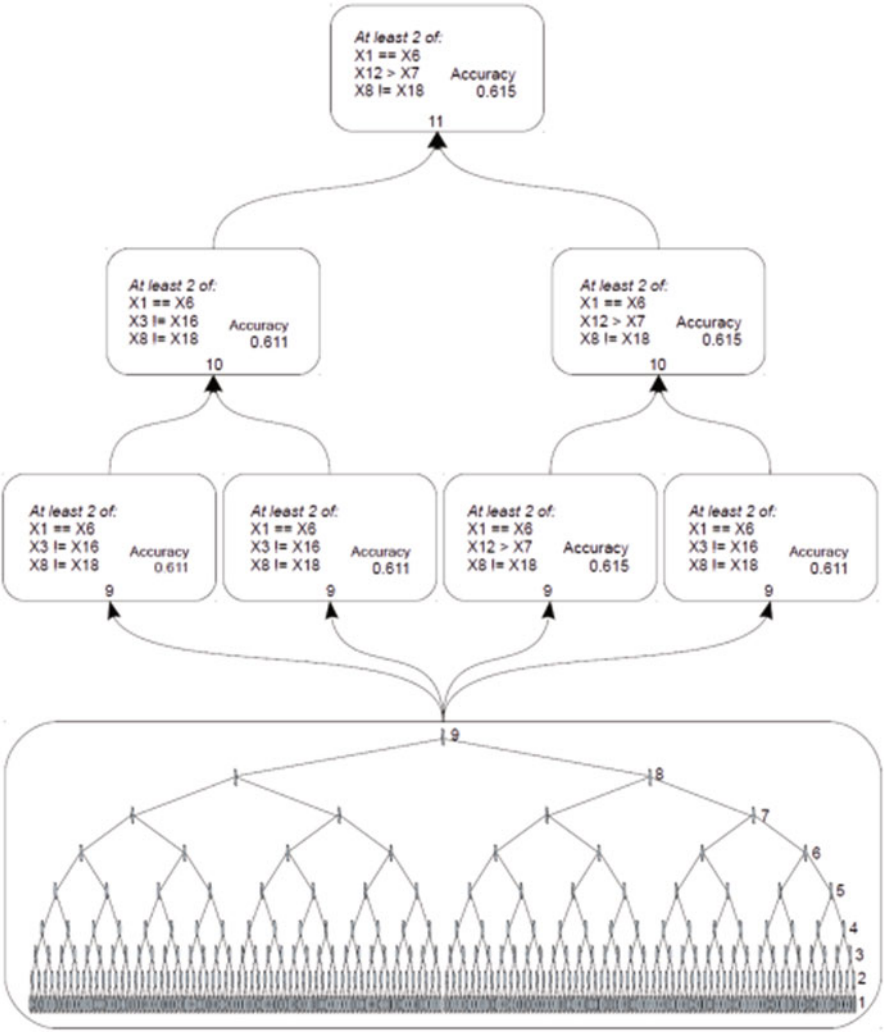

Fig. 2.2 The EMERGENT algorithm was first run 1,024 times with different random seeds (level 1 of the tree). The Pareto optimal results from pairs of runs were used to seed 512 additional independent runs. This hierarchical organization of the runs continued until there were four runs at level 9, two at level 10 and one final run at level 11. Shown are simplified versions of the best models for levels 9, 10 and 11. Note that we chose models with six features from simpler and more complex models along the Pareto front because this model size gave the most consistent results across independent runs

of independent runs with different random seeds was reassuring given the stochastic nature of the EMERGENT algorithm and the effectively infinite search space. Fig. 2.3 shows the version of the final best model prior to simplification. Note that the Boolean result of $X8$!= $X18$ is added to the Boolean result of $X12 > X7$. This result, with

http://www.springer.com/978-3-319-16029-0

Genetic Programming Theory and Practice XII
Riolo, R.; Worzel, B.; Kotanchek, M. (Eds.)
2015, XII, 182 p. 59 illus., 12 illus. in color., Hardcover
ISBN: 978-3-319-16029-0