Chapter 2

ENVIRONMENTAL SENSING OF EXPERT KNOWLEDGE IN A COMPUTATIONAL EVOLUTION SYSTEM FOR COMPLEX PROBLEM SOLVING IN HUMAN GENETICS

Casey S. Greene[1], Douglas P. Hill[1], Jason H. Moore[1]

[1] *Dartmouth College, One Medical Center Drive, HB7937, Lebanon, NH 03756 USA.*

Abstract The relationship between interindividual variation in our genomes and variation in our susceptibility to common diseases is expected to be complex with multiple interacting genetic factors. A central goal of human genetics is to identify which DNA sequence variations predict disease risk in human populations. Our success in this endeavour will depend critically on the development and implementation of computational intelligence methods that are able to embrace, rather than ignore, the complexity of the genotype to phenotype relationship. To this end, we have developed a computational evolution system (CES) to discover genetic models of disease susceptibility involving complex relationships between DNA sequence variations. The CES approach is hierarchically organized and is capable of evolving operators of any arbitrary complexity. The ability to evolve operators distinguishes this approach from artificial evolution approaches using fixed operators such as mutation and recombination. Our previous studies have shown that a CES that can utilize expert knowledge about the problem in evolved operators significantly outperforms a CES unable to use this knowledge. This environmental sensing of external sources of biological or statistical knowledge is important when the search space is both rugged and large as in the genetic analysis of complex diseases. We show here that the CES is also capable of evolving operators which exploit one of several sources of expert knowledge to solve the problem. This is important for both the discovery of highly fit genetic models and because the particular source of expert knowledge used by evolved operators may provide additional information about the problem itself. This study brings us a step closer to a CES that can solve complex problems in human genetics in addition to discovering genetic models of disease.

Keywords: Genetic Epidemiology, Symbolic Discriminant Analysis, Epistasis

R. Riolo et al. (eds.), *Genetic Programming Theory and Practice VII*,
Genetic and Evolutionary Computation, DOI 10.1007/978-1-4419-1626-6_2,

1. Introduction

Computational Challenges in Human Genetics

Human genetics is quickly transitioning away from the study of single genes to evaluating the entire genome. This has been made possible by inexpensive new technologies for measuring 10^6 or more single nucleotide polymorphisms (SNPs) across the genome and emerging technologies that allow us to measure all 3×10^9 nucleotides. As this technological shift occurs, it is critical that the bioinformatics and data analysis approaches for sifting through these large volumes of data keep pace. The development of machine learning and data mining methods that are capable of identifying important patterns of genetic variations that are predictive of disease susceptibility will depend critically on the complexity of the mapping relationship between genotype and phenotype. For common human disease such as breast cancer and schizophrenia this mapping relationship is expected to be very complex with multiple interacting genetic and environmental factors (Moore, 2003; Moore and Williams, 2005; Thornton-Wells et al., 2004).

For the purposes of this paper we will focus exclusively on the SNP, which is a single nucleotide or point in the DNA sequence that differs among people. Most SNPs have two alleles (e.g. `A` or `a`) that combine in the diploid human genome in one of three possible genotypes (e.g. `AA`, `Aa`, `aa`). It is anticipated that at least one SNP occurs approximately every 100 nucleotides across the human genome making it the most common type of genetic variation. Some SNPs will be predictive of disease risk only in the context of other SNPs in the genome (Moore, 2003). This phenomenon has been referred to as epistasis for more than 100 years now (Bateson, 1909) and is the focus of the present study. The general challenge of modeling attribute interactions has been previously described (Freitas, 2001). The question we address is whether a computational evolution system is capable of identifying combinations of interacting SNPs when the fitness landscape is large and rugged. Our results reinforce the idea that expert knowledge is critical to solving these problems.

A Simple Example of the Concept Difficulty

Epistasis or gene-gene interaction can be defined as biological or statistical (Moore and Williams, 2005). Biological epistasis occurs at the cellular level when two or more biomolecules physically interact. In contrast, statistical epistasis occurs at the population level and is characterized by deviation from additivity in a linear mathematical model. Consider the following simple example of statistical epistasis in the form of a penetrance function. Penetrance is simply the probability (P) of disease (D) given a particular combination of genotypes (G) that was inherited (i.e. $P[D|G]$). A single genotype is deter-

Table 2-1. Penetrance values for genotypes from two SNPs.

	AA (0.25)	Aa (0.50)	aa (0.25)
BB (0.25)	0	1	0
Bb (0.50)	1	0	1
bb (0.25)	0	1	0

mined by one allele (i.e. a specific DNA sequence state) inherited from the mother and one allele inherited from the father. For most single nucleotide polymorphisms or SNPs, only two alleles (encoded by `A` or `a`) exist in the biological population. Therefore, because the order of the alleles is unimportant, a genotype can have one of three values: `AA`, `Aa` or `aa`. The model illustrated in Table 2-1 is an extreme example of epistasis. Let's assume that genotypes `AA`, `aa`, `BB`, and `bb` have population frequencies of 0.25 while genotypes `Aa` and `Bb` have frequencies of 0.5 (values in parentheses in Table 2-1). What makes this model interesting is that disease risk is dependent on the particular combination of genotypes inherited. Individuals have a very high risk of disease if they inherit `Aa` or `Bb` but not both (i.e. the Exclusive–OR function). The penetrance for each individual genotype in this model is 0.5 and is computed by summing the products of the genotype frequencies and penetrance values. Thus, in this model there is no difference in disease risk for each single genotype as specified by the single-genotype penetrance values. This model was first described by Li and Reich (Li and Reich, 2000). Heritability, or the size of the genetic effect, is a function of these penetrance values. In this model, the heritability is 1.0, the maximum possible, because the probability of disease is completely determined by the genotypes at these two DNA sequence variations. All the heritability in this model is due to epistasis. As Freitas reviews, this general class of problems has high concept difficulty (Freitas, 2002).

Artificial and Computational Evolution

Numerous machine learning and data mining methods have been developed and applied to the detection of gene-gene interactions in population-based studies of human disease. These include, for example, traditional methods such as neural networks (Lucek and Ott, 1997) and novel methods such as multifactor dimensionality reduction (Ritchie et al., 2001). Evolutionary computing methods such as genetic programming (GP) have been applied to both attribute selection and model discovery in the domain of human genetics. For example, Ritchie et al (Ritchie et al., 2003) used GP to optimize both the weights and the architecture of a neural network for modeling the relationship between genotype and phenotype in the presence of gene-gene interactions. More recently, GP has been successfully used for both attribute selection (Moore and White,

2006a; Moore and White, 2007a; Moore, 2007; Greene et al., 2007) and genetic model discovery (Moore et al., 2007).

Genetic programming is an automated computational discovery tool that is inspired by Darwinian evolution and natural selection (Banzhaf et al., 1998a; Koza, 1992; Koza, 1994; Koza et al., 1999; Koza et al., 2003; Langdon, 1998; Langdon and Poli, 2002). The goal of GP is to evolve computer programs to solve problems. This is accomplished by first generating random computer programs composed of the building blocks needed to solve or approximate a solution. Each randomly generated program is evaluated and the good programs are selected and recombined to form new computer programs. This process of selection based on fitness and recombination to generate variability is repeated until a best program or set of programs is identified.

Genetic programming and its many variations have been applied successfully to a wide range of different problems including data mining and knowledge discovery (e.g. (Freitas, 2002)) and bioinformatics (e.g. (Fogel and Corne, 2003)). Despite the many successes, there are a large number of challenges that GP practitioners and theorists must address before this general computational discovery tool becomes one of several tools that a modern problem solver calls upon (Yu et al., 2006). Spector, as part of an essay regarding the roles of theory and practice in genetic programming, discusses the push towards biology by GP practitioners (Spector, 2003). Banzhaf et al. propose that overly simplistic and abstracted artificial evolution (AE) methods such as GP need to be transformed into computational evolution (CE) systems that more closely resemble the complexity of real biological and evolutionary systems (Banzhaf et al., 2006). Evolution by natural selection solves problems by building complexity. We are thus interested in testing the working hypothesis that a GP-based genetic analysis system will find better solutions faster if it is implemented as a CE system that can evolve a variety of complex operators that in turn generate variability in solutions. This is in contrast to an AE system that uses a fixed set of operators.

Research Questions Addressed and Overview

We have previously developed a prototype CE system and have shown that it is capable of evolving complex operators for problem solving in human genetics (Moore et al., 2008b). We have also previously extended and evaluated this new open-ended computational evolution system for the detection and characterization of epistasis or gene-gene interactions that are associated with risk of human disease (Moore et al., 2008a). New features in this previous study included simpler operator building blocks, list-based solutions with stack-based evaluation and an attribute archive that provides the system with a feedback loop between the population of solutions and the solution operators. These re-

cently added features are consistent with the idea of transforming an AE system to a CE system. This study showed that a CE system that could exploit expert knowledge performed better than a system that could not. This provides the basis for the present study that addresses the question of whether the CE system is capable of identifying and exploiting a good source of expert knowledge from among several other randomly generated sources.

2. A Computational Evolution System

Our primary goal was to develop, extend and evaluate a computational evolution system that is capable of open-ended evolution for bioinformatics problem-solving in the domain of human genetics. Figure 2-1 gives a graphical overview of our hierarchically-organized and spatially-extended GP system that is capable of open-ended computational evolution. At the bottom layer of this hierarchy is a grid of solutions. At the second layer of the hierarchy is a grid of operators of any size and complexity that are capable of modifying the solutions (i.e. solution operators). At the third layer is a grid of mutation operators that are capable of modifying the solution operators. At the highest level of the hierarchy is the mutation frequency that determines the rate at which operators are mutated. An attribute archive provides a feedback loop between the solutions and the solution operators. One or more sources of expert knowledge is also provided to the system for environmental sensing. The details of the experimental design used to evaluate this system are described in Section 3.

Problem Solutions: Their Representation, Fitness Evaluation and Reproduction

The goal of a classifier is to accept as input two or more discrete attributes (i.e. SNPs) and produce a discrete output that can be used to assign class (i.e. healthy or sick). Here, we used symbolic discriminant analysis or SDA as our classifier. The SDA method (Moore et al., 2002) has been described previously for this problem domain (Moore et al., 2008b; Moore et al., 2007; Moore and White, 2007a). SDA models consist of a set of attributes and constants as input and a set of mathematical functions that produce for each instance in the data set a score called a symbolic discriminant score. Here, our SDA function set was $+, -, *, /, \%, <, <=, >, >=, ==, != $ where the $\%$ operator is a mod operation and $/$ is a protected division. The SDA models are represented as a list of expressions here instead of as expression trees as has been used in the past to facilitate stack-based evaluation of the classifiers and to facilitate their representation in text files. This is similar to the GP implementation using arrays and stack as described by Keith and Martin (Keith and Martin, 1994), Perkis (Perkis, 1994), and Banzaf et al. (Banzhaf et al., 1998b).

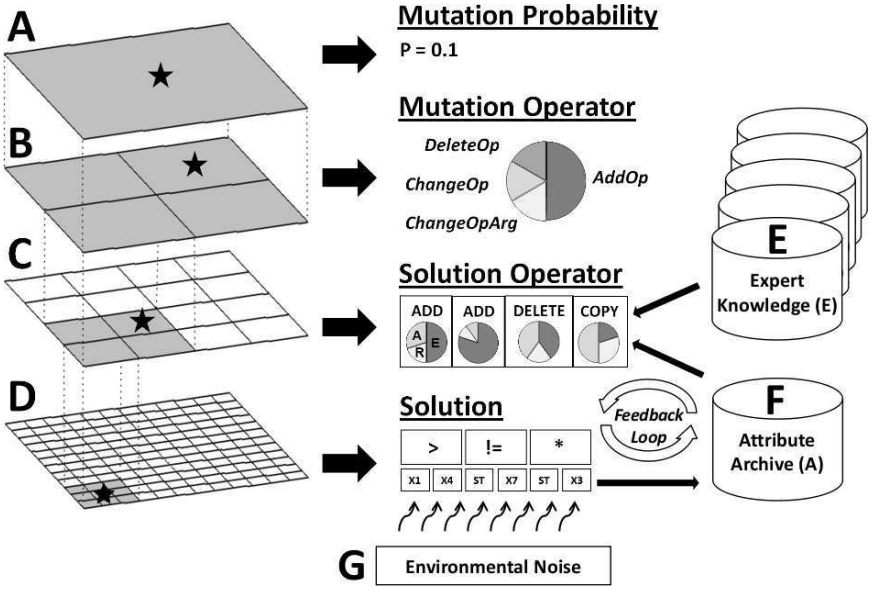

Figure 2-1. Visual overview of our computational evolution system for discovering symbolic discriminant functions that differentiate disease subjects from healthy subjects using information about single nucleotide polymorphisms (SNPs). The hierarchical structure is shown on the left while some specific 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) 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 expert knowledge (E) or the archive (F, denoted by A in the probability pie) are used instead of a random generator (denoted by R in the probability pie). The attribute archive (F) is derived from the frequency with which each attribute occurs among solutions in the population. Finally, environmental noise (G) perturbs the data in small ways to prevent over fitting. 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. Note that we used 18×18 grids of 324 solutions in the present study. A 12×12 grid is shown here as an example.

Classification of instances into one of the two classes requires a decision rule that is based on the symbolic discriminant score. Thus, for any given symbolic discriminant score (S_{ij}) in the ith class and for the j^{th} instance, a decision rule can be formed such that if $S_{ij} > S_o$ then assign the instance to one class and if $S_{ij} <= S_o$ then assign the observation to the other class. When the prior probability that an instance belongs to one class is equal to the probability that it belongs to the other class, S_o can be defined as the arithmetic mean of the median symbolic discriminant scores from each of the two classes. This is the classification rule we used in the present study and is consistent with

previous work in this domain (Moore et al., 2008b; Moore et al., 2007; Moore and White, 2007a). Using this decision rule, the classification accuracy for a particular discriminant function can be estimated from the observed data. Here, accuracy is defined as (TP + TN)/(TP + TN + FP + FN) where TP are true positives (TP), TN are true negatives, FP are false positives, and FN are false negatives. We used accuracy as the fitness measure for SDA solutions as has been described previously but lightly weight it such that for solutions with equivalent accuracy, ones with shorter genome sizes are preferable (Moore et al., 2008b; Moore et al., 2007; Moore and White, 2007a).

All SDA solutions in a population are organized on a toroidal grid with specific X and Y coordinates (see example in Figure 2-1). As such, they resemble previous work on cellular genetic programming (Folino et al., 1999). In the present study we used a grid size of 18×18. Reproduction of solutions in the population is handled in a spatial manner. Each solution is considered for reproduction in the context of its Moore neighborhood using an elitist strategy. That is, each solution in question will compete with its eight neighbors and be replaced in the next generation by the neighbor with the highest fitness. This combines ideas of tournament selection that is common in GP with a set of solutions on a grid. Variability in solutions is generated using hierarchically organized operators. This is described below.

Operators for Computational Evolution: Generating Solution Variability

Traditional artificial evolution approaches such as GP use a fixed set of operators that include mutation and recombination, for example. The goal of developing a computational evolution system was to provide building blocks (i.e. simple functions) for operators that could be combined to create new operators. We started with the following three basic operator building blocks. The first operator building block, ADD, adds a new function and its arguments to the list of functions and arguments that comprise a solution. The second operator building block, DELETE, deletes a function from the list of functions. The third operator, COPY, copies one function from the list of functions in the Moore neighborhood. These operators can combine in any number and order to generate solution operators of arbitrary complexity. The mutation operators described below increase or decrease the size and content of the solution operators.

Each of the operator building blocks has a vector of three probabilities associated with it. The first number specifies the probability that the function that is added, deleted or copied to a solution is determined stochastically. The second specifies the probability that the function that is added, deleted or copied to a solution is determined according to an archive of attributes that is ranked

according to the frequency that they occur in the population of solutions (see below). The third specifies the probability that the function that is added, deleted or copied to a solution is determined according to ReliefF scores for the attributes (see below). The ability to use expert knowledge (i.e. environmental sensing) is important in this domain. For example, pre-processed ReliefF scores have been shown to improve the performance of GP as a wrapper in this domain when used in a multiobjective fitness function (Moore and White, 2007a), when used to guide recombination (Moore and White, 2006a) and when used to guide mutation (Greene et al., 2007). This is consistent with Goldberg's ideas about exploiting good building blocks in competent genetic algorithms (Goldberg, 2002) and provides a source of complexity as recommended by Banzhaf et al. (Banzhaf et al., 2006). For example, the use of the archive creates a feedback loop between the solutions and the solution operators. In the present study we evaluated whether this system is able to identify a good source of expert knowledge from among five candidates. Here, each building block had six probabilities associated with it, one for each of the five sources of expert knowledge and one for the stochastic element. We did not use the archive in this study given the focus was on understanding the role of multiple sources of expert knowledge.

As with the solutions, each operator is organized on a toroidal grid with a specific X and Y coordinate. We assigned each operator to a set of solutions. This allows for averaging an operator's positive or negative effects on multiple solutions. In this study, we assigned each operator to a 3×3 grid of nine solutions. Thus, the population of solution operators is organized in a 6×6 grid when an 18×18 grid is used for the solutions and 12×12 when a 36×36 grid is used for the solutions. The assignment of fitness to solution operators is a variant of Edmond's Meta-GP framework (Edmonds, 1998; Edmonds, 2001). To assign fitness to an operator, we first identify the two solutions under the operator's control that show the most positive change in fitness, on the basis that an operator is more fit if it greatly increases fitness in a few solutions, even if it reduces fitness in many cases. We average these changes in fitness and this becomes the fitness of the operator. If the operator has not been modified in this generation, we smooth its fitness by adding half of the previous generation's fitness and multiplying by two thirds, so the fitness scale is comparable between new and unchanged operators.

Mutation of Operators for Computational Evolution: Generating Operator Variability

An important goal for the computational evolution system is the ability to generate variability in the operators that modify solutions. To accomplish this goal we previously developed an additional level in the hierarchy (Figure 2-1)

with mutation operators that specifically alter the operators described above. We defined four different fixed mutation operators that are each assigned to a 2×2 grid of solution operators. Solution operators can be modified in the following four ways. First, an operator can have a specific operator building block deleted (DeleteOperator). Second, an operator can have a specific operator building block added (AddOperator). Third, an operator can have a specific operator building block changed (ChangeOperator). Finally, an operator can have its probabilities changed (ChangeOperatorArguments). In this study, we initialized the probabilities with which each the these mutation operators are used to 0.25. These are randomly regenerated at a frequency equal to the overall mutation probability (see below) and their fitness is determined by the change in fitness of the solution operators that they act on.

Mutation Frequency

The top level of the computational evolution system hierarchy (see Figure 2-1) is the mutation frequency that controls the probability that one of the four mutation sets in the next level down will mutate a given solution operator two levels down. In the present study we fixed this to 0.1. In the future this will be an evolvable parameter. This frequency does not control the frequency with which a solution operator modifies a solution. That is controlled by the operator when it specifies which solution(s) it will modify.

Environmental Sensing Using an Archive

Previous studies have demonstrated the utility of archiving GP results for reuse (Vladislavleva et al., 2007). We have previously implemented an archive that ranks the attributes by the frequency with which they appear in solutions from the population. These are ranked by their frequency and then used by the ADD, DELETE and COPY operators to decide what gets added, deleted or copied. We have previously used a cumulative archive that updates the previous results each generation. The archive is an important part of the complexity of the CE system because it provides a feedback loop between the solutions and the solution operators. The archive was not used in the present study to allow us to focus on the use of multiple source of expert knowledge.

Environmental Sensing Using Expert Knowledge

As mentioned above, the use of expert knowledge is important for the application of GP strategies to solving complex problems in human genetics. Here, we used pre-processed ReliefF scores for all of the attributes in the dataset as a source of statistical knowledge for the analysis. Kira and Rendell developed the Relief algorithm that is capable of detecting attribute dependencies (Kira and Rendell, 1992). Relief estimates the quality of attributes through a type

of nearest neighbor algorithm that selects neighbors (instances) from the same class and from the different class based on the vector of values across attributes. Weights (W) or quality estimates for each attribute (A) are estimated based on whether the nearest neighbor (nearest hit, H) of a randomly selected instance (R) from the same class and the nearest neighbor from the other class (nearest miss, M) have the same or different values. This process of adjusting weights is repeated for m instances. The algorithm produces weights for each attribute ranging from -1 (worst) to +1 (best). Kononenko improved upon Relief by choosing n nearest neighbors instead of just one (Kononenko, 1994). This new ReliefF algorithm has been shown to be more robust to noisy attributes and is widely used in data mining applications. We have developed a modified ReliefF algorithm for the domain of human genetics called Tuned ReliefF (TuRF). We have previously shown that TuRF is significantly better than ReliefF in this domain (Moore and White, 2007b). The TuRF algorithm systematically removes attributes that have low quality estimates so that the ReliefF values if the remaining attributes can be re-estimated. We applied TuRF as described by Moore and White (Moore and White, 2007b) to the data set analyzed and provided the results to the CE system as expert knowledge that can then used by the ADD, DELETE and COPY operators to decide what gets added, deleted or copied (Moore et al., 2008a). We also provided four random permutations of the TuRF knowledge as additional null sources of knowledge to assess whether the CE system could identify and exploit the correct source.

Implementation

The computational evolution system described above was programmed entirely in C++. A single run of the system with a population of 324 solutions on a 18×18 grid for 1000 generations took approximately 15 minutes on a 3.0 GHz AMD Opteron processor. Multiple runs for the experiments described below were carried out in parallel using 100 or more processors.

3. Experimental Design and Data Analysis

Our goal was to provide an evaluation of the CE system described above using a repeated measures experimental design. The central question addressed in this study is whether the CE system has the ability to identify and exploit the correct source of expert knowledge out of a total of five. Here, the probability of a given operator such as ADD using any given source of knowledge is initialized randomly for the first generation. The probability associated with each source of knowledge can change over time based on its fitness reward that is assessed by the fitness change in the solutions that operator operates on.

Here, we ran the CE system for a total of 1000 generations with a solution grid size of 18×18. A total of 100 runs each with different random seeds

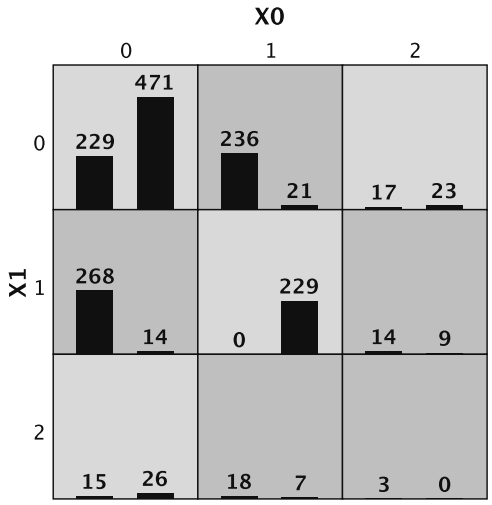

Figure 2-2. Distribution of healthy controls (left bars) and diseased cases (right bars) for each combination of genotypes (coded 0, 1 and 2) for the two functional attributes or SNPs (X0 and X1). Note the nonlinear pattern of high-risk (dark grey) and low-risk (light grey) genotype combinations that is indicative of a nonlinear interaction.

were performed on the simulated data described below. For each of 100 runs we recorded the average probability for each source of knowledge at generation zero and 1000. We used a repeated measures analysis of variance (RMANOVA) to test three hypotheses about the results. First, we tested the null hypothesis that the mean probabilities are the same for each source of expert knowledge (i.e. the treatment effect). Second, we tested the null hypothesis that the vector or profile of mean probabilities across generations zero and 1000 are flat for each source of expert knowledge (i.e. the time effect). Third, we tested the null hypothesis that the mean probabilities don't change across generations in a manner that is dependent on the particular source of knowledge (i.e. treatment by time interaction). Treatment, time and treatment by time effects were considered statistically significant at the 0.05 level. Following the RMANOVA analysis we performed a post-hoc analysis of the time effect within each treatment using a paired t-test. Specifically, we tested the null hypothesis that difference in means between generation zero and generation 1000 is zero within each source of expert knowledge or the random element. Specific contrasts were considered statistically significant at the 0.008 level. The is a Bonferroni-corrected level of significance that accounts for the multiple statistical tests that were performed across contrasts.

We used a simulated data set consisting of 1000 total attributes (SNPs) and 1600 instances (800 cases and 800 controls). Two of the 1000 SNPs are associated with disease class through a nonlinear interaction as described in the

introduction. This dataset has been previously described (Velez et al., 2007). Figure 2-2 illustrates the distribution of healthy controls (left bars) and diseased cases (right bars) for each combination of genotypes (coded 0, 1 and 2) for the three functional attributes or SNPs (X0, X1). Note the nonlinear pattern of high-risk (dark grey) and low-risk (light grey) genotype combinations. The optimal classification of this dataset yields a classification accuracy of approximately 0.8. This is the fitness target.

4. Results

Figure 2-3 summarizes the mean probabilities for selecting attributes for each source of expert knowledge and random for generation zero and 1000. The RMANOVA analysis showed a highly significant difference in mean probabilities between the treatment groups independent of time ($P < 0.001$). Figure 2-3 shows that the mean probability for the correct source of expert knowledge is higher than the others. We also found no overall time or generation effect independent of knowledge source ($P > 0.1$). This is consistent with what we see in Figure 2-3. On average there is no generation effect. Finally, the RMANOVA indicated a highly significant source of knowledge by generation interaction ($P < 0.001$). Figure 2-3 illustrates this very clearly with the mean

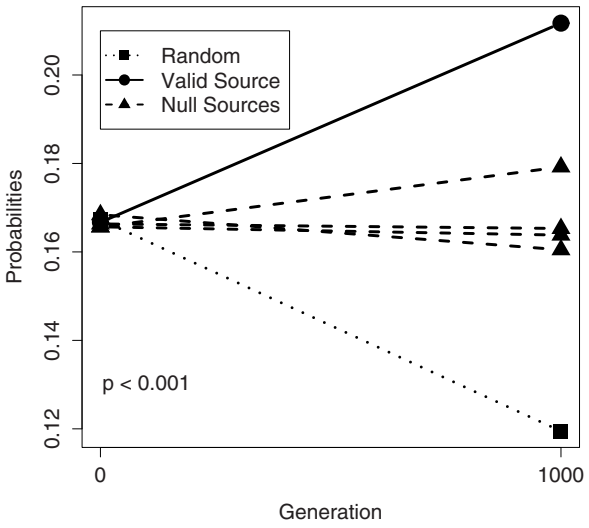

Figure 2-3. The mean probabilities of operators using expert knowledge increases from the beginning to the end of the run. The probabilities of operators acting randomly decreases. The probabilities of the null sources do not significantly change.

probability increasing from generation zero to 1000 for the correct source of expert knowledge while staying the same for other null sources of knowledge and decreasing with the random element. The different slopes of these lines accounts for the statistically significant interaction. We used a paired t-test with correction for multiple testing to carry out a post-hoc analysis to verify that the probabilities for the correct source of expert knowledge do in fact increase. We found that statistically significant evidence to reject the null hypothesis that the difference in mean probabilities for generation zero and 1000 are zero ($P \leq 0.001$). This same null hypothesis for each of the null sources of knowledge were not rejected ($P > 0.1$). Interestingly, the probabilities for the random element significantly decrease ($P < 0.001$). These results provide significant evidence in support of our working hypothesis that the CE system is capable of identifying and exploiting an important source of expert knowledge in the context of multiple other null sources.

Figure 2-4 illustrates the results from a single run of the CE system for 1000 generations. Plotted in this figure is the maximum fitness (classifier accuracy) for each generation. Note the first major increase in fitness is associated with the best model obtaining the correct two attributes while the second major increase

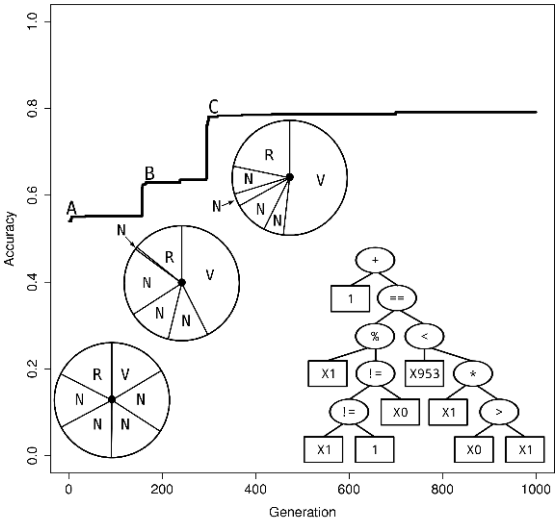

Figure 2-4. The line shows the fitness of the best solution from generation 1 to 1000. The pies, in ascending order, show the average probabilities associated with the different sources of expert knowledge at the initial generation (Point A) as well as generations 165 (Point B) and 310 (Point C). The probabilities are shown for Random (R), the four null sources (N) and the valid source of knowledge (V). The tree representation of the final solution is shown in the bottom right.

is associated with obtaining a set of functions that approximates the optimal solution to the problem. Also shown in Figure 2-4 are the average probabilities tied to the operator that operated on the best solution. Note that in generation zero the probabilities start out approximately equal. By the first increase in fitness the probability of picking attributes based on the good source of expert knowledge has increased to approximately 0.5. This illustrates the ability of the CE system to identify and exploit a particular source of knowledge. The solution shown in Figure 2-4 illustrates an example solution to the problem that was discovered by the CE system. The CE system was able to discover an optimal or near optimal solution to this problem in each of the 100 runs.

5. Discussion and Conclusions

Human genetics is transitioning away from the study of single genes to the study of the entire genome as a risk factor for common human diseases (Moore, 2009). This means we need to prepare the next generation of computational intelligence approaches that are able to model multiple interacting genetic risk factors simultaneously in data derived from large epidemiological and genetic studies. We present here a computational evolution (CE) approach to this problem that builds on the successes and failures of artificial evolution (e.g. genetic programming) to provide a comprehensive framework for genetic analysis. We have previously shown that adding complexity to these algorithms improves their ability to identify complex genetic models (Moore et al., 2008b; Moore et al., 2008a). This is consistent with our previous work showing how expert knowledge can greatly improve the performance of these algorithms (Moore and White, 2007a; Moore and White, 2006b; Greene et al., 2007). The goal of this study was to determine whether a CE system could learn to recognize and exploit a good source of expert knowledge from among several different options. Our results demonstrate that the CE system does indeed learn to use a valid source of expert knowledge to discover optimal solutions in this domain.

The ability of the system to identify and exploit a particular source of expert knowledge to solve a complex problem is important. However, equally exciting is the possibility of inferring from the behavior of the evolved system what source(s) of expert knowledge seems to be important. The results summarized in Figure 2-3 show the change in probabilities for each source of knowledge shift from being approximately equal to favoring one particular source of knowledge. This is important because that source of knowledge may tell us something about the problem itself. For example, let's assume that each source of knowledge was biological in nature representing perhaps biochemical pathways, gene ontology, chromosomal location, protein-protein interactions and prior knowledge derived from microarray experiments. Preferential use of microarray knowledge may tell us that the DNA sequence variations in the

best model might have something to do with gene expression. This in turn provide an important basis for interpreting the model and understanding why it is important. One ultimate goal of these studies is to understand why particular genetic factors increase or decrease risk. A biological understanding may play an important role in developing interventions and treatments for the disease. The present study opens the door to using multiple sources of biological and statistical knowledge for solving real world genetic analysis problems.

An important future goal will be to explore how multiple sources of knowledge might be used together. Could the CE system learn to use two or three sources of knowledge that each provide complementary information? How will we need to modify the operators to effectively use joint information? How will the sources of expert knowledge interact with the archive? This last point will be particularly interesting to explore. We turned the archive off in the present study so as not to confound the question being addressed about multiple sources of expert knowledge. However, a logical next step will be to turn this back on to determine whether there is a benefit to having both working together. It is reasonable to assume that the expert knowledge will be important early in the process when it is important to find the functional attributes. Once they are found and rewarded these important building blocks will spread throughout the population and then become part of the archive. The relative weighting of the attributes in the archive could be greater than that provided by the expert knowledge. If this is the case, one might predict that archive would take over and become more important than the source of expert knowledge. These are all interesting new directions to pursue. These questions and others will need to be addressed before this system is ready for the routine analysis of real data.

Acknowledgment

This work was supported by National Institutes of Health (USA) grants LM009012 and AI59694. We thank the attendees of the 2008 Genetic Programming Theory and Practice (GPTP) Workshop for their insightful ideas about computational evolution.

References

Banzhaf, W., Beslon, G., Christensen, S., Foster, J. A., Kepes, F., Lefort, V., Miller, J., Radman, M., and Ramsden, J. J. (2006). From artificial evolution to computational evolution: a research agenda. *Nature Reviews Genetics*, 7:729–735.

Banzhaf, Wolfgang, Nordin, Peter, Keller, Robert E., and Francone, Frank D. (1998a). *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, San Francisco, CA, USA.

Banzhaf, Wolfgang, Poli, Riccardo, Schoenauer, Marc, and Fogarty, Terence C., editors (1998b). *Genetic Programming*, volume 1391 of *LNCS*, Paris. Springer-Verlag.

Bateson, W. (1909). *Mendel's Principles of Heredity*. Cambridge University Press, Cambridge.

Edmonds, Bruce (1998). Meta-genetic programming: Co-evolving the operators of variation. CPM Report 98-32, Centre for Policy Modelling, Manchester Metropolitan University, UK, Aytoun St., Manchester, M1 3GH. UK.

Edmonds, Bruce (2001). Meta-genetic programming: Co-evolving the operators of variation. *Elektrik*, 9(1):13–29. Turkish Journal Electrical Engineering and Computer Sciences.

Fogel, G.B. and Corne, D.W. (2003). *Evolutionary Computation in Bioinformatics*. Morgan Kaufmann Publishers.

Folino, Gianluigi, Pizzuti, Clara, and Spezzano, Giandomenico (1999). A cellular genetic programming approach to classification. In Banzhaf, Wolfgang, Daida, Jason, Eiben, Agoston E., Garzon, Max H., Honavar, Vasant, Jakiela, Mark, and Smith, Robert E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1015–1020, Orlando, Florida, USA. Morgan Kaufmann.

Freitas, A. (2001). Understanding the crucial role of attribute interactions. *Artificial Intelligence Review*, 16:177–199.

Freitas, A. (2002). *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer.

Goldberg, D. E. (2002). *The Design of Innovation*. Kluwer.

Greene, C. S., White, B. C., and Moore, J. H. (2007). An expert knowledge-guided mutation operator for genome-wide genetic analysis using genetic programming. *Lecture Notes in Bioinformatics*, 4774:30–40.

Keith, M. J. and Martin, M. C. (1994). *Advances in Genetic Programming*. MIT Press.

Kira, K. and Rendell, L. A. (1992). A practical approach to feature selection. *In: Machine Learning: Proceedings of the AAAI'92*.

Kononenko, I. (1994). Estimating attributes: Analysis and extension of relief. *Machine Learning: ECML-94*, pages 171–182.

Koza, John R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA.

Koza, John R. (1994). *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge Massachusetts.

Koza, John R., Andre, David, Bennett III, Forrest H, and Keane, Martin (1999). *Genetic Programming 3: Darwinian Invention and Problem Solving*. Morgan Kaufman.

Koza, John R., Keane, Martin A., Streeter, Matthew J., Mydlowec, William, Yu, Jessen, and Lanza, Guido (2003). *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Kluwer Academic Publishers.

Langdon, W. B. and Poli, Riccardo (2002). *Foundations of Genetic Programming*. Springer-Verlag.

Langdon, William B. (1998). *Genetic Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming!*, volume 1 of *Genetic Programming*. Kluwer, Boston.

Li, W. and Reich, J. (2000). A complete enumeration and classification of two-locus disease models. *Human Heredity*, 50:334–49.

Lucek, P.R. and Ott, J. (1997). Neural network analysis of complex traits. *Genetic Epidemiology*, 14(6):1101–1106.

Moore, J. H. (2003). The ubiquitous nature of epistasis in determining susceptibility to common human diseases. *Human Heredity*, 56:73–82.

Moore, J. H. (2007). Genome-wide analysis of epistasis using multifactor dimensionality reduction: feature selection and construction in the domain of human genetics. In *Knowledge Discovery and Data Mining: Challenges and Realities with Real World Data*. IGI.

Moore, J. H. and White, B. C. (2006a). Exploiting expert knowledge in genetic programming for genome-wide genetic analysis. *Lecture Notes in Computer Science*, 4193:969–977.

Moore, J. H. and White, B. C. (2007a). Genome-wide genetic analysis using genetic programming: The critical need for expert knowledge. In Riolo, Rick L., Soule, Terence, and Worzel, Bill, editors, *Genetic Programming Theory and Practice IV*, Genetic and Evolutionary Computation. Springer.

Moore, J. H. and White, B. C. (2007b). Tuning relieff for genome-wide genetic analysis. *Lecture Notes in Computer Science*, 4447:166–175.

Moore, J. H. and Williams, S. W. (2005). Traversing the conceptual divide between biological and statistical epistasis: Systems biology and a more modern synthesis. *BioEssays*, 27:637–46.

Moore, Jason H., Greene, Casey S., Andrews, Peter C., and White, Bill C. (2008a). Does complexity matter? artificial evolution, computational evolution and the genetic analysis of epistasis in common human diseases. In Riolo, Rick L., Soule, Terence, and Worzel, Bill, editors, *Genetic Programming Theory and Practice VI*, Genetic and Evolutionary Computation, chapter 9, pages 125–145. Springer, Ann Arbor.

Moore, Jason H. and White, Bill C. (2006b). Exploiting expert knowledge in genetic programming for genome-wide genetic analysis. In Runarsson, Thomas Philip, Beyer, Hans-Georg, Burke, Edmund, Merelo-Guervos, Juan J., Whitley, L. Darrell, and Yao, Xin, editors, *Parallel Problem Solving from Nature - PPSN IX*, volume 4193 of *LNCS*, pages 969–977, Reykjavik, Iceland. Springer-Verlag.

Moore, J.H. (2009). From genotypes to genometypes: putting the genome back in genome-wide association studies. *Eur J Hum Genet*.

Moore, J.H., Andrews, P.C., Barney, N., and White, B.C. (2008b). Development and evaluation of an open-ended computational evolution system for the genetic analysis of susceptibility to common human diseases. *Lecture Notes in Computer Science*, 4973:129–140.

Moore, J.H, Barney, N., Tsai, C.T, Chiang, F.T, Gui, J., and White, B.C (2007). Symbolic modeling of epistasis. *Human Heridity*, 63(2):120–133.

Moore, J.H, Parker, J.S., Olsen, N.J, and Aune, T. (2002). Symbolic discriminant analysis of microarray data in autoimmune disease. *Genetic Epidemiology*, 23:57–69.

Perkis, Tim (1994). Stack-based genetic programming. In *Proceedings of the 1994 IEEE World Congress on Computational Intelligence*, volume 1, pages 148–153, Orlando, Florida, USA. IEEE Press.

Ritchie, M. D., Hahn, L. W., and Moore, J. H. (2003). Power of multifactor dimensionality reduction for detecting gene-gene interactions in the presence of genotyping error, phenocopy, and genetic heterogeneity. *Genetic Epidemiology*, 24:150–157.

Ritchie, M. D., Hahn, L. W., Roodi, N., Bailey, L. R., Dupont, W. D., Parl, F. F., and Moore, J. H. (2001). Multifactor dimensionality reduction reveals high-order interactions among estrogen metabolism genes in sporadic breast cancer. *American Journal of Human Genetics*, 69:138–147.

Spector, Lee (2003). An essay concerning human understanding of genetic programming. In Riolo, Rick L. and Worzel, Bill, editors, *Genetic Programming Theory and Practice*, chapter 2, pages 11–24. Kluwer.

Thornton-Wells, T. A., Moore, J. H., and Haines, J. L. (2004). Genetics, statistics and human disease: Analytical retooling for complexity. *Trends in Genetics*, 20:640–7.

Velez, D.R., White, B.C., Motsinger, A.A., Bush, W.S., Ritchie, M.D., Williams, S.M., and Moore, J.H. (2007). A balanced accuracy function for epistasis modeling in imbalanced datasets using multifactor dimensionality reduction. *Genetic Epidemiology*, 31(4).

Vladislavleva, Ekaterina, Smits, Guido, and Kotanchek, Mark (2007). Soft evolution of robust regression models. In Riolo, Rick L., Soule, Terence, and Worzel, Bill, editors, *Genetic Programming Theory and Practice V*, Genetic and Evolutionary Computation, chapter 2, pages 13–32. Springer, Ann Arbor.

Yu, T., Riolo, R., and Worzel, B. (Eds.) (2006). *Genetic Programming Theory and Practice III*. Springer.