Grammatical Evolution Strategies for Bioinformatics and Systems Genomics

Jason H. Moore and Moshe Sipper

DRAFT

Abstract Evolutionary computing methods are an attractive option for modeling complex biological and biomedical systems because they are inherently parallel, they conduct stochastic search through large solution spaces, they capitalize on the modularity of solutions, they have flexible solution representations, they can utilize expert knowledge, they can consider multiple fitness criteria, and they are inspired by how evolution optimizes fitness through natural selection. Grammatical evolution (GE) is a promising example of evolutionary computing because it generates solutions to a problem using a generative grammar. We review here several detailed examples of GE from the bioinformatics and systems genomics literature and end with some ideas about the challenges and opportunities for integrating GE into biological and biomedical discovery.

1 Introduction

Bioinformatics has its origins in the late 1970s with the convergence of DNA sequencing, internetworking, and microcomputers. Early demand for bioinformatics centered on the use of computers and the internet to store, manage, manipulate, and analyze DNA sequences derived from experimental studies in the biological and biomedical sciences. This demand exploded in the mid-1990s with the advent of high-throughput methods for measuring

Jason H. Moore
Institute for Biomedical Informatics, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, 19087 USA, e-mail: jhmoore@upenn.edu

Moshe Sipper
Institute for Biomedical Informatics, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, 19087 USA, and Computer Science Department, Ben-Gurion University, Israel, e-mail: sipper@upenn.edu

biomolecules such as messenger RNA levels in cells and tissues [17]. This explosion of data has continued and, when combined with questions about the complexity of biological systems, creates computational challenges that often require machine learning and artificial intelligence (AI) approaches [6].

Evolutionary computation has emerged as a useful artificial intelligence approach for the study of complex biological systems because these methods are inherently parallel, conduct stochastic search through large solution spaces, capitalize on the modularity of solutions—which is an important characteristic of biological systems, have flexible solution representations, can utilize expert knowledge, can consider multiple fitness criteria, and are inspired by how evolution optimizes fitness through natural selection that is understood by biologists. Genetic programming (GP) is a population type of evolutionary computing [14, 26]. The goal of GP is to 'evolve' computer programs to solve complex problems. This is accomplished by first generating, or initializing, a population of random computer programs that are composed of the basic building blocks needed to solve or approximate a solution to the problem. The power of GP is its ability to recombine building blocks to create new solutions through an iterative process that involves selection of the best solutions. GP and its many variations have been applied successfully in a wide range of different problem domains including bioinformatics. The potential for evolutionary methods to impact complex problem solving was discussed in a recent editorial [27]. The goal of this chapter is to review bioinformatics and systems genomics applications of a type of GP called grammatical evolution (GE) that generates computer programs or solutions using a grammar. These grammar-based approaches provide tremendous flexibility.

Grammatical evolution (GE) was introduced by [25] 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. BNF 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. The GE approach 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 for evaluating solutions is specified.

We begin in the next section with a brief summary of GE applications and some thoughts about the future of this approach for solving complex biological and biomedical problems. We then review in some detail in the next two sections a bioinformatics application of GE for machine learning in human genetics and a systems genomics application of GE for simulating discrete dynamical systems.

2 A Survey of Grammatical Evolution Approaches to Bioinformatics and Systems Genomics

A search of the phrase "grammatical evolution" on PubMed revealed only 25 publications. In addition to the studies discussed below, several other application of GE have been reported. For example, [28] used GE to do feature selection and feature engineering to analyze electroencephalogram (EEG) data from patients experiencing epileptic seizures. In this case, the GE performed as well as other methods and provided the added benefit of the grammar for rapid development and testing. As another example, [4] used GE to study the behavior of insects. They found that GE could model self-organized task specialization using low-level behavioral primitives as building blocks for more complex behaviors. As a third example, [7] used GE to model and predict glucose concentrations in physiological systems. The results of this study have important implications for predicting insulin need in diabetic patients following carbohydrate intake. More recently, [3] used grammatical genetic programming to evolve control heuristics for heterogeneous cellular networks. Finally, GE has been used in the context of artificial life experiments. For example, [1] used GE to study the ecology of mathematical expressions as a way to study biological evolution. We also searched for "grammatical evolution" and the keyword "bioinformatics" in the genetic programming bibliography to capture publications in computer science conferences and other venues not captured by PubMed. This search returned 13 publications nearly all of which will be discussed below.

3 A Grammatical Evolution Approach to Neural Network Analysis of Human Genetics Data

An important goal of human genetics and genetic epidemiology is to understand the mapping relationship between interindividual variation in DNA sequences, variation in environmental exposure, and variation in disease susceptibility. In other words, how do one or more changes in an individual's DNA sequence increase or decrease their risk of developing disease through complex networks of biomolecules that are hierarchically organized, highly interactive, and dependent on environmental exposures? Understanding the role of genomic variation and environmental context in disease susceptibility is likely to improve diagnosis, prevention, and treatment. Success in this important public-health endeavor will depend critically on the amount of nonlinearity in the mapping of genotype to phenotype and our ability to address it. Here, we define as nonlinear an outcome that cannot be easily predicted by the sum of the individual genetic markers. Nonlinearities can arise from phenomena such as locus heterogeneity (i.e. different DNA sequence variations

leading to the same phenotype), phenocopy (i.e. environmentally determined phenotypes that don't have a genetic basis), and the dependence of genotypic effects on environmental exposure (i.e. gene-environment interactions or plastic reaction norms) and genotypes at other loci (i.e. gene-gene interactions or epistasis). The challenges associated with detecting each of these phenomena in big data has been reviewed and discussed by [18] who call for an analytical retooling to address these complexities.

The limitations of the linear model and other parametric statistical approaches for modeling nonlinear interactions have motivated the development of data mining and machine learning methods. The advantage of these computational approaches is that they make fewer assumptions about the functional form of the model and the effects being modeled [16]. In other words, data mining and machine learning methods are much more consistent with the idea of letting the data tell us what the model is rather than forcing the data to fit a preconceived notion of what a good model should be. Neural networks represent one machine learning approach that can complement parametric statistical approaches such as linear regression. [23, 24] introduced a GP approach to evolving neural networks (NN) for genetic analysis where both the architecture and the weights of the NN are optimized. This was later extended to include a grammar for generating NN models using GE [21]. The GENN approach was shown to be more powerful than GPNN for detecting and modeling gene-gene interactions in population-based studies of human disease susceptibility. More recent work has incorporated GENN into a pipeline [10] that includes multiple different data sources and that harnesses the power of feature selection [12, 13] (see also [9, 29]).

[10], who compared grammatical evolution neural networks (GENN) with grammatical evolution symbolic regression (GESR), noted that, "our results suggest that GENN is better at correctly and accurately detecting genetic models with no main effects ... In the simulated meta-dimensional data, Lasso had higher detection power for the full model than both GENN and GESR. However, when we used more powerful parameter settings, GENN was also able to identify the full model consistently ... Lasso is considerably faster than either GENN or GESR, so if computational resources are a major limitation, this may be the optimal method. However, Lasso is not robust to models with no main effects, so the overall benefit of a faster analysis would need to be weighted accordingly ..."

We now briefly review a simple example grammar for generating NN models with GE. The root of the grammar picks a node with a logistic activation function and transfer function with a mathematical function for combining multiple features (addition, subtraction, multiplication, division) along with some inputs that could be additional nodes and/or features with weights. The GE operates by generating an array of bits where each set of bits encodes and integer value that is used to execute the grammar. For example, an array of bits yielding integers [0,1,1,2] would generate a NN with a single node with a logistic activation function, a subtraction transfer function, and

a single input of feature number three modified by a randomly generated weight. A slightly more complex NN example that could be generated from this grammar with the right integer set is shown in Figure 1.

```
<root> ::= <node> <input>
<node> ::= <activation> <transfer>
<input> ::= <input> <input>          0
          | <feature> <weight>       1
          | <node> <input>           2
<activation> ::= logistic            0
          | linear                   1
<transfer> ::= addition              0
          | subtraction              1
          | multiplication           2
          | division                 3
<feature> ::= feature 1              0
          | feature 2                1
          | feature 3                2
<weight> ::= random number
```

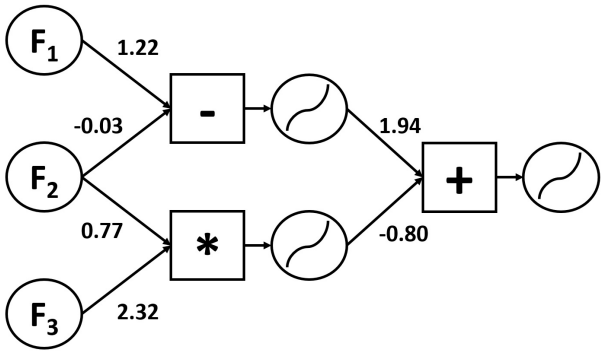

Fig. 1 A GE-evolved neural network with logistic activation nodes, arithmetic transfer functions, numeric weights, and feature inputs.

Once a grammar is specified a genetic algorithm or any other optimization approach that operates on an array of bits can be applied. Neural networks constructed and optimized in this manner provide tremendous flexibility for modeling complex patterns in big data. A key question is whether these methods could be extended to deep learning or whether smaller networks optimized using GE could approximate the performance of much larger NN.

4 A Grammatical Evolution Approach to Systems Genomics Modeling and Simulation

Understanding how interindividual differences in DNA sequences map onto interindividual differences in phenotypes is a central focus of human genetics. Genotypes contribute to the expression of phenotypes through a hierarchical network of biochemical, metabolic, and physiological systems. The availability of biological information at all levels in the hierarchical mapping between genotype and phenotype has given rise to a new field called systems biology. One goal of systems biology is to develop a bioinformatics framework for integrating multiple levels of biological information through the development of theory and tools that can be used for mathematical modeling and simulation [11]. The promise of both human genetics and systems biology is improved human health through the improvement of disease diagnosis, prevention, and treatment. We illustrate here the use of GE to discover and optimize Petri net models of discrete dynamical systems.

Petri nets are a type of directed graph that can be used to model discrete dynamical systems [2]. [5] 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 [5], places represent molecular species. Each place has a certain number of tokens that represent the number of molecules for that particular molecular species. 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 (or bandwidths), 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 supply 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 firing rate of the transition can be immediate, delayed deterministically, or delayed 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.

[19, 20] developed a BNF grammar for Petri nets in BNF. 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 the production rule that was executed to begin the model building process for the study by [20].

```
<root> ::= <pick_a_gene> <pick_a_gene> <pick_a_gene>
<net_iterations> <expr> <transition> <transition> <place_noarc>
```

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.

```
<expr> ::= <expr> <expr>  0
         | <arc>          1
         | <transition>   2
         | <place>        3
```

Here, the selection of one of the four nonterminals (0, 1, 2, or 3) on the right-hand side of the production rule is determined by a combination of bits in the genetic algorithm.

The base or minimum Petri net that is constructed using the <root> production rule consists of a single place, two transitions, and an arc that connects each transition to the place. 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 define 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.

Figure 2 shows an example Petri net constructed by [20]. This model was evolved using GE to map genotypic variation across different genes to disease susceptibility determined by levels of protein product. Here, the GE evolved different arcs (A) connecting transitions (T) to molecular species (P) to be dependent on different genotypic values at a particular gene. Thus, the GE was able to evolve both the structure of the network and the parameter settings to reach some target behavior.

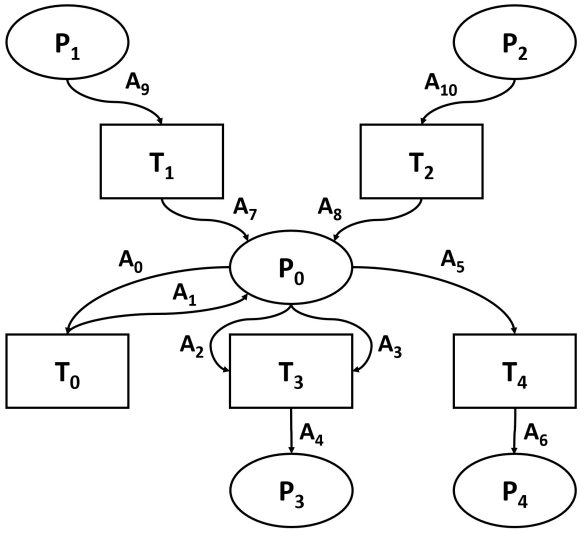

Fig. 2 A GE-evolved Petri net with different arcs (A) connecting transitions (T) to molecular species or places (P). Each arc, transition, and place has several different parameters evolved by the GE that govern its behavior.

5 The Future of Grammatical Evolution Approaches to Bioinformatics and Systems Genomics

The potential impact of evolutionary computation in the biological and biomedical sciences is enormous [27]. Grammatical evolution has a place in this future given its flexible grammar-based method for representing solutions to complex problems. We list here several computational challenges that will need to be addressed for application of GE to biological problems. We then end with some of the hot new biological problems that GE might be useful for.

The most important challenge of using GE or other similar approaches is the inherent complexity of biological systems. Biological systems are driven by molecular, physiological, anatomical, environmental, and social interactions. Layer on top of this big data from technologies such as high-throughput DNA sequencing and the modeling challenges become manyfold more significant. No computational modeling approach is immune to these challenges. Here are a few research topics that will need to be explored in the coming years. First, what is the best way to adapt GE to handle diverse data types coming from different sources and technologies? [12, 13] have started to address this with the GENN system described above. Second, what is the best way to integrate expert knowledge into GE to help identify and exploit good building blocks? This is important to provide the GE with some direction in an effectively infinite search space. Fortunately, there are many sources of expert

knowledge for biological systems including literature sources such as PubMed and biological knowledge bases such as the hetionet database that integrates 29 different sources of information about genes, diseases, drugs, pathways, anatomies, processes, etc. [8]. Third, what is the best way to parallelize GE for use in cluster or cloud computing technology? Fourth, what is the best way to store GE results and to create knowledge from those results that can in turn be used by the GE in future runs? Fifth, what is the best way to perform multiobjective optimization? This is important for biological problems where there are often multiple fitness objectives. For example, using GE to identify genetic risk factor for disease could benefit from rewarding models for the drugability of the genes it is finding in addition to measures such as classification accuracy. This helps the GE reward models with genes that are actionable in addition to being predictive. Finally, what is the best way to interpret GE models and results? This is perhaps the most important challenge because at the end of the day biologists want actionable results. They want to be able to learn something from a GE result that will make it easy for them to design a validation experiment. This is not easy and is an area where many machine learning and artificial intelligence efforts fall short. If we want to solve the world's most complex problems, we need to keep in mind the ability to derive impact from those solutions. This is something the deep learning community is struggling with.

Regarding the interpretability issue it is worth mentioning the work of [30]. They developed a system dubbed *G-PEA* (GP Post-Evolutionary Analysis), for use with tree-based GP. First, one defines a functionality-based similarity score between expressions, which G-PEA uses to find subtrees that carry out similar semantic *tasks*. Then, the system clusters similar sub-expressions from a number of independently-evolved fit solutions, thus identifying important *semantic building blocks* ensconced within the hard-to-read GP trees. These blocks help identify the important parts of the evolved solutions and are a crucial step in understanding how they work. Though developed within the context of tree-based GP, ideas from G-PEA may well transfer to GE.

An emergent, important theme in artificial intelligence is that of usability and accessibility to a person not versed with machine learning. Towards this end [22] have developed PennAI, an accessible artificial intelligence whose ultimate goal is to deliver an open-source, user-friendly AI system that is specialized for machine learning analysis of complex data in the biomedical and healthcare domains. It would be interesting to examine the use of GE within the context of PennAI.

The biological and biomedical sciences are changing rapidly. We highlight here a few hot areas where GE could be focused in the coming years. First, cell biology and genomics continues to be transformed by high-throughput technologies such as DNA sequencing, mass spectrometry, and imaging. Each of these technologies generates massive amounts of data about different molecules and cellular processes. A central challenge in bioinformatics is the integration of these data to facilitate new scientific questions. Understanding

how different molecular and cellular levels interact to influence a biological process or outcome is a place where grammatical evolution can have a significant impact given its inherent flexibility for program or solution representation. Second, mobile devices and remote sensors are starting to have a big impact on the biological and biomedical sciences. Remote sensors can track animals and plants in ecological settings while wearable devices can measure physiology and behavior of human subjects in their natural environment. These new technologies generate massive amounts of heterogeneous data that often have a time component adding an additional dimension of complexity. This is an area that could greatly benefit from GE. Finally, electronic health records (EHR) have exploded over the last several years for capturing, storing, integrating, and managing health data. There is an unprecedented opportunity to develop and apply methods such as GE for identifying patterns of health measures that are predictive of disease outcomes and drug response, for example. This is an emerging area that needs machine learning and artificial intelligence strategies for improving health and healthcare. An example application is the use of GE for real-time monitoring of patient data synced with clinical decision support systems that can provide instantaneous alerts to clinicians about patient characteristics that are urgent. Some of the technical challenges mentioned above will need to be solved for GE use in these domains to become a reality.

References

1. Alfonseca, M., Gil, F.J.S.: Evolving an ecology of mathematical expressions with grammatical evolution. Biosystems **111**(2), 111–119 (2013)
2. Desel, J., Juhás, G.: "What is a Petri net?" Informal answers for the informed reader. In: H. Ehrig, J. Padberg, G. Juhás, G. Rozenberg (eds.) Unifying Petri Nets: Advances in Petri Nets, pp. 1–25. Springer Berlin Heidelberg, Berlin, Heidelberg (2001)
3. Fenton, M., Lynch, D., Kucera, S., Claussen, H., O'Neill, M.: Multilayer optimization of heterogeneous networks using grammatical genetic programming. IEEE Transactions on Cybernetics **47**(9) (2017)
4. Ferrante, E., Turgut, A.E., Duéñez-Guzmán, E., Dorigo, M., Wenseleers, T.: Evolution of self-organized task specialization in robot swarms. PLoS computational biology **11**(8), e1004,273 (2015)
5. Goss, P.J., Peccoud, J.: Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. Proceedings of the National Academy of Sciences **95**(12), 6750–6755 (1998)
6. Greene, C.S., Tan, J., Ung, M., Moore, J.H., Cheng, C.: Big data bioinformatics. Journal of cellular physiology **229**(12), 1896–1900 (2014)
7. Hidalgo, J.I., Colmenar, J.M., Kronberger, G., Winkler, S.M., Garnica, O., Lanchares, J.: Data based prediction of blood glucose concentrations using evolutionary methods. Journal of Medical Systems **41**(9), 142 (2017)
8. Himmelstein, D.S., Lizee, A., Hessler, C., Brueggeman, L., Chen, S.L., Hadley, D., Green, A., Khankhanian, P., Baranzini, S.E.: Systematic integration of biomedical knowledge prioritizes drugs for repurposing. eLife (2017)

9. Holzinger, E.R., Buchanan, C.C., Dudek, S.M., Torstenson, E.C., Turner, S.D., Ritchie, M.D.: Initialization parameter sweep in ATHENA: optimizing neural networks for detecting gene-gene interactions in the presence of small main effects. In: Proceedings of the 12th annual conference on Genetic and evolutionary computation, pp. 203–210. ACM (2010)
10. Holzinger, E.R., Dudek, S.M., Frase, A.T., Pendergrass, S.A., Ritchie, M.D.: ATHENA: the analysis tool for heritable and environmental network associations. Bioinformatics **30**(5), 698–705 (2013)
11. Ideker, T., Galitski, T., Hood, L.: A new approach to decoding life: systems biology. Annual review of genomics and human genetics **2**(1), 343–372 (2001)
12. Kim, D., Li, R., Dudek, S.M., Frase, A.T., Pendergrass, S.A., Ritchie, M.D.: Knowledge-driven genomic interactions: an application in ovarian cancer. BioData mining **7**(1), 20 (2014)
13. Kim, D., Li, R., Dudek, S.M., Ritchie, M.D.: ATHENA: Identifying interactions between different levels of genomic data associated with cancer clinical outcomes using grammatical evolution neural network. BioData mining **6**(1), 23 (2013)
14. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection, vol. 1. MIT press (1992)
15. Marcotty, M., Ledgard, H.: The World of Programming Languages. Springer-Verlag, Berlin (1986)
16. McKinney, B.A., Reif, D.M., Ritchie, M.D., Moore, J.H.: Machine learning for detecting gene-gene interactions. Applied bioinformatics **5**(2), 77–88 (2006)
17. Moore, J.H.: Bioinformatics. Journal of Cellular Physiology **213**(2), 365–369 (2007). DOI 10.1002/jcp.21218. URL http://dx.doi.org/10.1002/jcp.21218
18. Moore, J.H., Asselbergs, F.W., Williams, S.M.: Bioinformatics challenges for genome-wide association studies. Bioinformatics **26**(4), 445–455 (2010)
19. Moore, J.H., Hahn, L.W.: Petri net modeling of high-order genetic systems using grammatical evolution. BioSystems **72**(1), 177–186 (2003)
20. Moore, J.H., Hahn, L.W.: An improved grammatical evolution strategy for hierarchical Petri net modeling of complex genetic systems. In: EvoWorkshops, pp. 63–72. Springer (2004)
21. Motsinger-Reif, A.A., Dudek, S.M., Hahn, L.W., Ritchie, M.D.: Comparison of approaches for machine-learning optimization of neural networks for detecting gene-gene interactions in genetic epidemiology. Genetic epidemiology **32**(4), 325–340 (2008)
22. Olson, R.S., Sipper, M., La Cava, W., Tartarone, S., Vitale, S., Fu, W., Holmes, J.H., Moore, J.H.: A system for accessible artificial intelligence. In: Genetic Programming Theory & Practice XV. Springer (2017). URL https://arxiv.org/abs/1705.00594. (to appear)
23. Ritchie, M.D., Motsinger, A.A., Bush, W.S., Coffey, C.S., Moore, J.H.: Genetic programming neural networks: A powerful bioinformatics tool for human genetics. Applied Soft Computing **7**(1), 471–479 (2007)
24. Ritchie, M.D., White, B.C., Parker, J.S., Hahn, L.W., Moore, J.H.: Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases. BMC bioinformatics **4**(1), 28 (2003)
25. Ryan, C., Collins, J.J., O'Neill, M.: Grammatical evolution: Evolving programs for an arbitrary language. In: Genetic Programming, First European Workshop, EuroGP'98, Paris, France, April 14-15, 1998, Proceedings, pp. 83–96 (1998). DOI 10.1007/BFb0055930. URL https://doi.org/10.1007/BFb0055930
26. Sipper, M.: Machine Nature: The Coming Age of Bio-Inspired Computing. McGraw-Hill, New York (2002)
27. Sipper, M., Olson, R.S., Moore, J.H.: Evolutionary computation: the next major transition of artificial intelligence? BioData Mining **10**(1), 26 (2017). DOI 10.1186/s13040-017-0147-3. URL https://doi.org/10.1186/s13040-017-0147-3

28. Smart, O., Tsoulos, I.G., Gavrilis, D., Georgoulas, G.: Grammatical evolution for features of epileptic oscillations in clinical intracranial electroencephalograms. Expert systems with applications **38**(8), 9991–9999 (2011)
29. Turner, S.D., Dudek, S.M., Ritchie, M.D.: ATHENA: A knowledge-based hybrid backpropagation-grammatical evolution neural network algorithm for discovering epistasis among quantitative trait loci. BioData mining **3**(1), 5 (2010)
30. Wolfson, K., Zakov, S., Sipper, M., Ziv-Ukelson, M.: Have your spaghetti and eat it too: Evolutionary algorithmics and post-evolutionary analysis. Genetic Programming and Evolvable Machines **12**(2), 121–160 (2011)