HHMI Author Manuscript

HHMI Author Manuscript

HHMI Author Manuscript

PubMed Central

HHMI
HOWARD HUGHES MEDICAL INSTITUTE

AUTHOR MANUSCRIPT
Accepted for publication in a peer-reviewed journal

Published as: *Int J Biochem Cell Biol*. 2009 February ; 41(2): 405–413.

Applications of Genetic Programming in Cancer Research

William P. Worzel[1], **Jianjun Yu**[2,4], **Arpit A. Almal**[1], and **Arul M. Chinnaiyan**[2,3,4]

[1]Genetics Squared Inc., 401 W. Morgan Rd., Ann Arbor, MI 48108, USA

[2]Department of Pathology, University of Michigan Medical School, Ann Arbor, MI 48109, USA

[3]Department of Urology, University of Michigan Medical School, Ann Arbor, MI 48109, USA

[4]The Comprehensive Cancer Center, University of Michigan Medical School, Ann Arbor, MI 48109, USA

Abstract

The theory of Darwinian evolution is the fundamental keystones of modern biology. Late in the last century, computer scientists began adapting its principles, in particular natural selection, to complex computational challenges, leading to the emergence of evolutionary algorithms. The conceptual model of selective pressure and recombination in evolutionary algorithms allows scientists to efficiently search high dimensional space for solutions to complex problems. In the last decade, genetic programming has been developed and extensively applied for analysis of molecular data to classify cancer subtypes and characterize the mechanisms of cancer pathogenesis and development. This article reviews current successes using genetic programming and discusses its potential impact in cancer research and treatment in the near future.

Keywords

genetic programming; evolutionary algorithms; cancer diagnosis; cancer classification; cancer prognosis

BACKGROUND

Darwin's fundamental insight into the nature of evolution has been one of the cornerstones of modern biology. The diversity of species and individuals in the fossil record shows the evolution of complex features, not once, but many times, demonstrating that evolution is a powerful, creative force. Starting in the late 1950s, computer scientists began using the concepts of evolution and natural selection to find creative solutions for computing problems (Fraser 1957, Barricelli 1957, Friedberg 1958, Bremermann 1962). The key ideas of these algorithms were the creation of a population of solutions, with some method of varying the composition of the individuals in the population based on a measure of fitness, or how well the individuals perform on a particular task. The most fit individuals created

© 2008 Elsevier Ltd. All rights reserved.

Corresponding author: Arul M. Chinnaiyan, M.D., Ph.D., Howard Hughes Medical Institute, Department of Pathology and Urology, University of Michigan Medical School, 1500 E. Medical Center Drive, 5410 CCGC, Ann Arbor, MI-48109, Phone: 734-615-4062, Fax: 734-615-4055, arul@umich.edu.

Publisher's Disclaimer: This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final citable form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

would replace less fit individuals so that over time the overall fitness of the population would increase.

The broad adoption of genetic algorithms, where a population of individual parameter values are encoded as strings of bits, was introduced in (Holland 1962), and has been used since many different disciplines including engineering design (Selig 1996), economic modeling (Arifovic 1994), and genetics (Hill 2005). A key component of genetic algorithms is the emphasis on crossover or recombination of individual solutions in a population rather than small, incremental mutations. Recombination takes the bit strings on individuals in a populations and swaps large pieces of the bit stings that make up the "genetic material" with other individuals to create offspring solutions. While this was not a new idea, the emphasis on crossover as a key component of the process, the representation of complex structures as bit strings mapped to complex components, and Holland's development of a theory describing the effect of crossover put genetic algorithms at the forefront of evolutionary computing.

Shortly thereafter, a new effort to "evolve" computer programs began. In essence, this was an effort to create computer programs whose outputs were computer programs that solved a problem. This may be contrasted with earlier evolutionary algorithms that were mainly focused on optimizing numerical solutions by finding values for computational models that produced the best results. So, while a genetic algorithm might be used to find the optimal set of pressures and mix of reactants in a chemical reaction, these new algorithms would try to find programs that could create computer models of chemical reactions learned from a set of inputs and the resultant products. The fitness measure for such a process would be measured by how accurately the program would predict the resulting products across a number of starting conditions.

(Rechenberg 1965) created evolution strategies that started with a population of computer programs, altered or "mutated" individual programs in the population and tested whether the mutated program was better than its parent by using a fitness measure. If it was more fit, then the mutated program would replace its parent in the population. (Fogel 1966) described a system of evolutionary programming which mutated individuals within a population of programs and put them into competition with the other members of the population, not just their parents. By the 1980s, more complex method were proposed for evolving computer programs including Forsyth (1981) and (Cramer 1985).

The most widely used of these systems was proposed by John Koza (1989) and is called genetic programming. Genetic programming (GP) uses a fixed representation for programs and, based on this representation, created a recombination operation that could combine programs to create offspring programs. This followed the pattern established in genetic algorithms and meant that offspring programs could combine useful features of their parents. In this case the "building blocks" exchanged during crossover were function pieces of the programs. The flexibility of the representation proposed by Koza has made GP the most common form used to evolve computer programs.

Figure 1 shows the representation used by GP for simple programs and the method of recombining the programs to programs to produce offspring programs. In 1a, the simple function A/B > C is a Boolean function that produces a TRUE/FALSE result when given input values for the variables A, B and C. It is represented as a tree structure where individual variables, or "terminals" are combined with mathematical functions. This is a standard representation used internally by computer compilers to transform high level languages like C or C++ to the machine code in executable programs.

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

Figure 1c shows how two such programs may be combined to produce offspring programs during crossover. Here the program (A*B) > (C+D) is combined with the program G < (E/F) by swapping parts of the program trees to produce the offspring programs (A*B) > (E/F) and G < (C+D). While not shown, mutation is achieved by stochastically selecting a portion of the tree and randomly replacing it. For example, mutation might change (E/F) in one of the offspring programs to (H–I) after the crossover is completed. It is important to note that both crossover and mutation operations are stochastic in nature with a probability assigned by the control parameters of the GP process. Typical values of crossover and mutation rates are 10%–50% for crossover and 1%–5% for mutation.

Because GP usually operates in a supervised learning mode, a training set of data is necessary with known outcomes that can be used to test the fitness of each program. For example, microarray expression data of tumor tissues and patient outcome information could be used to develop a cancer prognostic program that could be used to predict which patients are likely have a post-operative recurrence. The GP system would “learn” by example, using the evolutionary process to search for and find a combination of mathematical functions that can model recurrence features.

Figure 1b is a flowchart that shows how the evolutionary process works. The initial step is to create a population of program trees that are randomly assembled from the available inputs in an unbiased way. This allows a broad sampling of possible inputs and begins the search for productive combinations. Though the initial population will probably have a poor performance overall, by random chance, some individuals will perform better than others because of the features selected, or the way they are combined mathematically.

Once the initial population is created, a small number of programs are randomly selected from the population and applied to the training data, producing outputs that can be compared with the true results. In the prognostic example above, this small sub-population of selected programs make a prediction about each patient in the training set as to whether he or she will have a recurrence within the bounding timeframe. The results of these predictions will be used to calculate a measure of effectiveness, or fitness, for each individual in this group. The fitness measure is used to select the two best individuals from this sub-population and by combining elements from each program, produces new “offspring” programs by applying the crossover operation shown in Figure 1c with a chance of mutation as described above. The generated offspring programs then replace the two least fit individuals in the population. Each of these operations has a separate probability of being applied to the mating pair after selection, which creates a stochastic element in the evolutionary process.

This cycle of sub-group selection, fitness testing, recombination, mutation and replacement is repeated until a halting condition is reached. An example of such a criteria might be a pre-determined level of prediction accuracy or a number of replacements equal to the number of individuals in the main population (a “generation”). While the initial, randomly created programs of the population are very poor in terms of performance, the internal cycle of evolution quickly improves the fitness of the programs in the population, and usually focuses on a small combination of input features that are most useful in solving the problem.

Almal et al (2008) describes a surprising similarity in behavior between genetic programming and natural evolution despite the significant differences in genetic representation and evolutionary mechanisms. This hints at the power of the basic, underlying mechanisms of evolution, regardless of the medium in which it operates.

From an analytic point of view, some of the desirable features of genetic programming include:

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

- The ability to control the bias incorporated into feature selection – this can be varied from a completely unbiased approach in feature selection to a partial bias that uses prior knowledge or even a paradigm that imposes complete constraint on feature selection.
- The ability to integrate diverse data to produce mixed models of the data such as integrating demographic, clinical and molecular data
- The ability to create human readable results in contrast to "black box" solutions
- The ability to create solutions whose form is not constrained; it is equally likely to create linear or non-linear models from an analysis.

While genetic programming is not the only analytic approach to have many of these features, but it is almost unique in providing all of them in a single system. Because of these features, genetic programming has a rich history of applications. It has been used for novel design such as patented antenna designs (Lohn et al 2004), and patented analog electronic circuits (Koza et al 2004), and small molecule design (Nachbar 2000). It is also being used commercially to characterize dynamic processes such as chemical processes (Hinchliffe and Willis 2003), financial markets (Becker et al, 2006) and image processing (Zhang 2007).

Given that genetic programming is one among many possible options, both from machine learning and from analyzing data, one must ask the question of where it fits into the possible choices for analyzing data. One way to view this is suggested by (Vadislavleva 2008) who points out that the choice of analytic tool is in part driven by your knowledge about the likely solution. In this assessment, there are usually two main issues: whether the key factors in the solution are known, and whether the structure of the solution is known. For example, if the key factors of a problem are known, and it is clear that a linear solution is likely to work, the linear regression is probably the best choice: it is computationally inexpensive, reliable and interpretable. Similarly, if a non-linear solution is likely but the factors are known, then non-linear regression analysis is probably the best choice.

However, if the key features are known, but are large in number or the form of the solution is unknown, then machine learning approaches such as artificial neural networks, support vector machines (SVMs), or fuzzy rule systems would be a better fit. This is because such algorithms can flexibly assemble a model from a fairly limited and known set of variables, "learning" by testable example such as how well they predict an outcome. If the key features are not known, and the number of features are too large to provably identify the key features, than these techniques will usually require a combined approach such as selecting features using statistical univariate or multivariate analysis to select likely features or hybrid approaches such as combining a genetic algorithm with a neural network to find and build a usable model (Yao 1999).

Genetic programming has the advantage of extreme flexibility, but at a high computational cost. It can model systems where the key features are not known (though there are upper limits on the number of features it can sift through (Moore 2006)) and can model systems where the structure and complexity of the desired models are not known (Kordon 2002). With GP, you can use whole genome chip data sets as inputs, and allow the system to search for models from a range of possible model structures.

However, it is important to realize that while the selection pressure toward better solutions that drives the evolutionary process tends to produce better and better solutions, it can never be said to be an optimal solution unless it clearly meets all desired criteria. This is because the flexibility that allows it to search a very large space of possible solutions, cannot guarantee that it has found the best solution. It cannot make an exhaustive search of all

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

possible programs of greater than a trivial size so the stochastic nature of the algorithm samples the space while the fitness and selection pressure it exercises focuses on preferred areas of the space. Nevertheless, we can seldom guarantee that there is not a better solution "just around the corner" we merely have a solution that is "good enough" for our purposes. Therefore it cannot be viewed as an optimization algorithm, but rather as a powerful search algorithm that can discover novel and useful solutions to problems.

However, GP has been used as a way to discover both the key features and the structure of a model that is a good approximation of a first principles model. This insight allows first principle modelers to arrive at a solution more quickly than would otherwise be possible. (Kordon 2002) estimates that using GP to develop an approximate model based on empirical data reduces the time it takes to develop a first principles model for chemical processes from a year to 3–4 months, a significant saving of time and resources. In the context of biology, such models may suggest interactions that can be used in a "systems biology" approach to create better models of disease or disease processes.

CANCER TREATMENT AND RESEARCH

Genetic programming has been used in a number of areas of cancer research as well as clinical treatment. The advent of modern molecular techniques such as microarrays, proteomics and single nucleotide polymorphism arrays (SNP chips) have provided a large amount of interrelated data that is not easily modeled using conventional data analysis methods. Genetic programming has been used alone, and in combination with other methods to produce useful and informative analyses. Some of these applications are described in the following section.

Genomic Studies

A number of studies have been published on the use of genetic programming to classify tumors on the basis of genomic signatures (Gilbert et al 2000, Moore and Parker 2002, Driscoll et al 2003, Hong and Cho 2004, Mitra et al 2006, Yu et al 2007). Because of GP's unbiased feature selection and ability to mathematically combine expression levels, the classification models produced are quite parsimonious and able to capture maximum information from a small number of features. Moreover, as these models are often easy to interpret, they are often informative of cellular and disease dynamics, pathway deregulations as well as epigenetic effects.

For example, in (Mitra 2006), 70 genes were profiled using qRT-PCR on 65 bladder samples. A training sub-cohort of 34 samples was used to generate a predictive classification rule for the presence of nodal metastases using the primary bladder cancer samples. From this data a simple 3-gene mathematical signature was discovered for predicting nodal metastases from primary tumor tissue samples. This signature took the form of a mathematical expression of three genes to produce a single value that classified the patients based on an identified slice-point (Equation 1) that separated training samples into those with associated nodal metastases and those without metastases. The prediction of this signature on the remaining 31 samples yielded a positive predictive value of 100% and a negative predictive value of 88%.

None of the 3-genes in the predictive rule stood out in terms of up or down regulation –their expression levels were quite varied (Figure 2B). However, an examination across multiple, high performing rules in the evolved population revealed a consistent pattern of ICAM1 being up-regulated *relative* to MAP2K6 which was subsequently up-regulated *relative* to KDR. Moreover, ICAM1 and MAP2K6 are both in the ICAM1 pathway, which has been reported as being associated with cancer progression, while KDR has been reported as being

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

associated with the vascularization supporting tumors (Hubbard and Rothlein 2000). This demonstrates both the ability of genetic programming to find and quantify relationships as well as discovering novel interactions.

In a recent study (Yu 2007), we demonstrated that GP classifiers could be validated on independent data sets. In the first instance, two independent prostate cancer genomic data sets profiled on Affymetrix U95A chips at different institutes were selected and gene-wise standardization was performed in both datasets to adjust dataset-specific measurement differences. Samples in the first set were used to produce a set of predictive rules each of which used only two genes and perfectly classified metastatic prostate cancer from primary prostate cancer. These GP rules were then tested on the 2nd data set and the best two rules yielded perfect sensitivity and 93.5% specificity on the testing set.

As different microarray platforms may be used at individual laboratories, it is worth noting that GP rules may not predict well on sample sets profiled by different platforms due to discrepancy on probe design, genome mapping, and array schema etc. A simple gene-wise standardization may not be capable of capturing such profound variations. However, the mathematical expression of genes included in GP rules may still be predictive across datasets. For example, sample classification based on a rule of "A + B > C" may perform poorly in another data set using different platform, but the expression of "A + B – C" and its resulting value may remain predictive across datasets where the performance can be estimated by Area Under the ROC Curve statistic (ROC-AUC). As an example, we collected 6 prostate cancer data sets and used two-thirds of samples in the first set to produce GP rules to classify primary prostate cancer from benign samples. Three perfect GP rules were generated from the training samples and applied to the rest samples in the 1st set and other 5 data sets. While the GP rules performed well on the remaining samples in the first dataset, testing on the other 5 datasets showed poor prediction as expected. However, ROC-AUC tests revealed that all the three rules yielded high AUC statistics in each test dataset (Figure 3), supporting our hypothesis that the mathematic relation between genes holds true across datasets.

The study also demonstrated that, while genetic programming uses significantly fewer genes than other classification methods, it produces comparable performance in terms of prediction accuracy. This suggests that genetic programming may be particularly useful in developing clinical tests since fewer biomarkers are required so clinical costs would be reduced. Work is underway to use GP to develop signatures for cancer diagnosis and recurrence and it is expected that such work will have an impact on clinical practice in the future.

Recently (Langdon 2008) published on the development of a predictive classifier for breast cancer survival from genomic data generated from the Affymetrix U133 chip set from the GEO GSE3494 dataset. It describes the development of a 3-gene classifier using 91 samples with a verification sample set of 90 samples. The classifier had an accuracy of 70% on the verification set and 78% accuracy on the entire data set. The Kaplan-Meier plot for this result showed a distinct improvement on the 32-gene predictor developed for the same data set and described in (Miller 2005).

This paper also highlighted an important feature of genetic programming. Since GP works over a population of possible programs, it is naturally quite easy to segregate the population into sub-populations that evolve in parallel to one another. This allows large populations to be evolved in a relatively short amount of time, reducing the computational cost of GP. Moreover, by establishing independent populations, or demes, different regions of program space may be searched in each population, and by periodically trading individuals, useful

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

components may be combined in the same way a valuable evolutionary trait may be fixed in a small population before spreading to a larger population in natural evolution.

While the natural parallelism of GP has been known for some time, (Langdon 2008) takes advantage of the rise of high-speed, compact graphics processing units (GPUs) to demonstrate that these processors can effectively be used to expand the computational resources available for GP.

Genetic Studies

Genetic Programming has also been used to analyze single nucleotide polymorphism (SNP) data. Nunkesser et al (2007) describes the use of GP to analyze limited SNP data from the GENICA study (GENICA 1999). Moore et al (2007) has described a system designed to work on full scale “SNP-chip” data.

The study (Nunkesser et al 2007) is particularly interesting as the authors applied genetic programming to 63 SNPs in a breast cancer study and identified a Boolean rule that includes a known interaction between two SNPs. Notably, introducing several additional SNPs to the rule developed, produced a more powerful predictor. Such findings would be difficult to discover using standard association analysis techniques. The ability to combine SNPs seamlessly makes genetic programming a useful tool in association studies and also for suggesting the interplay between mutations.

Moore and White (2006) not only used GP to integrate data to produce a predictive model, but it also highlighted the importance of integrating expert knowledge into the process by using a modified version of the ReliefF algorithm (Robnik-Sikonja and Kononenko 2003) to distinguish the quality of feature combinations. Moore and White comment that while they used a statistical measure of values based on association with results under study (such as data associated with susceptibility to a disease) to select significant features, the ReliefF algorithm could be used to select features based on associations in published literature.

This approach is integrated into a more complex algorithm in (Moore et al 2007) where a number of GP runs are made to sample the data and then a statistical analysis identifies useful combinations that are discovered. These combinations are assessed and a functional mapping of the best rules assesses the contribution each feature makes to the predictive power of the rule. While this study has a limited number of mutational features, it is designed to be applied to larger SNP sets and the authors have indicated that this is approach is being applied to a multi-factorial study of cancer susceptibility that includes a large amount of SNP data (private communication).

Radiological Data Analysis

In (Sheta 2005), GP was used to analyze mammography data taken from the Digital Database of Screening Mammography (DDMA) at the University of Southern Florida. They took 1,336 mammography image Regions of Interest (ROIs) from the database for their analysis. 656 were randomly selected images of normal breast parenchyma, 341 were benign masses, and 340 malignant tumors. After preprocessing the images to identify linear extents of the structures, a number of features of these processed images were extracted such as length, orientation, area, etc. They divided the data for these samples into a training set of ¾ of the samples and a validation set of ¼ of the samples.

The training set sample data was used by the GP system to evolve a classifier that differentiated between malignant and non-malignant tumors. The best result from the GP analysis had an accuracy of 86.2% on the training set and an accuracy of 85.7% on the validation set. This classifier also had an Area-Under-the-Curve (AUC) of 0.91 of the

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

receiver operator curve (ROC) on the entire set which compared favorably with a backpropogation artificial neural network (BP-ANN) which had an AUC of 0.88. As the authors noted, the GP result had a statistically better result than the ANN and also had the advantage of being a more comprehensible classifier.

In (Dekker 2005), genetic programming is used to analyze MALDI-TOF mass spectrometry data to identify a group of protein clusters that predict leptomeningeal metastasis in breast cancer patients from spectral analysis of cerebrospinal fluid in 106 patients. 54 breast cancer patients in the cohort were diagnosed with leptomeningeal metastasis, while 52 breast cancer patients were not diagnosed with leptomeningeal metastasis. The results showed that the GP developed classifier had a sensitivity of 79%, a specificity of 74% and an overall accuracy of 77%. In this case GP was used to select the best cluster of protein groups to create a model that accurately predicted leptomeningeal metastases.

(Gray 1998) describes the use of genetic programming to analyze nuclear magnetic resonance (NMR) spectral data from brain tumor samples. 75 surgical biopsy samples were profiled using NMR and digitized and normalized. Of these 75 samples, 28 were meningiomas and the others were a variety of other tumor types. The spectral peaks were digitized, normalized and principal component analysis reduced the number of spectral features considered from 400 to 20. Analysis using GP produced a simple classifier that was 97% accurate in classifying all samples as being either meningioma or not. To test the generality of a solution produced from this data, they re-ran the evolution reserving 10 samples from the original set of 75 samples and a simple classifier was produced that had a 90% classification accuracy on both the training and the test set. The authors again note the simplicity of the solution, using only 3 features in a very comprehensible result that involved several plausible metabolites. The GP results were also compared to a BP-ANN and had about the same level of accuracy with the advantage of being more accessible to the researchers.

Other Cancer Related Data Analyses

While GP tends to be applied to studies related to data that is either difficult to interpret, or has other features such as high dimensionality that make it hard to work with, it has also been used successfully to analyze more common pathology data.

In (Bojarczuk 2004), a number of datasets were analyzed using genetic programming, including two breast cancer datasets and an adrenocortical cancer dataset. The first breast cancer dataset is the Wisconsin breast cancer dataset described in the UC Irvine Machine Learning Data Repository (MLDR). It is a set of data derived from cellular images from a fine needle aspirate from the breast mass. Each sample has a set of cell nucleus measurements such as radius, texture, area, etc, and a diagnosis of benign or malignant for each tumor. There is data for a total of 683 samples in the database,. Using GP the authors obtained an accuracy of 93.5% over the entire dataset which was slightly worse than the C4.5 decision tree algorithm that it was compared with, though still within the margin of error.

The second breast cancer dataset, the Ljubljana breast cancer dataset from the same repository, has 277 records that were analyzed using GP. The data in these records included demographic data such as age range, menopausal status and pathology data such as size, location and stage and treatment information as to whether they were treated with radiatiotherapy or not. The goal was to predict recurrence from the nine attributes available. Here the GP classifier had an overall accuracy of 71.8% - virtual identical to the decision tree system.

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

The adrenocortical cancer dataset focused on predicting survival based on a combination of demographic, pathology, and histological features. While the dataset had 124 cases, the authors analyzed the dataset on the basis of survival time with three different endpoints: survival of more than one year, survival of between one and two years and survival between two and five years. However, these endpoints produce an unbalanced dataset with 22 target cases and 84 negative cases in the first analysis, 8 positive case and 83 negative cases in the second analysis and 6 positive cases and 62 negative cases in the third analysis. For this reason, sensitivity and specificity are a better measure of success than accuracy in order to avoid being mislead by an accuracy figure that only has useful results for the numerically larger negative cases.

Given this, the study of one-year survival produced a balanced classifier with a sensitivity of 0.79 and a specificity of 0.725. The study of 1–2 year survival range as the target class produced a classifier with a sensitivity of 0.9 and a specificity of 0.781. While the third study of 3–5 year survival as the target class produced a classifier with a sensitivity of 0.1 and a specificity of 0.735.

While the first two classifiers reach a level which, if validated in sufficient numbers, would have a clinical significance in terms of informing treatment options, the third classifier is clearly too poor a performer to be informative about patient prognosis.

OTHER APPLICATIONS

In addition to the cancer-specific papers outlined above, GP has applications in other areas that may have an impact on cancer research and treatment in the future. This includes studies on drug discovery and development (Langdon 2008, Archetti et al 2007) and modeling of dynamic processes such as deducing a metabolic pathway from empirical data (Koza et al 2001). Reif et al (2004) demonstrates GP's ability to integrate diverse types of complex molecular data to produce a model based in part on SNP data and in part on mass spectrometry data

Limitations of Genetic Programming

Despite the advantages of GP, there are several limitations in its use. The first is that genetic programming is not an optimization algorithm in the sense of finding the "best" solution to a problem. While the best solution found during a run may in fact be the global optimum, since genetic programming is a stochastic method that is driven only by a relative measure of fitness (i.e., how much fitter one individual program is to another), there is no way to tell whether a given solution is the best possible. Instead, the results must be assessed on the basis of what is needed and what is possible given the available data.

Next, because we are often dealing with situations where the number of inputs is significantly large than the number of samples, the driving power that allows GP to find solutions makes it very easy to find solutions that are overfit to the training data and will not perform well on unseen samples. For this reason we often work to find the most parsimonious solution that meets our halting criteria. This constrains the GP search process to look for the most general solutions within a set of data, which, based on Occam's razor, has a greater chance of being a general purpose solution. We also typically review the solutions for internal consistency in how features are used among the population of solutions (e.g., whether a gene is consistently up-regulated in multiple rules predicting cancer recurrence) and, if possible, that at least some of the features identified make sense based on the published literature.

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

Finally, GP is a computationally intensive process that can require tens or hundreds of thousands of programs to be tested in searching for a desired solution. Even though the basic algorithm is highly parallel by its nature, and can effectively use parallel and grid computing techniques, it may still require more computing power than is readily available.

In summary, GP cannot be considered a “turn-key” solution that can be run on a desktop machine to analyze biological data. Instead, a spirit of trial-and-error and observational science should be brought to the process of applying GP to biological studies. Within these limitations, it can be a powerful addition to current analytic tools for the analysis of biological data that may yield surprising, and potentially insightful results.

CONCLUSION

Genetic programming has a growing record of success in the analysis of large, complex data sets associated with cancer both from a research perspective and from a clinical context. Its abilities to find useful combinations of features from molecular data set in an unbiased way and to produce human comprehensible non-linear models to describe disease states and characteristics make it particularly useful for such analyses.

Though the development of artificial populations competing within a computer to provide solutions to complex problems of molecular and cellular biology seems a long way from the deck of the Beagle and Darwin’s garden, it is a testament to the power and vision embodied in his ideas and his thorough analysis of observed data that evolutionary theory has provided a strong basis for solving problems in the 21st century.

References

Almal, AA.; MacLean, CD.; Worzel, B. Program Structure-Fitness Disconnect and Its Impact On Evolution In GP. In: Riolo, RL.; Soule, T.; Worzel, B., editors. Genetic Programming Theory and Practice V. Santa Clara: Springer; 2008. p. 145-160.

Archetti F, Lansetti S, Messina E, Vanneschi L. Genetic programming for computational pharmacokinetics in drug discovery and development. Genetic Programming and Evolvable Machines. 2007; 8(4):413–432.

Arifovic J. Genetic Algorithm Learning and the Cobweb Model. Journal of Economic Dynamics and Control. Jan; 1994 18(1):3–28.

Barricelli NA. Symbiogenetic evolution processes realized by artificial methods. Methodos. 1957:143–182.

Becker, Y.; Fei, P.; Lester, AM. Stock Selection : An Innovative Application of Genetic Programming Methodology. In: Riolo, RL.; Soule, T.; Worzel, B., editors. Genetic Programming Theory and Practice IV. Santa Clara: Springer; 2007. p. 315-334.

Bojarczuk CC, Lopes HS, Freitas AA, Michalkiewicz EL. A constrained-syntax genetic programming system for discovering classification rules: application to medical data sets. Artificial Intelligence in Medicine. Jan; 2004 30(1):27–48. [PubMed: 14684263]

Bremermann, HJ. Optimization through evolution and recombination. In: Yovits, M.; Jacobi, GT.; Goldstein, GD., editors. Self-Organizing Systems. Spartan Books; Washington D.C: 1962. p. 93-106.

Cramer, NL. A representation for the adaptive generation of simple sequential programs. International Conference on Genetic Algorithms and their Applications (ICGA’85); Pittsburgh. 1985.

Dekker LJ, Boogerd W, Stockhammer G, Dalebout JC, Siccama I, Zheng P, Bonfrer JM, Verschuuren JJ, Jenster G, Verbeek MM, Luider TM, Smitt PA. MALDI-TOF mass spectrometry analysis of cerebrospinal fluid tryptic peptide profiles to diagnose leptomeningeal metastases in patients with breast cancer. Mol Cell Proteomics. 2005 Sep; 4(9):1341–9. Epub 2005 Jun 21. [PubMed: 15970584]

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

Digital Database of Screening Mammography. http://marathon.csee.usf.edu/Mammography/Database.html

Driscoll, JA.; Worzel, B.; MacLean, CD. Classification of Gene Expression Data with Genetic Programming. Riolo, RL.; Worzel, B., editors. Boston: Kluwer; 2003. p. 25-42.

Fogel, LJ.; Owens, AJ.; Walsh, MJ. Artificial Intelligence Through Simulated Evolution. Wiley & Sons, Inc; New York: 1966.

Forsyth R. Beagle: A Darwinian approach to pattern recognition. Kybernetes. 1981; 10:159–166.

Fraser AS. Simulation of genetic systems by automatic digital computers. I. Introduction. Aust J Biol Sci. 1957; 10:484–491. Friedberg.

Friedberg RM. A learning machine: Part 1. IBM Journal of Research and Development. 1958; 2(1):2–13.

GENICA: Interdisciplinary Study Group on Gene Environment Interaction and Breast Cancer In Germay. http://www.genica.de/english.html

Gilbert, RJ.; Rowland, JJ.; Kell, DB. Genomic computing: explanatory modelling for functional genomics. In: Whitley, D.; Goldberg, D.; Cantu-Paz, E.; Spector, L.; Parmee, I.; Beyer, HG., editors. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000). San Francisco: Morgan Kaufmann; 2000. p. 551-557.

Gray HF, Maxwell RJ, Martinez-Perez I, Arus C, Cerdan S. Genetic programming for classification and feature selection: analysis of 1H nuclear magnetic resonance spectra from human brain tumour biopsies. NMR Biomedicine. Jun-Aug;1998 11(4–5):217–224.

Hill T, Lundgren A, Fredriksson R, Schiöth HB. Genetic algorithm for large-scale maximum parsimony phylogenetic analysis of proteins. Biochimica et Biophysica Acta. 2005; 1725:19–29. [PubMed: 15990235]

Hinchliffe, M.; Willis, M.; Tham, M.; Montague, G. Dynamic Chemical Process Modelling Using a Multiple Basis Function Genetic Programming Algorithm. In: Banzhaf, W.; Daida, J.; Eiben, AE.; Garzon, HE.; Honavar, V.; Jakiela, M.; Smith, RE., editors. Proceedings of the Genetic and Evolutionary Computation Conference (v2). Orlando: Morgan Kaufmann; 1999. p. 13-17.

Holland JH. Outline for a logical theory of adaptive systems. J ACM. 1962; 9:279–314.

Hong, JH.; Cho, SB. Lymphoma Cancer Classification Using Genetic Programming with SNR Features. Keijzer, M.; O'Reilly, UM.; Lucas, SM.; Costa, E.; Soule, T., editors. Berlin: Springer-Verlag; 2004. p. 78-88.

Hubbard AK, Rothlein R. Intercellular adhesion molecule-1 (ICAM-1) expression and cell signaling cascades. Free Radic Biol Med. 2000; 28:1379–1386. [PubMed: 10924857]

Kordon A, Pham H, Bosnyak C, Kotanchek M, Smits G. Accelerating Industrial Fundamental Model Building with Symbolic Regression: A Case Study with Structure-Property Relationships. GECCO-2002 Presentations in the Evolutionary Computation in Industry Track. Jul.2002 111–116:11–13.

Koza, JR. Hierarchical genetic algorithms operating on populations of computer programs. In: Sridharan, NS., editor. Proceedings of the 11th International Joint Conference on Artificial Intelligence. San Mateo: Morgan Kaufmann; 1989. p. 768-774.

Koza, JR. Genetic Programming: On the Programming of Computers by Means of Natural Selection. 1. Cambridge: MIT Press; 1992.

Koza, JR.; Mydlowec, W.; Lanza, G.; Yu, J.; Keane, MA. Automated reverse engineering of metabolic pathways from observed data using genetic programming. In: Hiroaki, K., editor. Foundations of Systems Biology. Cambridge: MIT Press; 2001. p. 95-117.

Koza JR, Keane MA, Streeter MJ. Routine automated synthesis of five patented analog circuits using genetic programming. Soft Computing - A Fusion of Foundations, Methodologies and Applications. 2004; 8:318–324.

Langdon, WB. Evolving GeneChip Correlation Predictors on Parallel Graphics Hardware. In: Wang, J., editor. 2008 IEEE World Congress on Computational Intelligence. IEEE Press; Jun 1–6. 2008 p. 4152-4157.

Langdon, WB. Technical Report CES-481. Genetic Programming for Drug Discovery. Computing and Electronic Systems, University of Essex 2008.

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

Lohn, J.; Hornby, G.; Linden, D. Evolutionary Antenna Design for a NASA Spacecraft. In: O'Reilly, UM.; Riolo, RL.; Worzel, B., editors. Genetic Programming Theory and Practice II. Santa Clara: Springer; 2005. p. 310-315.

Miller LD, Smeds J, Joshy G, Vega VB, Vergara L, Ploner A, Pawitan Y, Hall P, Klaar S, Liu ET, Bergh J. An expression signature for p53 status in human breast cancer predicts mutation status, transcriptional effects, and patient survival. Proceedings of the National Academy of Sciences. Sep 20; 2005 102(38):13550–5.

Mitra AP, Almal AA, George B, Fry DW, Lenehan PF, Pagliarulo V, Cote RJ, Datar RH, Worzel WP. The use of genetic programming in the analysis of quantitative gene expression profiles for identification of nodal status in bladder cancer. BMC Cancer. 2006; 6:159. http://www.biomedcentral.com/1471-2407/6/159. [PubMed: 16780590]

Moore, JH.; Parker, JS. Evolutionary computation in microarray data analysis. In: Lin, S.; Johnson, K., editors. Methods of Microarray Data Analysis. Boston: Kluwer; 2002. p. 23-35.

Moore, JH.; White, BC. Using expert knowledge in initialization for genome-wide analysis of epistasis using genetic programming. In: Riolo, RL.; Soule, T.; Worzel, B., editors. Genetic Programming Theory and Practice IV. New York: Springer; 2006. p. 11-28.

Moore JH, Barney N, Tsai CT, Chiang FT, Gui J, White BC. Symbolic modeling of epistasis. Human Heredity. 2007; 63(2):120–33. [PubMed: 17283441]

Nachbar RB. Molecular Evolution: Automated Manipulation of Hierarchical Chemical Topology and Its Application to Average Molecular Structures. Genetic Programming and Evolvable Machines. 2000; 1:57–94.

Nunkesser R, Bernholt T, Schwender H, Ickstadt K, Wegener I. Detecting high-order interactions of single nucleotide polymorphisms using genetic programming. Bioinformatics. 2007; 23:3280–3288. [PubMed: 18006552]

Rechenberg, I. Farnborough: Lybrary Translation n1122. Cybernetic solution path of an experimental problem.

Reif DM, White BC, Moore JH. Integrated analysis of genetic, genomic, and proteomic data. Expert Review of Proteomics. 2004; 1(1):67–75. [PubMed: 15966800]

Selig MS, Coverstone-Carroll VL. Application of a genetic algorithm to wind turbine design. Journal of Energy Resources Technology. 1996; 118(1):22–28.

Sheta W, Eltonsy N, Tourasy G. Automated Detection of Breast Cancer From Screening Mammograps Using Genetic Programming. International Journal of Intelligent Computing and Information Sciences. 2005; 5(1):309–318.

Vladislavleva, E. Doctoral Thesis. Tilburg University; Tilburg, the Netherlands: 2008. Model-based Problem Solving through Symbolic Regression via Pareto Genetic Programming.

Wilson AF, Bailey-Wilson JE, Pugh EW. The Genometric Simulation Analysis Program (G.A.S.P.): a software tool for testing and investigating methods in statistical genetics. Am J Hum Gen. 1996; 59:A193.

Yao, X. Proceedings of the IEEE. Vol. 87. IEEE; Sep. 1999 Evolving Artificial Neural Networks [find similar] [try Google]; p. 1423-1447.

Yu J, Yu J, Almal AA, Dhanasekaran SM, Ghosh D, Worzel WP, Chinnaiyan AM. Feature Selection and Molecular Classification of Cancer Using Genetic Programming. Neoplasia. 2007; 9:292–303. [PubMed: 17460773]

Zhang, M. Genetic Programming Techniques for Multi-class Object Recognition. In: Canoni, S.; Lutton, E.; Olague, G., editors. Genetic and Evolutionary Computation for Image Processing and Analysis. Cairo: Hindawi; 2007. p. 349-370.

HHMI Author Manuscript
HHMI Author Manuscript
HHMI Author Manuscript

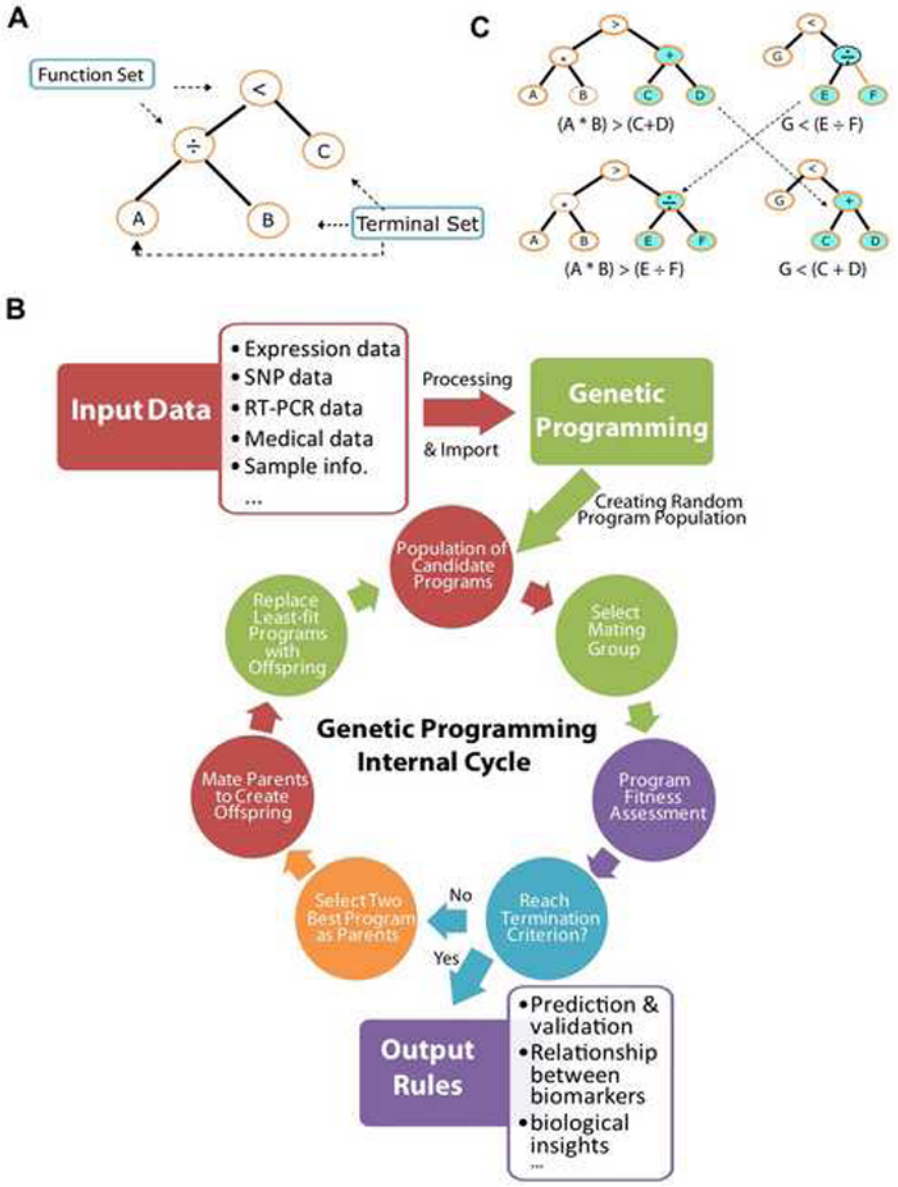

Figure 1.

Genetic Programming process. Programs are represented by tree structures in A. B shows the process of evolution starting from input data until output rules are produced. After rules are produced they are reviewed for biological insights and relationships or, particularly for clinical use, tested on an independent set to validate the results. **C.** An illustration of a *crossover* operator in genetic programming.

HHMI Author Manuscript
HHMI Author Manuscript
HHMI Author Manuscript

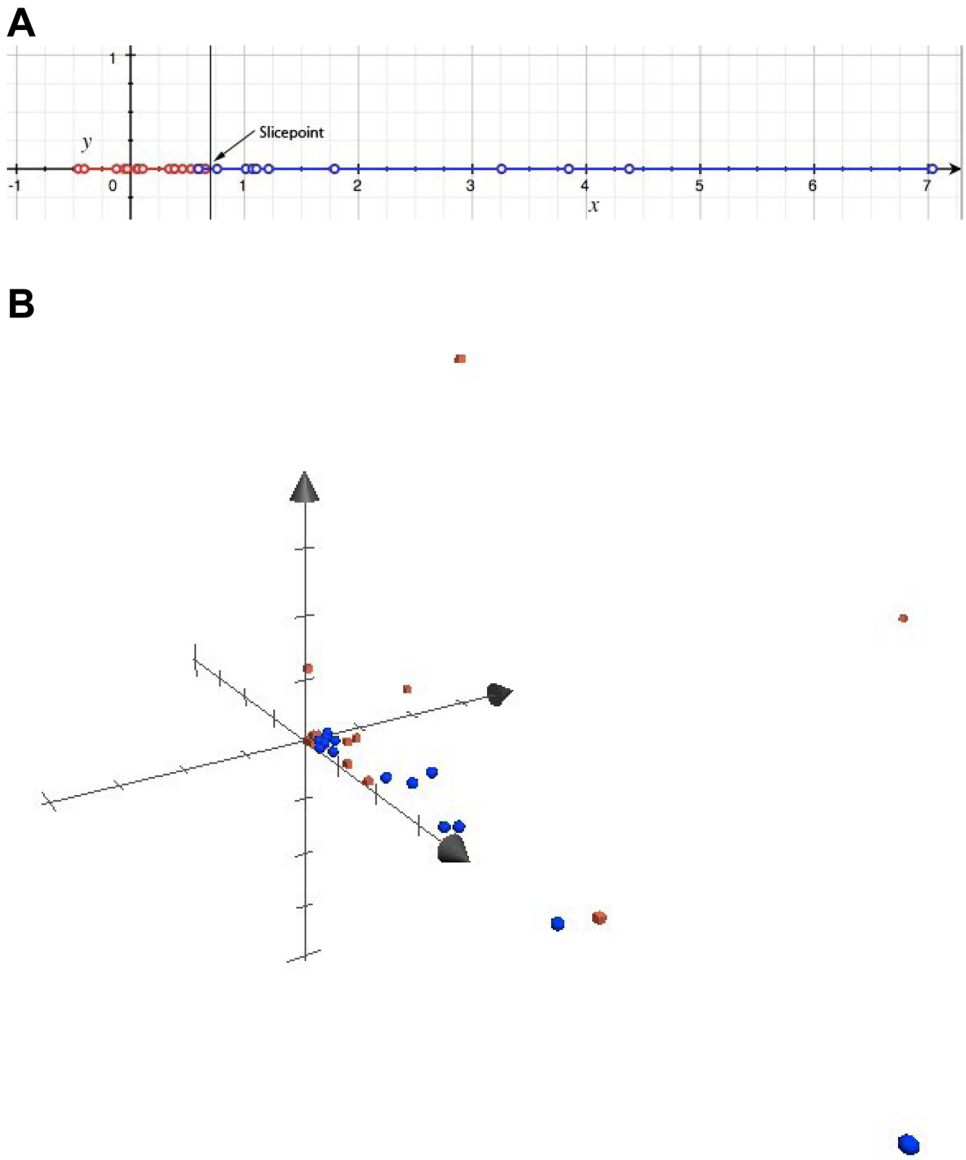

Figure 2.

A. Plot of values produced by classification function (Equation. 1). The line shows the location of the slicepoint separating the node negative samples (in red) from the node positive samples (in blue). **B.** Plot of ICAM1 (x-axis), KDR (y-axis) and MAP2K6 (z-axis) showing no clear separation between the node negative samples (red blocks) and the node positive samples (blue blocks).

HHMI Author Manuscript HHMI Author Manuscript HHMI Author Manuscript

A

Rules to classify PCA and benign prostate
IF (ENC1 + GJB1)>=-0.89 THEN PCA
IF (MYO6 + AMACR) >= -2.678 THEN PCA
IF (TSPAN13 + PRKCBP1) >= -0.417 THEN PCA
Ensemble rule: summation of the above rules

B

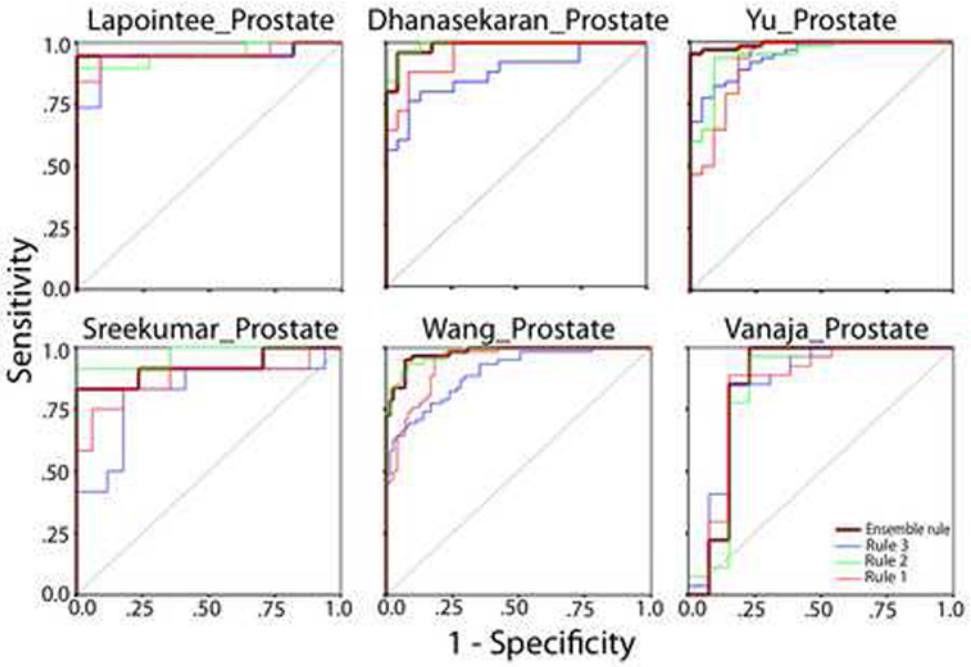

Figure 3.

The ROCs of three classification rules and one ensemble rule for six prostate cancer validation sets. The rules were generated from a training subset of Lapointee_Prostate to distinguish primary prostate cancer (PCA) from benign prostate. The ROCs are based on continuous values calculated from the left side of the rule inequation. The values of the ensemble rule is the summation of values of individual rules. All prostate cancer datasets are downloaded from ONCOMINE (http://www.oncomine.org).

HHMI Author Manuscript

HHMI Author Manuscript

HHMI Author Manuscript

IF ((MAP2K6 / KDR) * (1.0 – (MAP2K6/ICAM1)) >= 0.71 THEN Node-Positive

Equation 1.
Classifier rule for identifying node-positive bladder cancer samples