Predicting Reactions from Amino Acid Sequences in *S. cerevisiae*: an Evolutionary Computation Approach

Kyle Harrington
Hampshire College
893 West St
Amherst, MA
kyle@kephale.com

ABSTRACT

Evolutionary computation has been used many times for protein function prediction. In this paper a new approach is taken by constraining the problem to predicting the products of enzyme catalysis. Genetic programming with the Push programming language is used to evolve predictors within multiple search spaces. Predictors are evolved within multiple search spaces to reduce the complexity of solutions and represent sequence analysis, protein domain recognition, protein folding, and informatic approaches.

Categories and Subject Descriptors

J.3 [**Computer Applications**]: Life and Medical Sciences—*biology and Genetics*; I.2.2 [**Artificial Intelligence**]: Automatic Programming—*program synthesis*

General Terms

Evolutionary computation, genetic programming, sequence analysis, reaction prediction, protein function, protein analysis, protein folding

Keywords

GP^2, Push, PushGP

1. INTRODUCTION

Being able to determine what products will be produced by an amino acid and a set of chemical substrates is not a trivial task. Both chemicals and proteins are 3-dimensional objects, and the catalysis that is facilitated by proteins is dependent upon the 3-dimensional interactions between them. However, when considering the proteins from an informational perspective, information is conserved at the level of amino acid sequences. One protein is the same as another protein if it folded from the same amino acid sequence, disregarding the exceptions of misfolded proteins. Is it possible to look at this amino acid sequence and determine what it will catalyze a set of substrates into, without considering the fully folded protein? GP^2(Gene Prediction by Genetic Programming) uses evolutionary methods to develop predictors of catalytic reactions based upon the properties and representation of an amino acid sequence for a gene.

The underlying notion of GP^2 is that the products of a folded amino acid sequence's catalysis can be computed. Current efforts in protein folding, and functional analysis [3, 10, 14, 9], including the popular Folding@Home [15] compute the structure of proteins in one way or another. Computational protein folding models the process of an amino acid sequence forming a folded protein by simulating dynamics. Instead of taking this kinetic approach, an informatics approach is taken, similar to [5, 6]. The laws of physics are not simulated, instead variables representing information about physical properties are used.

GECCO'07, July 7–11, 2007, London, England, United Kingdom.

2. THE BIOLOGY

Proteins are macromolecules, encoded in DNA as genes, within cells acting as the functional building blocks of the cell. These functions range from the transport of molecules into and out of the cell, to the catalysis of reactions, to intracellular signaling [8]. The capability of proteins to provide extensive functionality for cellular control is what makes them such a valuable molecule for life. The function of many proteins still remain unknown, just as many functional proteins have yet to be discovered. This work is a step in the directions of understanding proteins of unknown function and the discovery of new proteins.

Enzymes are proteins which catalyze reactions, statically or dynamically [1]. Catalysis is the conversion of a set of chemical substrates into a set of chemical products. There are many types of catalysis that an enzyme can perform: hydrolases, isomerases, ATPases, etc.. Such reactions make up the overall chemical metabolism of biological organisms.

S. cerevisiae, budding yeast, has a long history as an organism for research and as a result there is a large body of research focusing on this organism. Although there are many reasons for its frequent study, a few are that it is a single-celled eukaryote, it is common, it is essential for the production of fermented goods, and the function of many of its genes are already known. Compared to humans, which are estimated have approximately $20,000$ to more than $100,000$ genes, yeast are estimated to have $6,128$ [16].

More recently ethanol is becoming an important chemical with its increasing use as an alternative fuel, which serves as a drive for the optimization of the ethanol fermentation process [7]. Genes from the energy metabolism of *S. cerevisiae*

are the subject of the catalysis predictors in this thesis. Given an accurate predictor of the given energy metabolism genes, alternative genes that provide similar functionality could be extrapolated.

3. PUSH AND GENETIC PROGRAMMING FOR PROTEIN ANALYSIS

Push is a stack-based programming language that has a syntax which lends itself to genetic programming [17]. A problem that frequently arises after the variation phase of genetic programming is whether the syntax of a statement is correct or not, such as "1+" the "+" operator needs one more argument. Push ignores this operator and continues on with the next. This means that every combination of operators can be executed. The Push programming language is stack-based, stacks being used to store both variables and code. The standard set of stacks allow for floating point, integer, boolean, code, and execution operations.

Evolutionary computation has repeatedly been used for protein classification [13, 12, 2, 4]. The results of these studies clearly demonstrate that evolutionary techniques can produce useful results without expert knowledge. Previously explored techniques investigate classification (is a protein segment transmembrane or not [13]), 3-dimensional structure, and other techniques of similar complexities. The techniques used in this thesis define a problem space of greater complexity than a binary classification problem, yet avoid the computational intensity of full 3-dimensional structural or protein folding.

3.1 Additional Push Stacks

In order for Push to be used to represent predictors of protein interaction with chemical compounds additional data types were required. Additional stacks were used to represent both chemicals, as well as amino acid sequences. At the beginning of each evaluation the chemical stack is filled with the substrate and the catalyzing amino acid sequence. The program performs a series of operations on the stacks (table 3.1). Once the program has been completely evaluated the products, or in the case of no interaction, the substrate, are remaining on the chemical stack.

Elements are simplified to 2 integers, symbol and quantity. This simplification could be considered a limiting factor when considering the amount of available information about chemicals, their structure and properties; however, when reducing the search space of to a computable complexity the amount of information was significantly reduced. This was done by using representative variables, including theoretical isoelectric point, molecular weight, atomic composition, hydropathicity, and chromosomal positioning. These coordinated pairs can be manipulated by converting symbols and quantities to and from integers, or converting elements to and from chemicals. Chemicals are groupings of N elements, which are used as a query/response data type.

Amino acid sequence operators allow for two levels of information about sequences. The first includes information about the sequence on the stack, as well as simple pattern searching operations. The second covers information about the initial input sequence, including operations covering physical forces, general sequence information, and the location of the sequence on/within a chromosome.

An amino acid sequence has a mobile single amino acid

Data Type	Operator
Amino Acid Sequence	Size
Amino Acid Sequence	Curr
Amino Acid Sequence	CurrHead
Amino Acid Sequence	CurrFoot
Amino Acid Sequence	CurrNext
Amino Acid Sequence	CurrPrev
Amino Acid Sequence	CurrDomainHDACInteract
Amino Acid Sequence	CurrDomainPGM PMM 1
Amino Acid Sequence	CurrDomainFromInteger
Amino Acid Sequence	DomainHDACInteract
Amino Acid Sequence	DomainPGM PMM 1
Amino Acid Sequence	DomainFromInteger
Amino Acid Sequence	AAComposition
Amino Acid Sequence	AACompositionX
Amino Acid Sequence	AACompositionXY
Amino Acid Sequence	nHydrophobic
Amino Acid Sequence	nHydrophilic
Amino Acid Sequence	nHydrophilicRun
Amino Acid Sequence	Fold
Amino Acid Sequence	RatioX
Amino Acid Sequence	RatiopairXY
Amino Acid Sequence	Molweight
Amino Acid Sequence	TheopI
Amino Acid Sequence	AtomiccompX
Amino Acid Sequence	Aliphatic
Amino Acid Sequence	Hydro
Amino Acid Sequence	Strand
Amino Acid Sequence	Position
Amino Acid Sequence	Cai
Amino Acid Sequence	Motif
Amino Acid Sequence	TransmembraneSpans
Amino Acid Sequence	Chromosome
Chemical	+
Chemical	FromAllElements
Chemical	FromNElements
Element	NewFromInteger
Element	SymbolFromInteger
Element	QuantityFromInteger
Element	FromChemical
Element	FromChemicalAtX
Element	FromChemicalSymbolX
Element	IncrementQuantity
Element	DecrementQuantity

Table 1: A listing of operators available in addition to the standard Push operators. These operators define how predictors can manipulate chemicals, elements, and amino acid seq uences to predict catalytic reactions.

Gene name	ENO1
Reactants	$C_3H_7O_7P$
Products	$C_3H_5O_6P$ and H_2O
Amino Acid Sequence	MAVSKVYARS...GENFHHGDKL a total of 472 amino acids

Table 2: An example data point which was used in training for experiment 1. In order for a predictor to be correct for these gene it would have to convert the the single reactant $C_3H_7O_7P$ into the two listed products, $C_3H_5O_6P$ and H_2O.

reading frame, referred to as a pointer. This pointer can be relocated to a variety of locations along the sequence for reading. It is also possible to evaluate the composition of the amino acid sequence. Composition is available as percentage composition of each amino acid, pairs of sequential amino acids, hydrophobic amino acids, hydrophilic amino acids, and N-run hydrophilic amino acids.

4. EVALUATION

Data was obtained from the Kyoto Encyclopedia of Genes and Genomes [11] and this thesis [6]. The data set of 33 cases included known reactions occurring within the starch/sucrose, glycolysis, and gluconeogenesis metabolic pathways of *S. cerevisiae*, both examples and counter-examples. An example data point for experiment 1 is shown in Table 2.

The fitness of solutions was evaluated by comparing the prediction to the actual products, and in some cases to the original substrate. Solutions are first constrained to the correct number of predicted chemicals with negative reward based upon error in length. Fitness was awarded based upon the similarity between each predicted product, and the closest matching actual product. Due to the simplified chemical data structure similarity between chemicals was determined by the similarity in atomic composition. In reactions where the substrate should be modified into new products but the predictor does not, fitness is docked. In reactions where it is correct to not modify the substrate(a null reaction), a constant reward is added. The fitness function was designed with the following prioritization: correct number of products, accuracy of products, null reactions, and substrate modification when required.

5. RESULTS

The results for 5 experiments are shown in figure 1. Experiment 1 used an instruction set limited to amino acid sequence pattern matching and chemical/element manipulation. Experiment 2 expanded on this to include previously discovered protein domains (functional patterns). Amino acid sequence folding was introduced into experiment 3. Many additional operators providing information about the inputted amino acid sequence were added for experiment 4. Experiment 5 is uses the same instruction set as experiment 4 with an increased size and duration of solution programs, and population size. These 4 different instruction sets represent sequence analysis, domain recognition, protein folding, and informatic approaches to protein function analysis.

Fitness values continue to increase over time, and are preliminary due to the computational complexity of the search. Experiments 1-4 all compute at approximately the same rate, whereas experiment 5 is significantly slower due to the increased parameters. Fitness values are normalized with respect to the fitness of a null predictor (one that does nothing), and a perfect predictor. Sporadic spikes in fitness values caused by a random number generator are discussed in the following section.

6. DISCUSSION AND CONCLUSION

The use of ephemeral random constants benefits greatly from the use of a random value operator. For example in pattern matching, if an integer, 17, is needed and no individual in the population contains the value, 17. If an individual contains a random integer then it has the possibility of being 17. This increases: the fitness of the solution, its reproductive competence, and the chances of an ephemeral 17 being inserted into a daughter program.

The results presented in this paper show the evolved predictors improving over the course of generations. The search space for such predictors is enormous due to the complexity of the problem. By confining the search space for solutions to the problem of enzyme catalysis prediction the search complexity can be decreased. Evolved predictors do manipulate the additional stacks; however, they do not fully complete any of the test cases. Continued work will demonstrate the whether such predictors can be evolved.

7. FUTURE WORK

Databases of metabolic information, such as KEGG, prove to be abundant data sources. By including additional metabolic networks, and additional organisms it should be possible to increase the quality of evolved predictors. Improved predictors are more likely to utilize unknown approaches.

Chemicals are physical structures, which means that proteins can, and do, utilize the orientation of their elements. It would be useful for the system to reference the position of elements within a compound, as well as the structure of amino acids.

8. ACKNOWLEDGMENTS

Much thanks to my adviser, Lee Spector, for useful and guiding comments and suggestions, and to my other fellow YeastRunners. This material is based upon work supported by the U.S. National Science Foundation under Grant No. 0308540. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] P. Agarwal. Enzymes: An integrated view of structure, dynamics and function. *Microb Cell Fact*, 5(1):2, Jan 2006.

[2] M. Brameier, J. Haan, A. Krings, and R. M. MacCallum. *Automatic discovery of cross-family sequence features associated with protein function.* BioMed Central Ltd., Jan. 12 2006.

[3] N. A. Burton, M. J. Harrison, J. C. Hart, I. H. Hillier, and D. W. Sheppard. Prediction of the mechanisms of enzyme-catalysed reactions using hybrid quantum mechanical/molecular mechanical methods. *Faraday Discuss*, (110):463–75; discussion 477–520, 1998.

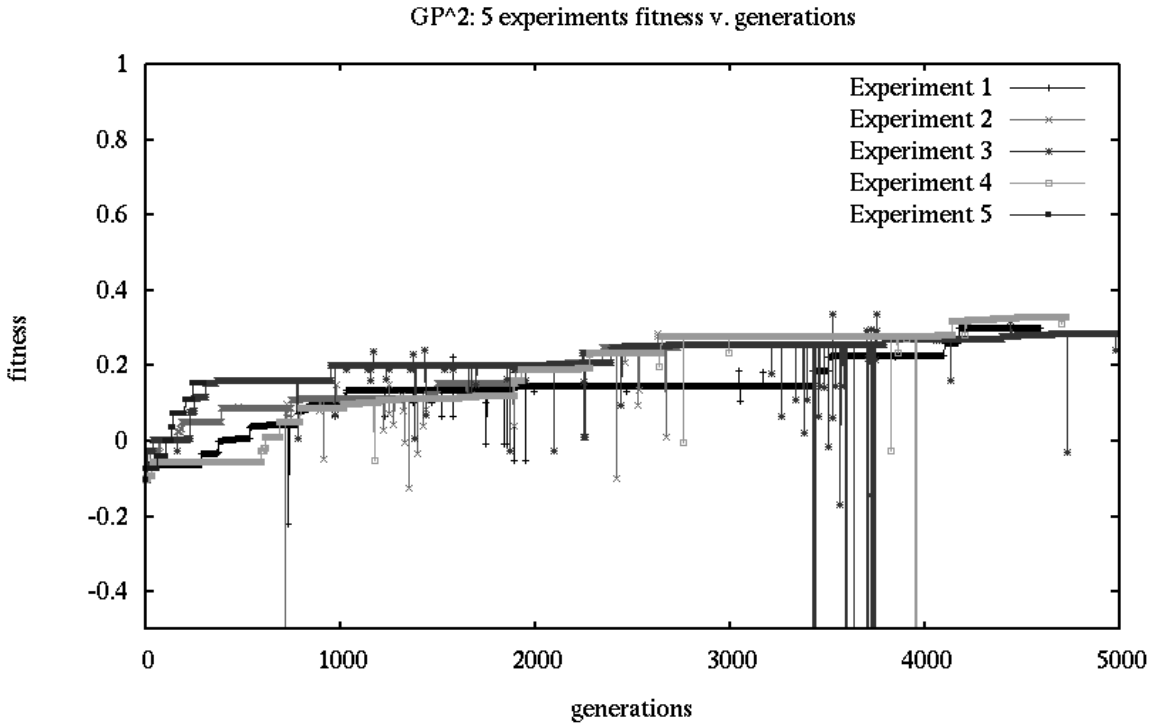

Figure 1: Preliminary results are shown for evolved predictors of catalytic function from amino acid sequence data, experiments 1,2,4 for 5000, 3 for 3000, and 5 for 500 generations. Fitness continues to improve over time for all experiments. Sporadic changes in fitness value are due to the use of random values.

[4] B. C. H. Chang, A. Ratnaweera, S. K. Halgamuge, and H. C. Watson. Particle swarm optimisation for protein motif discovery. *Genetic Programming and Evolvable Machines*, 5(2):203–214, June 2004.

[5] A. Clare. *Machine learning and data mining for yeast functional genomics*. PhD thesis, UWA, 2003.

[6] A. Clare and R. D. King. Predicting gene function in saccharomyces cerevisiae. *Bioinformatics*, 19:ii42–ii49, 2003.

[7] A. Costa, A. Henriques, T. Alves, R. M. Filho, and E. Lima. A hybrid neural model for the optimization of fed-batch fermentations. *Braz. J. Chem. Eng*, 16(1):53–63, October 2006.

[8] T. E. Creighton. *Proteins: Structures and Molecular Principles*. W. H. Freeman and Company, 1984.

[9] I. Famili, J. Forster, J. Nielsen, and B. O. Palsson. Saccharomyces cerevisiae phenotypes can be predicted by using constraint-based analysis of a genome-scale reconstructed metabolic network. *Proceedings of the National Academy of Sciences of the United States of America*, 100:13134–13139, 2003.

[10] N. Haspel, C.-J. Tsai, H. Wolfson, and R. Nussinov. Reducing the computational complexity of protein folding via fragment folding and assembly. *Protein Sci*, 12(6):1177–1187, Jun 2003.

[11] M. Kanehisa, S. Goto, M. Hattori, K. F. Aoki-Kinoshita, M. Itoh, S. Kawashima, T. Katayama, M. Araki, and M. Hirakawa. From genomics to chemical genomics: new developments in KEGG. *Nucleic Acids Res*, 34(Database issue):D354–D357, Jan 2006.

[12] J. Koza, F. Bennett, and D. Andre. Using programmatic motifs and genetic programming to classify protein sequences as to extracellular and membrane cellular location. 1447, 25-27 Mar. 1998.

[13] J. R. Koza. Recognizing patterns in protein sequences using iteration-performing calculations in genetic programming. 1:244–249, 27-29 June 1994.

[14] L. Malmstrm, M. Riffle, C. E. M. Strauss, D. Chivian, T. N. Davis, R. Bonneau, and D. Baker. Superfamily assignments for the yeast proteome through integration of structure prediction with the gene ontology. *PLoS Biol*, 5(4):e76, Mar 2007.

[15] C. D. Snow, Y. M. Rhee, and V. S. Pande. Kinetic definition of protein folding transition state ensembles and reaction coordinates. *Biophys J*, 91(1):14–24, Jul 2006.

[16] M. Snyder and M. Gerstein. Genomics. defining genes in the genomics era. *Science*, 300(5617):258–260, Apr 2003.

[17] L. Spector, C. Perry, J. Klein, and M. Keijzer. Push 3.0 programming language description. Technical report, Hampshire College, http://hampshire.edu/lspector/push3-description.html, 2004.