DISCRETE AND CONTINUOUS
DYNAMICAL SYSTEMS–SERIES B
Volume **4**, Number **1**, February **2004**

Website: http://AIMsciences.org

pp. **275–287**

EVALUATION OF A DISCRETE DYNAMIC SYSTEMS APPROACH FOR MODELING THE HIERARCHICAL RELATIONSHIP BETWEEN GENES, BIOCHEMISTRY, AND DISEASE SUSCEPTIBILITY

Jason H. Moore and Lance W. Hahn

Program in Human Genetics
Department of Molecular Physiology and Biophysics
Vanderbilt University Medical School
Nashville, TN 37232

Abstract. A central goal of human genetics is the identification of combinations of DNA sequence variations that increase susceptibility to common, complex human diseases. Our ability to use genetic information to improve public health efforts to diagnose, prevent, and treat common human diseases will depend on our ability to understand the hierarchical relationship between complex biological systems at the genetic, cellular, biochemical, physiological, anatomical, and clinical endpoint levels. We have previously demonstrated that Petri nets are useful for building discrete dynamic systems models of biochemical networks that are consistent with nonlinear gene-gene interactions observed in epidemiological studies. Further, we have developed a machine learning approach that facilitates the automatic discovery of Petri net models thus eliminating the need for human-based trial and error approaches. In the present study, we evaluate this automated model discovery approach using four different nonlinear gene-gene interaction models. The results indicate that our model-building approach routinely identifies accurate Petri net models in a human-competitive manner. We anticipate that this general modeling strategy will be useful for generating hypotheses about the hierarchical relationship between genes, biochemistry, and measures of human health.

1. **Introduction.** There is a growing awareness that susceptibility to common human diseases such as sporadic breast cancer is largely due to nonlinear interactions among multiple genes and multiple environmental factors [11, 19, 23]. For example, Ritchie et al. [23] identified a combination of four genetic variations from three estrogen metabolism genes that are associated with risk of sporadic breast cancer. In this study, no one genetic variation was associated with breast cancer by itself. Rather, information from all four was necessary. These genes represent important candidates for determining breast cancer susceptibility because they are involved in the metabolism of estrogen. Estrogen can increase risk for sporadic breast cancer if it is metabolized into a compound that can damage DNA. The genes from the estrogen metabolism pathway play an important functional role in determining how much carcinogenic estrogen is produced.

2000 *Mathematics Subject Classification.* 92D30.
Key words and phrases. epistasis, gene-gene interactions, Petri nets, grammatical evolution.

New statistical and computational methods such as multifactor dimensionality reduction [8, 23, 24] are making it feasible to detect genes that influence breast cancer susceptibility primarily through nonlinear interactions with other genes. The ultimate public health goal is to use information about DNA sequence variations in genes to improve the diagnosis, prevention, and treatment of common human diseases such as sporadic breast cancer. Realizing this objective will partly depend on understanding how information at the genetic level is realized at the population level through biochemical and physiological systems. Understanding how combinations of genetic variations in estrogen metabolism genes increase levels of carcinogenic estrogen, and thus breast cancer susceptibility, is expected to lead to improved clinical management of this common disease. Making the connection between genes, biochemistry, and disease susceptibility using a discrete dynamic systems modeling approach is the focus of the present study.

We took the first step towards hierarchical systems modeling of disease susceptibility by addressing the following questions. First, is it possible to develop simple discrete dynamic systems models of biochemical networks that are consistent with nonlinear gene-gene interactions that are observed at the population level? Second, are these simple biochemical systems models biologically plausible? We used discrete dynamic system models called Petri nets to develop two independent, biologically plausible, biochemical systems models of a well-known nonlinear gene-gene interaction model (unpublished results). This preliminary study demonstrated the utility of Petri nets for modeling biochemical systems that are consistent with nonlinear gene-gene interactions in complex diseases. However, an important limitation of this modeling approach is that the Petri net models were developed by a human-based trial and error approach that is time consuming and difficult due to combinatorial complexities. In response to this limitation, Moore and Hahn [17] developed a machine intelligence strategy that uses an evolutionary computation approach called grammatical evolution for the automatic discovery of Petri net models. This approach routinely generates Petri net models that are consistent with two published genetic models in which disease susceptibility is dependent on nonlinear interactions between two DNA sequence variations [17].

The goal of the present study is to evaluate the ability of the grammatical evolution approach proposed by Moore and Hahn [17] to discover Petri net models of biochemical systems that are consistent with nonlinear gene-gene interactions for a wide range of different genetic models. We have selected four different genetic models in which disease susceptibility is dependent on nonlinear interactions between two DNA sequence variations. We find that the modeling approach routinely identifies Petri net models that are consistent with each of the four gene-gene interaction models.

2. The Nonlinear Gene-Gene Interaction Models.

Our four nonlinear gene-gene interaction models are based on penetrance functions. Penetrance functions represent one approach to modeling the relationship between genetic variations and risk of disease. 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 determined by one allele (i.e. a specific DNA sequence state) inherited from the mother and one allele inherited from the father. For most genetic variations, only two alleles (encoded by A or a) exist in the biological population. Therefore, because the ordering of the alleles is unimportant, a genotype can have

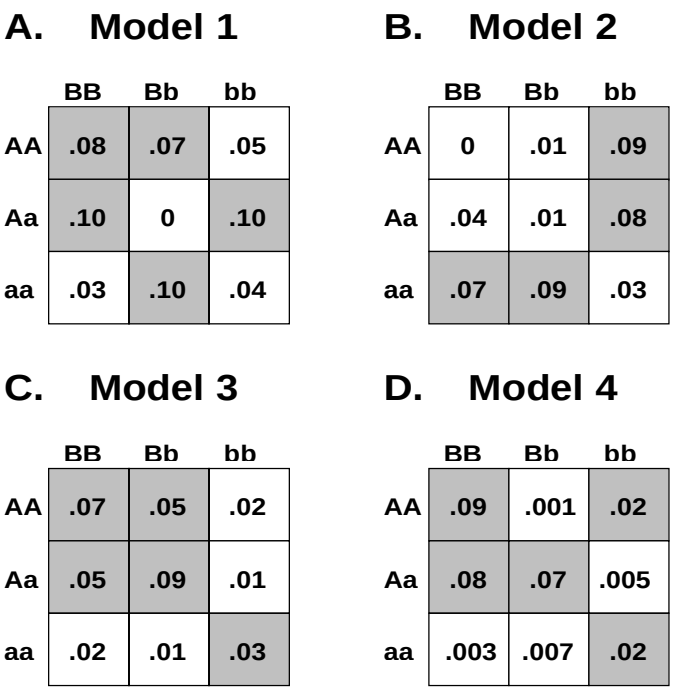

FIGURE 1. Penetrance functions for each nonlinear gene-gene interaction model. Note that the average penetrance values for each specific genotype are all approximately equal (not shown). Shaded cells represent the high risk genotype combinations while unshaded cells represent low risk combinations.

one of three values: AA, Aa or aa. Figure 1 illustrates each of the four penetrance function models we used. Each of these models was discovered using the software of Moore et al. [18] and have been described previously by Ritchie et al. [24]. What makes these models interesting is that disease risk is dependent on the particular combination of genotypes inherited. There is effectively no difference in disease risk for each single genotype as specified by the single-genotype penetrance values in the margins of the tables. Also illustrated in Figure 1 for each model is the distribution of high risk and low risk genotype combinations. A genotype combination was considered high risk if the probability of disease was greater than the prevalence of disease in the general population which is indicated by the marginal penetrance values based on single genotypes.

3. **Introduction to Petri Nets for Modeling Discrete Dynamic Systems.** Petri nets are a type of directed graph that can be used to model discrete dynamical systems [2]. Goss and Peccoud [7] demonstrated that Petri nets could be used to model molecular interactions in biochemical systems. The core Petri net consists of two different types of nodes: places and transitions. Using the biochemical systems analogy of Goss and Peccoud [57] places represent molecular species. Each place

has a certain number of tokens that represent the number of molecules for that particular molecular specie. A transition is analogous to a molecular or chemical reaction and is said to fire when it acquires tokens from a source place and, after a possible delay, deposits tokens in a destination place. Tokens travel from a place to a transition or from a transition to a place via arcs with specific weights or bandwidths. While the number of tokens transferred from place to transition to place is determined by the arc weights, the rate at which the tokens are transferred is determined by the delay associated with the transition. Transition behavior is also constrained by the weights of the source and destination arcs. A transition will only fire if two preconditions are met: 1) if the source place can completely use the capacity of the source arc and, 2) if the destination place has the capacity available to store the number of tokens provided by the full weight of the destination arc. Transitions without an input arc act as if they are connected to a limitless supply of tokens. Similarly, transitions without an output arc can consume a limitless supply of tokens. The transition firing rate can be immediate, delayed deterministically or stochastically depending on the complexity needed. The fundamental behavior of a Petri net can be controlled by varying the maximum number of tokens a place can hold, the weight of each arc, and the firing rates of the transitions.

4. **Our Petri Net Modeling Strategy.** Moore and Hahn [17] developed a strategy for identifying Petri net models of biochemical systems that are consistent with observed population-level gene-gene interactions. The specific Petri nets used to model the biochemical pathways are Petri Nets with Time [16, 21]. Transitions had either a fixed delay or fired as soon as the preconditions of the transition were met. If a place provided input to two or more transitions but had only enough tokens to satisfy one transition, then the transition with the shortest delay fired. If a place provided input to two or more transitions and had enough tokens to satisfy more than one transition, then the timers associated with both transitions began to count down. When the timers had counted down to 0, the transition fired unless two transitions were simultaneously ready to fire in which case one of the transitions is chosen to fire and the other transition(s) reset.

The goal of identifying Petri net models of biochemical systems that are consistent with observed population-level gene-gene interactions is accomplished by developing Petri nets that are dependent on specific genotypes from two or more genetic variations. Here, we make firing rates of transitions and/or arc weights genotype-dependent yielding different Petri net behavior. Each Petri net model is related to the genetic model using a discrete version of the threshold model from population genetics [3]. With a classic threshold or liability model, it is the concentration of a biochemical or environmental substance that is related to the risk of disease, under the hypothesis that risk of disease is greatly increased once a particular substance exceeds some threshold concentration (e.g. high concentrations of carcinogenic estrogen are a risk factor for sporadic breast cancer). Conversely, the risk of disease may increase in the absence of a particular factor or with any significant deviation from a reference level. In such cases, high or low levels are associated with high risk while an intermediate level is associated with low risk. Here, we use a discrete version of this model for our deterministic Petri nets. For each model, the number of tokens at a particular place is recorded and if they exceed a certain threshold, the appropriate risk assignment is made. If the number of tokens does not exceed the threshold, the alternative risk assignment is made. The

high-risk and low-risk assignments made by the discrete threshold from the output of the Petri net can then be compared to the high-risk and low-risk genotypes from the genetic model. A perfect match indicates the Petri net model is consistent with the gene-gene interactions observed in the genetic model. The Petri net then becomes a model that relates the genetic variations to risk of disease through an intermediate biochemical network.

Identifying Petri net models that are consistent with the genotype-dependent distribution of risk is challenging by hand. Therefore, Moore and Hahn [17] developed an evolutionary computing approach to the discovery of Petri net models. This approach is described in the next section.

5. **A Grammatical Evolution Approach to Discovering Petri Net Models.**

Overview of Grammatical Evolution

Evolutionary computation arose from early work on evolutionary programming [4, 5] and evolution strategies [22, 25] that used simulated evolution for artificial intelligence. The focus on representations at the genotypic level lead to the development of genetic algorithms by Holland [9, 10] and others. Genetic algorithms have become a popular machine intelligence strategy because they can be effective for implementing parallel searches of rugged fitness landscapes [6]. Briefly, this is accomplished by generating a random population of models or solutions, evaluating their ability to solve the problem at hand, selecting the best models or solutions, and generating variability in these models by exchanging model components among different models. The process of selecting models and introducing variability is iterated until an optimal model is identified or some termination criteria are satisfied. A limitation of genetic algorithms is that models or solutions are represented by linear arrays of bits. In response to this limitation, Koza [13] developed a more flexible evolutionary computation strategy called genetic programming where the models or solutions are represented by binary expression trees. Koza et al. [14] and others [12] have successfully applied genetic programming to modeling metabolic networks.

Grammatical evolution has been described by O'Neill and Ryan [20] as a variation on genetic programming. Here, a Backus-Naur Form (BNF) grammar is specified that allows a computer program or model to be constructed by a simple genetic algorithm operating on an array of bits. The ability to specify a grammar is appealing because only a text file specifying the grammar needs to be altered for different applications. There is no need to modify and recompile source code during development once the fitness function is specified. The end result is a decrease in development time and an increase in computational flexibility. It is for this reason that Moore and Hahn [17] selected grammatical evolution instead of genetic programming as the evolutionary computation method for the discovery of Petri net models. It is the goal of this study to evaluate the grammatical evolution approach to discovering Petri net models using four different nonlinear gene-gene interaction models that include two DNA sequence variations. We describe below the genetic algorithm used, the grammar for the Petri net, the fitness function, and the genetic algorithm parameters used.

A Grammar for Petri Net Models in Backus-Naur Form

Moore and Hahn [17] developed a grammar for Petri nets in Backus-Naur Form (BNF). Backus-Naur Form is a formal notation for describing the syntax of a

context-free grammar as a set of production rules that consist of terminals and nonterminals [15]. Nonterminals form the left-hand side of production rules while both terminals and nonterminals can form the right-hand side. A terminal is essentially a model element while a nonterminal is the name of a production rule. For the Petri net models, the terminal set includes, for example, the basic building blocks of a Petri net: places, arcs, and transitions. The nonterminal set includes the names of production rules that construct the Petri net. For example, a nonterminal might name a production rule for determining whether an arc has weights that are fixed or genotype-dependent. We show below in (1) the production rule that is executed to begin the model building process.

$$\begin{aligned} \text{<root>} \quad ::= \; & \text{<pick\_a\_gene><pick\_a\_gene><net\_iterations><expr>} \\ & \text{<transition><place\_noarc>} \end{aligned} \tag{1}$$

When the initial <root> production rule is executed, a single Petri net place with no entering or exiting arc (i.e. <place_noarc>) is selected and a transition leading into or out of that place is selected. The arc connecting the transition and place can be dependent on the genotypes of the genes selected by <pick_a_gene>. The nonterminal <expr> is a function that allows the Petri net to grow. The production rule for <expr> is shown below in (2). Here, the selection of one of the four nonterminals (0, 1, 2, or 3) in the right-hand side of the production rule is determined by a combination of bits in the genetic algorithm chromosome.

$$\begin{aligned} \text{<expr>} \quad ::= \; & \text{<expr><expr>} \quad 0 \\ & \mid \text{<arc>} \quad 1 \\ & \mid \text{<transition>} \quad 2 \\ & \mid \text{<place>} \quad 3 \end{aligned} \tag{2}$$

The base or minimum Petri net that is constructed using the <root> production rule consists of a single place, a single transition, and an arc that connects them. Multiple calls to the production rule <expr> by the genetic algorithm chromosome can build any connected Petri net. In addition, the number of times the Petri net is to be iterated is selected with the nonterminal <net_iterations>. Many other production rules control the arc weights, the genotype-dependent arcs and transitions, the number of initial tokens in a place, the place capacity, etc. All decisions made in the building of the Petri net model are made by each subsequent bit or combination of bits in the genetic algorithm chromosome. The complete grammar is too large for presentation in detail here but can be obtained from the authors upon request.

The Fitness Function

Once a Petri net model is constructed using the BNF grammar, as instructed by the genetic algorithm chromosome, the model fitness is determined. As described by Moore and Hahn [17], this is carried out by executing the Petri net model for each combination of genotypes in the genetic dataset and comparing the final token counts at a defined place to a threshold constant to determine the risk assignment. Let G be the set of $i = 1$ to n possible genotype combinations where $n = 9$ when there are two genetic variations, each with three genotypes. Let Z_i be the final number of tokens from the designated Petri net place for the ith genotype combination and let c be the threshold constant. Let $d(G_i)$ be the risk assignment for the ith genotype combination in the genetic model and let $f(G_i)$ be the risk assignment made by the Petri net. If $Z_i \geq c$ then $f(G_i) =$ "high risk" else if

TABLE 1. The Genetic Algorithm Parameter Settings

Objective	Discover Petri net models
Fitness function	Classification error
Number of runs	100
Stopping criteria	Classification error = 0
Population size	6000
Number of demes	6
Generations	800
Selection	Stochastic uniform sampling
Crossover	Uniform
Crossover probability	0.60
Mutation probability	0.02

$Z_i < c$ then $f(G_i) =$ "low risk." The dichotomous risk assignment is consistent with epidemiological study designs in which subjects with the disease (cases) and subjects without the disease (controls) are used to identify genetic risk factors. Genotypes that are more common in cases than controls can be thought of as high risk [8, 19, 23, 24]. Fitness (E) of the Petri net model is determined by comparing the high risk and low risk assignments made by the Petri net to those from the given nonlinear gene-gene interaction model. Calculation of the fitness value, E, is given by the classification error function in (3). In the present study, $\max(E) = 9$ and $\min(E) = 0$. The goal is to minimize E.

$$E = \sum_{i=1}^{|G|} e_i, \tag{3}$$

where

$$e_i = 0 \quad \text{if } f(G_i) = d(G_i),$$
$$e_i = 1 \quad \text{if } f(G_i) \neq d(G_i).$$

The Genetic Algorithm Parameters

Table 1 summarizes the genetic algorithm parameter settings used in this study. We ran the genetic algorithm a total of 100 times with different random seeds for each gene-gene interaction model. Each run consisted of a maximum of 800 generations. The genetic algorithm was stopped when a model with a classification error of zero (i.e. $E = 0$) was discovered. We used a parallel search strategy [1] with six demes (i.e. subpopulations) each with 1000 individuals or solutions for a total population size of 6000. A best chromosome migrated from each deme to all other demes every 25 generations. Each chromosome consisted of 14 32-bit bytes. It is possible to reach the end of a chromosome with an incomplete instance of the grammar. To complete the instance, chromosome wrap around was used [20]. In other words, the instance of the grammar was completed be reusing the chromosome as many times as was necessary to complete the instance. We also used a dynamic codon strategy in which the number of bits consumed in deciding on the right-hand side of a given production rule depended on the number of bits required for that decision rather than some fixed number of bits. For instance, a production with only two alternatives consumed only a single bit from the genetic algorithm chromosome.

Software and Hardware

TABLE 2. Summary of the distribution (mode and range) of the number of different Petri net elements identified across 100 grammatical evolution runs for the four nonlinear gene-gene interaction models.

	Mode (range) number of Petri net elements			
Petri net models	Model 1	Model 2	Model 3	Model 4
Place	1 (1-1)	1 (1-2)	1 (1-1)	1 (1-2)
Arc	2 (2-5)	2 (2-7)	2 (2-9)	2 (2-6)
Transition	2 (1-4)	2 (1-5)	2 (1-6)	2 (1-5)
Conditional	3 (2-6)	3 (2-6)	3 (2-8)	3 (2-7)

The parallel genetic algorithm used was a modification of the Parallel Virtual Machine (PVM) version of the Genetic ALgorithm Optimized for Portability and Parallelism System (GALLOPS) package for UNIX [http://garage.cps.msu.edu/software/software-index.html]. This package was implemented in parallel using message passing on a 110-processor Beowulf-style parallel computer cluster running the Linux operating system. Seven total processors were used for each separate run.

6. **Results.** The grammatical evolution algorithm was run a total of 100 times for each of the four nonlinear gene-gene interaction models. For model 1, the grammatical evolution strategy yielded a Petri net model that was perfectly consistent with the high-risk and low-risk assignments for each combination of genotypes with no classification error in 99 out of 100 runs. For models 2-4, the grammatical evolution strategy yielded a Petri net model that was perfectly consistent with the high-risk and low-risk assignments for each combination of genotypes with no classification error in 100 out of 100 runs. Thus, Petri net model discovery was routine and human competitive. Table 2 below summarizes the mode (i.e. most common) and range of the number of places, arcs, transitions, and conditionals (i.e. genotype-dependent elements) that define the best Petri net models found across the 100 runs for each model. For all gene-gene interaction models, most Petri net models consisted of one place, two arcs, and two transitions. In addition, the best models were most likely to have three Petri net elements that are conditional or dependent on genotype.

Figure 2A illustrates a Petri net architecture for model 1 that was commonly found by the grammatical evolution algorithm. This model consists of one place (P_0), two arcs (A_0 and A_1), and two transitions (T_0 and T_1). An arc or transition that is genotype-dependent is indicated by G_i $\{X_1, X_2, X_3\}$ for the ith genetic variation ($i = 0$ or 1) where the weights associated with the three genotypes are in brackets. For this model, the place is initialized with 11 tokens and has a maximum capacity of 16 tokens. Transition 0 has a fixed firing delay of 10 time steps. Both arcs and transition 1 were genotype-dependent. For example, transition 1 has a firing rate of 13 if the genotype for the first genetic variation is AA. This Petri net was iterated for 47 time intervals and was formed by using 87 of the 448 available bits on the genetic algorithm chromosome. The final token counts for each of the nine genotype combinations are shown in Figure 3A. Note that token counts greater than 7 are associated with high-risk while those equal to or less than 7 are associated with low-risk. This is consistent with the penetrance values for this model (see Figure 1A).

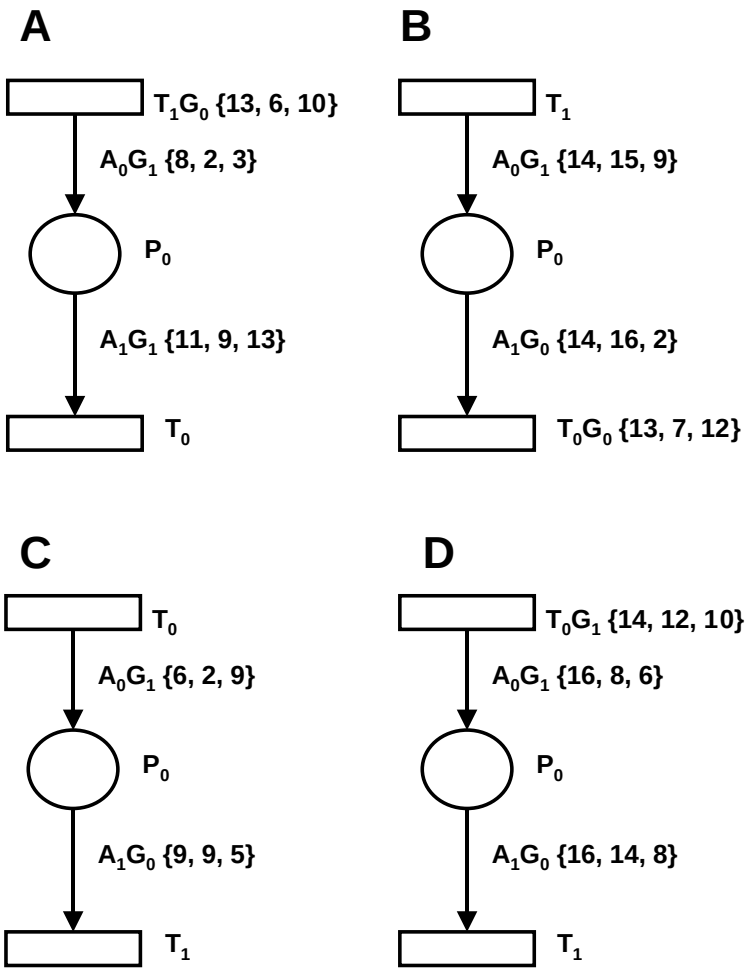

FIGURE 2. Common Petri net models identified in multiple runs of the grammatical evolution algorithm for model 1 (A), model 2 (B), model 3 (C), and model 4 (D).

Figure 2B illustrates a Petri net architecture for model 2 that was commonly found by the grammatical evolution algorithm. For this model, the place is initialized with 6 tokens and has a maximum capacity of 16 tokens. Transition 1 has no firing delay. Both arcs and transition 0 were genotype-dependent. This Petri net was iterated for 42 time intervals and was formed by using 82 of the 448 available bits on the genetic algorithm chromosome. The final token counts for each of the nine genotype combinations are shown in Figure 3B. Note that token counts greater than 9 are associated with high-risk while those equal to or less than 9 are associated with low-risk. This is consistent with the penetrance values for this model (see Figure 1B).

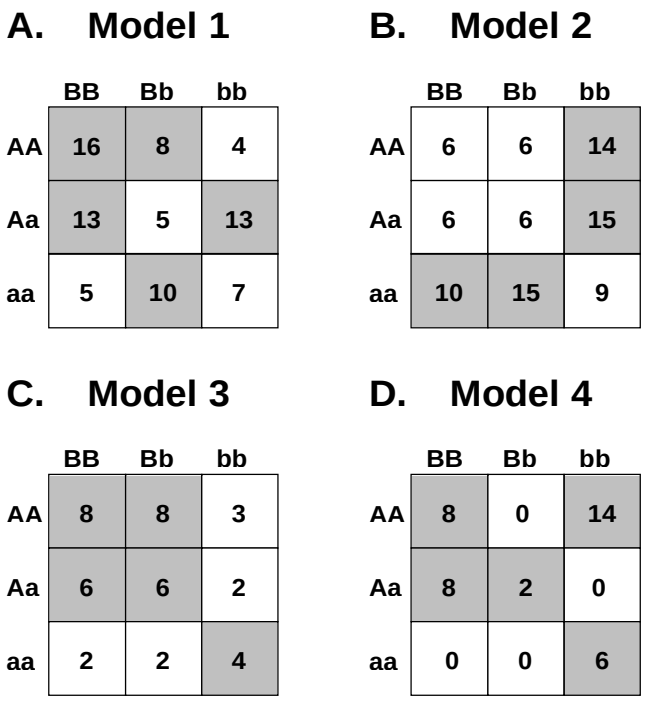

FIGURE 3. Final token counts for each of the nine genotype combinations after the final iteration of the Petri net models illustrated in Figure 2 for gene-gene interaction model 1 (A), model 2 (B), model 3 (C), and model 4 (D).

Figure 2C illustrates a Petri net architecture for model 3 that was commonly found by the grammatical evolution algorithm. For this model, the place is initialized with 2 tokens and has a maximum capacity of 12 tokens. Transitions 0 and 1 have no firing delays. Both arcs were genotype-dependent. This Petri net was iterated for 20 time intervals and was formed by using 68 of the 448 available bits on the genetic algorithm chromosome. The final token counts for each of the nine genotype combinations are shown in Figure 3C. Note that token counts greater than 3 are associated with high-risk while those equal to or less than 3 are associated with low-risk. This is consistent with the penetrance values for this model (see Figure 1C).

Figure 2D illustrates a Petri net architecture for model 4 that was commonly found by the grammatical evolution algorithm. For this model, the place is initialized with 8 tokens and has a maximum capacity of 16 tokens. Transition 0 has no firing delay. Both arcs and transition 0 were genotype-dependent. This Petri net was iterated for 20 time intervals and was formed by using 82 of the 448 available bits on the genetic algorithm chromosome. The final token counts for each of the nine genotype combinations are shown in Figure 3D. Note that token counts greater

than 1 are associated with high-risk while those equal to or less than 1 are associated with low-risk. This is consistent with the penetrance values for this model (see Figure 1D).

7. **Discussion.** The main conclusion of this study is that the Petri net modeling approach of Moore and Hahn [17] routinely identifies discrete dynamic systems models that are consistent with the genotype-specific distributions of disease risk. In fact, across 100 runs, the grammatical evolution algorithm discovered Petri net models that were consistent with the high-risk and low-risk assignments for each combination of genotypes with no classification error in at least 99 out of 100 runs across the four different nonlinear gene-gene interaction models. The results suggest that this modeling approach is flexible and is competitive with human trial and error model building approaches.

The next step in the evaluation of this approach is to determine whether it is capable of modeling higher-order interactions. That is, is it possible to construct Petri net models that are consistent with the effects of more than two genetic variations at a time. We anticipate that this will be a more difficult task since the number of genotype combinations goes up exponentially. For example, there are nine genotype combinations for two genetic variations, 27 for three genetic variations, and 81 for four genetic variations. We anticipate the Petri net models will need to be much larger and more complex to generate token counts that are consistent with the distribution of high risk and low risk genotype combinations in these larger spaces.

The ultimate goal is to apply this modeling strategy to real data. Ritchie et al. [23, 24] and Hahn et al. [8] have developed a statistical approach called multifactor dimensionality reduction (MDR) for identifying combinations of genotypes associated with high and low risk of disease. The Petri net approach could be used to construct biochemical systems models that are consistent with the high and low risk models obtained from MDR. While these models may in no way represent the true underlying biochemical system, they may tell us something about the nature of the interactions at the biochemical level. For example, Moore and Hahn [17] found that arcs were more likely to be genotype-dependent than transitions for the genetic model examined. Perhaps the functionality of an arc indicates the general type of biological function that may be important at the biochemical level.

In this study, we used the final token count at a single place to indicate the risk assignment. It may be useful to change this metric to a more biologically plausible metric that depends on a feature of the dynamics of the system rather than a fixed static endpoint. In this case, the steady state level of the system might be an indicator of risk. In addition, there is the opportunity to explore a wider range of Petri net model elements. For example, it is possible to use continuous instead of discrete places or to use both continuous and discrete places (these are called hybrid Petri nets).

It is well established that nonlinear interactions among multiple genes are likely to play an important role in susceptibility to common, complex human diseases such as essential hypertension and sporadic breast cancer [11, 19, 23]. This is partly due to the inherent complexity of genetic and biochemical networks. Understanding how interactions at the biochemical level manifest themselves as interactions among genes at the population level, will provide a basis for understanding the role of genes in diseases susceptibility. Making this hierarchical connection may ultimately lead

to an understanding of complex biological systems that will facilitate new treatment and prevention strategies. We anticipate that this study, and others like it, will open the door for the synthesis and analysis of biomedical data with the ultimate goal of improving the diagnosis, prevention, and treatment of common diseases that represent the greatest public health burden.

Acknowledgements This work was supported by National Institutes of Health grants HL65234, HL65962, GM31304, AG19085, and AG20135. We thank an anonymous reviewer for very helpful comments and suggestions. We also thank Marylyn Ritchie, Tricia Thornton, and Bill White for many thoughtful discussions.

REFERENCES

[1] E. Cantu-Paz. "Efficient and Accurate Parallel Genetic Algorithms," Kluwer Academic Publishers, 2000.

[2] J. Desel and G. Juhas. *What is a Petri net? Informal answers for the informed reader,* in "Unifying Petri Nets, Lecture Notes in Computer Science 2128" (eds. H. Ehrig, G. Juhas, J. Padberg, and G. Rozenberg), Springer, 2001.

[3] D. S. Falconer, T. F. C. Mackay, "Introduction to Quantitative Genetics," 4th ed., Longman, Essex, 1996.

[4] L. J. Fogel. *Autonomous automata,* Industrial Research **4** (1962), 14–19.

[5] L. J. Fogel, A. J. Owens, M. J. Walsh. "Artificial Intelligence through Simulated Evolution," John Wiley, New York, 1966.

[6] D. E. Goldberg. "Genetic Algorithms in Search, Optimization, and Machine Learning," Addison-Wesley, 1989.

[7] P. J. Goss, J. Peccoud. *Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets,* Proceedings of the National Academy of Sciences USA **95** (1998), 6750–5.

[8] L. W. Hahn, M. D. Ritchie, and J. H. Moore. *Multifactor dimensionality reduction software for detecting gene-gene and gene-environment interactions,* Bioinformatics **19** (2003), 376–82.

[9] J. H. Holland. *Adaptive plans optimal for payoff-only environments,* in "Proceedings of the 2nd Hawaii International Conference on Systems Sciences," University of Hawaii, North Hollywood, 1969.

[10] J. H. Holland. "Adaptation in Natural and Artificial Systems," University of Michigan Press, Ann Arbor, 1975.

[11] S. L. R. Kardia. *Context-dependent genetic effects in hypertension,* Current Hypertension Reports **2** (2000), 32–38.

[12] J. Kitagawa and H. Iba. *Identifying metabolic pathways and gene regulation networks with evolutionary algorithms,* in "Evolutionary Computation and Bioinformatics," (eds. G. B. Fogel and D. W. Corne), Morgan Kaufmann Publishers, 2003.

[13] J. R. Koza. "Genetic Programming: On the Programming of Computers by Means of Natural Selection," The MIT Press, 1992.

[14] J. R. Koza, W. Mydlowec, G. Lanza, J. Yu, and M. A. Keane. *Reverse engineering of metabolic pathways from observed data using genetic programming,* Pacific Symposium on Biocomputing **6** (2001), 434–45.

[15] M. Marcotty and H. Ledgard. "The World of Programming Languages," Springer-Verlag, 1986.

[16] P. Merlin. "A Study of the Recoverability of Computer Systems," Ph.D. Thesis, University of California, Irvine, 1974.

[17] J. H. Moore and L. W. Hahn. *Grammatical evolution for the discovery of Petri net models of complex genetic systems,* in "Lecture Notes in Computer Science" (eds. Cantu-Paz, E. et al.), in press, Springer-Verlag, Berlin, 2003.

[18] J. H. Moore, L. W. Hahn, M. D. Ritchie, T. A. Thornton and B. C. White. *Application of genetic algorithms to the discovery of complex genetic models for simulation studies in human genetics,* in "Proceedings of the Genetic and Evolutionary Computation Conference" (eds. W.B. Langdon et al.), Morgan Kaufmann Publishers, San Francisco, 2002.

[19] J.H. Moore and S.M. Williams. *New strategies for identifying gene-gene interactions in hypertension,* Annals of Medicine **34** (2002), 88–95.
[20] M. O'Neill and C. Ryan. *Grammatical evolution,* IEEE Transactions on Evolutionary Computation **5** (2001), 349–358.
[21] C. Ramchandani. "Analysis of Asynchronous Concurrent Systems by Timed Petri Nets," Ph.D. Thesis, MIT, Cambridge, 1974.
[22] I. Rechenberg. "Cybernetic solution path of an experimental problem," Royal Aircraft Establishment, Farnborough, U.K., Library Translation No. 1122, August, 1965.
[23] M. D. Ritchie, L. W. Hahn, N. Roodi, L. R. Bailey, W. D. Dupont, W. D. Plummer, F. F. Parl, and J. H. Moore. *Multifactor dimensionality reduction reveals high-order interactions among estrogen metabolism genes in sporadic breast cancer,* American Journal of Human Genetics **69** (2001), 138–147.
[24] M. D. Ritchie, L. W. Hahn, and J. H. Moore. *Power of multifactor dimensionality reduction for detecting gene-gene and gene-environment interactions,* Genetic Epidemiology **24** (2003), 150–157.
[25] H.-P. Schwefel. "Kybernetische Evolution als Strategie der experimentellen Forschung in der Stromungstechnik," Diploma Thesis, Technical University of Berlin, 1965.

Received January 2003; revised August 2003.

E-mail address: moore@phg.mc.vanderbilt.edu