An Expert Knowledge-Guided Mutation Operator for Genome-Wide Genetic Analysis Using Genetic Programming

Casey S. Greene[1], Bill C. White[1], and Jason H. Moore[1]

Dartmouth College, Hanover, NH 03755, USA
{Casey.S.Greene,Bill.C.White,Jason.H.Moore}@dartmouth.edu
http://www.epistasis.org

Abstract. Human genetics is undergoing a data explosion. Methods are available to measure DNA sequence variation throughout the human genome. Given current knowledge it seems likely that common human diseases are best predicted by interactions between biological components, which can be examined as interacting DNA sequence variations. The challenge is thus to examine these high-dimensional datasets to identify combinations of variations likely to predict common diseases. The goal of this paper was to develop and evaluate a genetic programming (GP) mutator suited to this task by exploiting expert knowledge in the form of Tuned ReliefF (TuRF) scores during mutation. We show that using expert knowledge guided mutation performs similarly to expert knowledge guided selection. This study demonstrates that in the context of an expert knowledge aware GP, mutation may be an appropriate component of the GP used to search for interacting predictors in this domain.

1 Introduction

Biological and biomedical sciences are undergoing a data explosion without a corresponding knowledge explosion. This is especially true in the domain of human genetics where it is now technically and economically feasible to measure thousands of DNA sequence variations from across the human genome. For the purposes of this paper we will focus exclusively on the single nucleotide polymorphism or SNP which is a single nucleotide or point in the DNA sequence that differs among people. It is anticipated that at least one SNP occurs approximately every 100 nucleotides across the $3x10^9$ nucleotide human genome. An important goal in human genetics is to determine which of the many thousands of SNPs are useful for predicting who is at risk for common diseases. This "genome-wide" approach is expected to revolutionize the genetic analysis of common human diseases. The charge for computer science and bioinformatics is to develop algorithms for the detection and characterization of those SNPs that are predictive of human health and disease. Success in this endeavor will be difficult due to nonlinearity in the genotype-to-phenotype mapping relationship that is due, in part, to epistasis or nonadditive gene-gene interactions. The

J.C. Rajapakse, B. Schmidt, and G. Volkert (Eds.): PRIB 2007, LNBI 4774, pp. 30–40, 2007.
© Springer-Verlag Berlin Heidelberg 2007

implication of epistasis from a data mining point of view is that SNPs need to be considered jointly in learning algorithms rather than individually. The challenge of modeling attribute interactions has been previously described [1]. Due to the combinatorial magnitude of this problem, intelligent analysis strategies are needed.

1.1 Concept Difficulty

Combining the difficulty of modeling nonlinear attribute interactions with the challenge of attribute selection yields for this domain what Goldberg [2] calls a needle-in-a-haystack problem. That is, there may be a particular combination of SNPs that together with the right nonlinear function are a significant predictor of disease susceptibility. Considered individually they may not look any different than thousands of other noisy SNPs not involved in the disease process. Under these models, the learning algorithm is truly looking for a genetic needle in a genomic haystack. A recent report from the International HapMap Consortium [3] suggests that approximately 300,000 carefully selected SNPs may be necessary to capture all of the relevant variation across the Caucasian human genome. Assuming this is true (it is probably a lower bound), we would need to scan $4.5x10^{10}$ pairwise combinations of SNPs to find a genetic needle. The number of higher order combinations is astronomical. Is GP suitable for a problem like this? At face value the answer is no. There is no reason to expect that a GP or any other wrapper method would perform better than a random attribute selector because there are no building blocks for this problem when accuracy is used as the fitness measure. The fitness of any given classifier would look no better than any other with just one of the two correct SNPs in the model. Indeed, we have observed this in our preliminary work [4,5]. Subsequent work has shown that by integrating expert knowledge into a selection scheme, it is possible to develop a GP wrapper that is able to perform better than a random attribute selector [6]. Work here examines whether or not it is also possible to integrate expert knowledge into mutation to develop a GP which performs better than one with a random attribute mutator.

1.2 Genetic Programming and Mutation

Genetic programming (GP) is an automated computational discovery tool that is inspired by Darwinian evolution and natural selection [7,8,9,10,11,12,13]. The goal of GP is to evolve computer programs to solve problems. This is accomplished by first generating computer programs that are composed of the building blocks needed to solve or approximate a solution to a problem. Each generated program is evaulated, and the good programs are selected, recombined, and mutated to form new computer programs. This process of selection based on fitness and recombination and mutation 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, knowledge discovery e.g. [14], and bioinformatics [15].

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 a standard in the modern problem solver's toolbox. Yu et al. [16] list 22 such challenges. Several of these are addressed by the present study. Previous work has shown that by integrating expert knowledge, the GP approach can successfully pick attributes from large and high-dimensional datasets [6]. Here we will examine another method of taking advantage of pre-processing based expert knowledge. Such methods may also be used for integrating domain specific or literature based expert knowledge. This paper explores the effect of a GP mutator which integrates expert knowledge on genome-wide genetic analysis in the domain of human genetics.

Previous work has shown that for a naïve mutation operator in a wide variety of problem domains, performance for naïve mutation and naïve crossver did not greatly differ [17]. There were, however, problem domains and parameter settings where mutation was found to be more or less suitable. In general mutation was more successful with more generations and crossover more successful with larger populations but the effect differed by problem domain. Previous work on expert knowledge guided mutation for genetic programming has used collective memory which contains knowledge gained by examining the population state at earlier time points within the same GP run [18]. Here we focus on expert knowledge gained by statistical pre-processing of the input data.

The goal of the present study was to develop and evaluate a GP mutation operator appropriate for genetic analysis of genome-wide data. Given the concept difficulty, we are generally interested in using expert knowledge to facilitate the generation and exploitation of good building blocks. We specifically address whether pre-processed expert knowledge can become useful for mutating trees after recombination and reproduction. In this specific situation mutation or random chance must, in addition to seeding the population with good attributes, combine the best attributes as no expert knowledge is utilized during selection or recombination. The success of expert knowledge in selection is limited to attributes available in the current population, but the use of expert knowledge in mutation is not limited to the state of the current population as attributes not present may be added. As both operators are dependent on different factors, integration of this mutation operator into a GP utilizing expert knowledge throughout may yield benefits beyond those found in this study.

2 Genetic Programming Methods

2.1 Expression Tree Representation

Figure 1A illustrates an example GP tree for this problem. We have intentionally kept the initial solution representation simple with one function in the root node and two children to evaluate the best GP parameterization. More complex trees (e.g. Figure 1B) will be explored once we understand when and how the GP works with the simpler trees. We have selected the multifactor dimensionality reduction or MDR approach as an attribute constructor for the function set

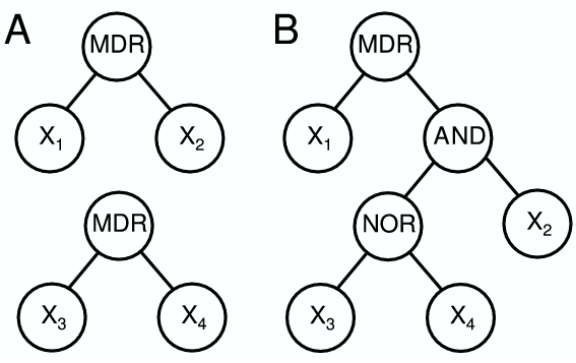

Fig. 1. Example GP trees for solutions (A). Example of a more complex tree that will be considered in future studies (B).

because it is able to capture interaction information (see Section 3). Each tree has two leaves or terminals consisting of attributes. In the case of our study, the terminal set consists of 1000 attributes.

2.2 Fitness Function

The fitness function used in this study was accuracy estimated using a naïve Bayes classifier. Here, accuracy is defined as how well the model predicts the case-control status of each simulated individual. Each tree is evaluated as a constructed attribute using the MDR function in the root node. It is this single constructed attribute with two levels that is assessed using the classifier. The classification accuracy of a tree is its fitness.

2.3 A Sensible Mutation Operator

The goal of this study was to use expert knowledge to ensure good building blocks are reintroduced into the population through mutation. We compared the random mutator with a new sensible or expert knowledge-guided mutator. The random mutator works by mutating the designated percentage of the population each generation. For each chosen individual a random attribute is picked for mutation and replaced with an attribute selected randomly from the set of all attributes. For the sensible mutator, we used pre-processed attribute quality from the Tuned ReliefF (TuRF) algorithm as our expert knowledge (see Section 4). The sensible mutation operator is modeled after the sensible selection operator from Moore and White [6] and mutates individuals from some percentage of the population with the greatest difference in TuRF scores between the attributes, chooses to mutate the attribute in that individual with the lowest TuRF score and iteratively creates trees where the mutated attribute is replaced with every attribute from the top 1% of TuRF scores. Next, trees are evaluated, and the tree with the best accuracy is retained in the population as the result of the

mutation. This sensible mutator ensures that poor building blocks are replaced by good building blocks throughout evolution. This method is designed to mutate attributes which are unlikely to lead to success and combine the newly chosen attribute with attributes which are likely to be involved in successful models. In the event that no combination of attributes leads to success, a higher TuRF scored attribute than the one which was mutated is retained in the population and may participate in recombination. The percentage of the population chosen for mutation is adjusted to account for the iterative replacement attempts so that the number of mutation attempts remains the same between the random mutator and the sensible mutator.

2.4 Parameter Settings

For this study, we used a population size of 500 and ran the GP for 10 generations as previously used by Moore and White [6]. We used a crossover probability of 0.9 and varied the mutation probability in increments of 10%. Since each tree has exactly two attributes, an initial population size of 500 trees will include 1,000 total attributes. Each initial population was generated such that each of the 1,000 attributes was represented once and only once across the 500 trees. This sensible initialization ensures that all building blocks are represented. It is important to note that the probability of any one tree receiving both functional attributes (i.e. the solution) is only $0.001x0.001$ or 10^{-6}. Thus, it is unlikely that any one tree in the initial population will be the correct solution. The size of the search space is approximately 500,000 or 1000 choose 2. With a population size of 500 and 10 generations the GP is exploring at most of 1% of the search space. The GP was implemented in C++ using GAlib (http://lancet.mit.edu/ga/). The crossover operator was modified to ensure binary trees of depth one.

3 Multifactor Dimensionality Reduction (MDR) for Attribute Construction

Multifactor dimensionality reduction (MDR) was developed as a nonparametric and genetic model-free data mining strategy for identifying combination of SNPs that are predictive of a discrete clinical endpoint [19,20,21,22]. The MDR method has been successfully applied to detecting gene-gene interactions for a variety of common human diseases including adverse drug reactions [23]. At the heart of the MDR approach is an attribute construction algorithm that creates a new attribute by pooling genotypes from multiple SNPs. Constructive induction using the MDR kernel is accomplished in the following way. Given a threshold T, a multilocus genotype combination is considered high-risk if the ratio of cases (subjects with disease) to controls (healthy subjects) exceeds or equals T, otherwise it is considered low-risk. Genotype combinations considered to be high-risk are labeled G1 while those considered low-risk are labeled G0. This process constructs a new one-dimensional attribute with levels G0 and G1. It is this new

single variable that is returned by the MDR function in the GP function set. The MDR method is described in more detail by Moore et al. [21]. Open-source MDR software is freely available from www.epistasis.org.

4 Expert Knowledge from Tuned ReliefF (TuRF)

Our goal was to provide an external measure of attribute quality that could be used as expert knowledge by the GP. Here this external measure used was statistical, but it could just as easily be biological. There are many statistical and computational methods for determining the quality of attributes. Our goal was to identify a method that is capable of identifying attributes that predict class primarily through dependencies or interactions with other attributes. Kira and Rendell [24] developed an algorithm called Relief that is capable of detecting attribute dependencies. Relief estimates the quality of attributes through a 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 [25] improved upon Relief by choosing n nearest neighbors instead of just one. This new ReliefF algorithm has been shown to be more robust to noisy attributes and missing data [26] 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 [27]. 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 [27] to each dataset.

5 Data Simulation and Analysis

The goal of the simulation study is to generate artificial datasets with high concept difficulty to evaluate the power of GP in the domain of human genetics.

Table 1. Penetrance values for an example epistasis model

	AA (0.36)	Aa (0.48)	aa (0.16)
BB (0.36)	0.077	0.656	0.880
Bb (0.48)	0.892	0.235	0.312
bb (0.16)	0.174	0.842	0.106

We first developed 60 different penetrance functions (i.e. genetic models) that on genotypes from two SNPs in the absence of any independent effects. The 60 penetrance functions include groups of five with heritabilities of 0.025, 0.05, 0.1, 0.2, 0.3, or 0.4. These heritabilities range from a very small to a large genetic effect size. In half of the cases, each functional SNP had two alleles with frequencies of 0.4 and 0.6. In the other half of the cases, each functional SNP had two alleles with frequencies of 0.2 and 0.8. Table 1 summarizes the penetrance values to three significant digits for one of the 60 models. The values in parentheses are the genotype frequencies. Each of the 60 models was used to generate 100 replicate datasets with a sample size of 1600. Each dataset consisted of an equal number of case (disease) and control (no disease) subjects. Each pair of functional SNPs was combined within a genome-wide set of 998 randomly generated SNPs for a total of 1000 attributes. A total of 6,000 datasets were generated and analyzed.

For each set of 100 datasets we counted the number of times the correct two functional attributes were selected as the best model by the GP. This count, expressed as a percentage, is an estimate of the power of the method. This percentage represents how often the GP finds the answer that we know is present.

6 Experimental Results

Figure 2 summarizes the average power for each method for the models with 0.6 major allele frequency. Results for 0.8 major allele frequency were similar and are available upon request. Each point represents the power averaged over 500 datasets (5 models with 100 datasets each). Power represents the number of times out of 100 that the GP found the right two attributes. The unfilled circles represent the average power for a GP using the random mutation operator with otherwise identical parameters. The filled circles represent the average power for a GP using our sensible mutation operator with TuRF pre-processing scores as expert knowledge. Within each parameter setting mutation was varied such that 0 to 100 percent of the individuals were mutated during each generation for the random mutator in increments of 10 percent. For the sensible mutation operator, this corresponds to 0 to 10 percent of the population being mutated during each generation in increments of 1 percent to retain identical numbers of attempted mutations. These results clearly show the value of using TuRF scores in the mutation operator.

Comparing the TuRF sensible mutator to TuRF sensible recombination shows that the TuRF mutator performs similarly to the TuRF selector [6]. The previously examined TuRF selector showed slightly higher power, though that is not unexpected. Without an expert knowledge guided selector, there was no way to utilize the strength of attributes already in the population for recombination.

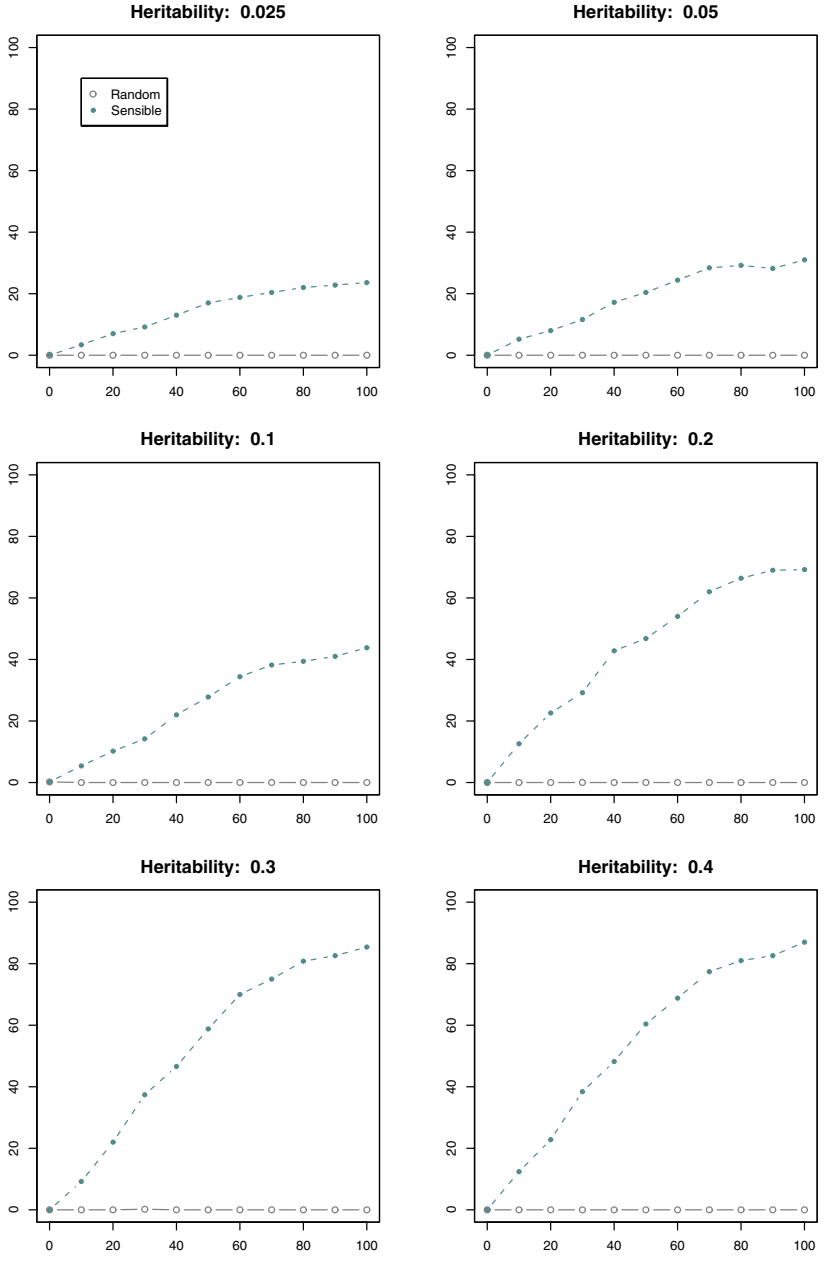

Fig. 2. Summary of the power of Random Mutation (Empty Circles) and Sensible Mutation (Filled Circles) using sensible initialization with accuracy for fitness under with random mutation equivalents (RMEs) from 0 to 100 under models with heritabilities from 0.025 to 0.4 and a major allele frequency of 0.6

7 Discussion and Conclusion

There are several conclusions to draw from this study. Again it has been shown that expert knowledge can provide building blocks necessary to find the genetic needle in the genome-wide haystack. Secondly, sensible mutation seems to perform similarly to using expert knowledge in selection and much better than random mutation. Sensible mutation itself is deterministic given a current population and set of TuRF scores. For the purposes of studying sensible mutation, we have separated random and sensible mutation. To retain the ability to reach the entire search space, in practical situations the pairing of sensible mutation and a non-deterministic mutation operator would probably be the best analysis strategy. In addition, combining sensible mutation with similar knowledge guided strategies in selection and recombination may provide additional benefits.

Previous work has focused on the use of expert knowledge in initialization, selection, and fitness [6,5]. We have focused on sensible mutation for this study. Future work will examine how different operators that integrate expert knowledge may be combined. Does using expert knowledge to guide both mutation and recombination work better than either alone?

In this work the building blocks of outside knowledge were obtained by preprocessing data with TuRF. For the realm of genetic studies, outside knowledge could also be obtained from the numerous public databases available to geneticists. Tools are being developed which integrate knowledge across these public databases and generate information about relationships between genes and disease in the context of protein interactions [28]. Future work will also focus on integrating multiple distinct expert knowledge types and sources.

We have again found that, given domain specific building blocks and operators which use these building blocks it is possible for a GP to outperform a random search, even for a needle-in-a-haystack problem. This indicates that GP may be a useful wrapper for genome wide analysis of common human diseases with a complex genetic architecture. Moore et al. have recently shown that Symbolic Discriminant Analysis (SDA), which uses a GP approach to generate models, was able to successfully model predictors of atrial fibrillation in a well characterized dataset which included a two-way epistatic interaction [29]. Integrating expert knowledge into the SDA approach should increase the efficiency of the search and assist SDA in finding higher order nonlinear interactions and allow SDA to be applied to larger genome-wide datasets.

Acknowledgement

This work was supported by NIH grants LM009012 and AI59694.

References

1. Freitas, A.A.: Understanding the crucial role of attribute interaction in data mining. Artif. Intell. Rev. 16(3), 177–199 (2001)
2. Goldberg, D.E.: The Design of Innovation: Lessons from and for Competent Genetic Algorithms. Kluwer Academic Publishers, Norwell, MA, USA (2002)

3. Consortium, T.I.H.: A haplotype map of the human genome. Nature 437(7063), 1299–1320 (2005)
4. White, B.C., Gilbert, J.C., Reif, D.M., Moore, J.H.: A statistical comparison of grammatical evolution strategies in the domain of human genetics. In: Proceedings of the IEEE Congress on Evolutionary Computing, pp. 676–682. IEEE Computer Society Press, Los Alamitos (2005)
5. Moore, J.H., White, B.C.: Genome-wide genetic analysis using genetic programming: The critical need for expert knowledge. In: Genetic Programming Theory and Practice IV, Springer, Heidelberg (2006)
6. Moore, J., White, B.: Exploiting expert knowledge in genetic programming for genome-wide genetic analysis. In: Runarsson, T.P., Beyer, H.-G., Burke, E., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) Parallel Problem Solving from Nature - PPSN IX. LNCS, vol. 4193, pp. 969–977. Springer, Heidelberg (2006)
7. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge, MA, USA (1992)
8. Koza, J.R.: Genetic programming II: automatic discovery of reusable programs. MIT Press, Cambridge, MA, USA (1994)
9. Koza, J.R., Andre, D., Bennett, F.H., Keane, M.A.: Genetic Programming III: Darwinian Invention & Problem Solving. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1999)
10. Koza, J.R.: Genetic Programming IV: Routine Human-Competitive Machine Intelligence. Kluwer Academic Publishers, Norwell, MA, USA (2003)
11. Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic programming: an introduction: on the automatic evolution of computer programs and its applications. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1998)
12. Langdon, W.B., Koza, J.R.: Genetic Programming and Data Structures: Genetic Programming + Data Structures = Automatic Programming! Kluwer Academic Publishers, Norwell, MA, USA (1998)
13. Langdon, W.B., Poli, R.: Foundations of Genetic Programming. Springer, Heidelberg (2002)
14. Freitas, A.A.: Data Mining and Knowledge Discovery with Evolutionary Algorithms, Secaucus, NJ, USA. Springer, New York (2002)
15. Fogel, G., Corne, D.: Evolutionary Computation in Bioinformatics. Morgan Kaufmann, San Francisco (2003)
16. Yu, T., Riolo, R., Worzel, B.: Genetic Programming: Theory and Practice (2006) 10.1007/0-387-28111-8_1
17. Luke, S., Spector, L.: A revised comparison of crossover and mutation in genetic programming. In: Koza, J.R., Banzhaf, W., Chellapilla, K., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M.H., Goldberg, D.E., Iba, H., Riolo, R. (eds.) Genetic Programming 1998: Proceedings of the Third Annual Conference, University of Wisconsin, Madison, Wisconsin, USA, pp. 208–213. Morgan Kaufmann, San Francisco (1998)
18. Bearpark, K., Keane, A.: The use of collective memory in genetic programming. In: Jin, Y. (ed.) Knowledge Incorporation in Evolutionary Computation. Studies in Fuzziness and Soft Computing, pp. 15–36. Springer, Heidelberg (2005)
19. Ritchie, M.D., Hahn, L.W., Roodi, N., Bailey, L.R., Dupont, W.D., Parl, F.F., Moore, J.H.: Multifactor dimensionality reduction reveals high-order interactions among estrogen metabolism genes in sporadic breast cancer. American Journal of Human Genetics 69, 138–147 (2001)

20. Moore, J.H.: Computational analysis of gene-gene interactions using multifactor dimensionality reduction. Expert Review of Molecular Diagnostics 4(6), 795–803 (2004)
21. Moore, J.H., Gilbert, J.C., Tsai, C.T., Chiang, F.T., Holden, T., Barney, N., White, B.C.: A flexible computational framework for detecting, characterizing, and interpreting statistical patterns of epistasis in genetic studies of human disease susceptibility. Journal of Theoretical Biology 241(2), 252–261 (2006)
22. Moore, J.H.: 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 (2007)
23. Wilke, R.A., Reif, D.M., Moore, J.H.: Combinatorial pharmacogenetics. Nature Reviews Drug Discovery 4, 911–918 (2005)
24. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: Machine Learning: Proceedings of the AAAI'92 (1992)
25. Kononenko, I.: Estimating attributes: Analysis and extension of relief. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)
26. Robnik-Sikonja, M., Kononenko, I.: Theoretical and empirical analysis of relieff and rrelieff. Mach. Learn. 53(1-2), 23–69 (2003)
27. Moore, J.H., White, B.C.: Tuning relieff for genome-wide genetic analysis. LNCS, vol. 4447, pp. 166–175 (2007)
28. Gonzalez, G., Uribe, J.C., Tari, L., Brophy, C., Baral, C.: Mining gene-disease relationships from biomedical literature: Weighting protein-protein interactions and connectivity measures. In: Pacific Symposium on Biocomputing, vol. 12, pp. 28–39 (2007)
29. Moore, J.H., Barney, N., Tsai, C.T., Chiang, F.T., Gui, J., White, B.C.: Symbolic modeling of epistasis. Hum. Hered. 63(2), 120–133 (2007)