Automatic Discovery Using Genetic Programming of an Unknown-Sized Detector of Protein Motifs Containing Repeatedly-Used Subexpressions

John R. Koza
Computer Science Department
Stanford University
Stanford, California 94305
Koza@CS.Stanford.Edu 415-941-0336
http://www-cs-faculty.stanford.edu/~koza/

David Andre
Visiting Scholar
Computer Science Department
Stanford University 94305-2140 USA
E-MAIL: Andre@flamingo.stanford.edu
Phone: 415-326-5113
Fax: 415-941-9430

Abstract

Automated methods of machine learning may be useful in discovering biologically meaningful patterns that are hidden in the rapidly growing databases of genomic and protein sequences. However, almost all existing methods of automated discovery require that the user specify, in advance, the size and shape of the pattern that is to be discovered. Moreover, existing methods do not have a workable analog of the idea of a reusable subroutine to exploit the recurring sub-patterns of a problem environment.

Genetic programming can evolve complicated problem-solving expressions of unspecified size and shape. When automatically defined functions are added to genetic programming, genetic programming becomes capable of efficiently capturing and exploiting recurring sub-patterns.

This paper describes how genetic programming with automatically defined functions successfully evolved motifs for detecting the D-E-A-D box family of proteins and for detecting the manganese superoxide dismutase family. Both motifs were evolved without prespecifying their length. Both evolved motifs employed automatically defined functions to capture the repeated use of common subexpressions. When tested against the SWISS-PROT database of proteins, the two genetically evolved consensus motifs detect the two families either as well, or slightly better than, the comparable human-written motifs found in the PROSITE database.

1. INTRODUCTION

Automated methods of machine learning may be useful in discovering biologically meaningful patterns that are hidden in the rapidly growing databases of genomic and protein sequences. Unfortunately, almost all existing methods of automated discovery require that the user specify, in advance, the size and shape of the pattern that is to be discovered. However, in practice, the discovery of the size and shape of the pattern may, in fact, be *the problem* (or at least a major part of the problem). Moreover, none of the existing methods of automated discovery have a workable analog of the idea of a reusable subroutine or subprogram to capture and exploit repeated occurrences of regularities or sub-patterns of the problem environment.

The problem of discovering biologically meaningful patterns in databases can be rephrased as a search for an unknown-sized task-performing computer program (i.e., a composition of primitive functions and terminals). When this motif discovery problem is so rephrased, genetic programming becomes a candidate for solving this

problem. Moreover, if it is also desired to reuse regularities in the problem environment, then genetic programming with automatically defined functions becomes a candidate.

Section 2 of this paper provides background on protein databases, motifs, the D-E-A-D box family of proteins, and the manganese superoxide dismutase family. Section 3 provides background on genetic programming. Section 4 identifies the preparatory steps required to apply genetic programming to the motif discovery problem. Section 5 describes the implementation of genetic programming on a parallel computer. Section 6 presents a genetically evolved consensus motif that is slightly better than the human-written motif found in the PROSITE database for detecting the D-E-A-D box family of proteins. Section 7 presents a genetically evolved consensus motif for detecting the manganese superoxide dismutase family which is as good as the human-written motif found in the PROSITE database. A biological interpretation is given for the genetically evolved motif of both sections 6 and 7.

2. BACKGROUND ON PROTEINS, PROTEIN DATABASES, MOTIFS

The structure and functions of living organisms are primarily determined by proteins (Stryer 1988). Proteins are large polypeptide molecules composed of sequences of up to several thousand amino acid residues. All proteins are composed from the same repertoire of 20 amino acid residues (denoted by the letters A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, and Y). Subject to only a few minor qualifications, the three-dimensional location of every atom of a protein is fully determined by its sequence of amino acid residues (its *primary structure*) (Anfinsen 1973). The protein's three-dimensional structure (its *tertiary structure* or *conformation*), in turn, determines the biological structure, function, and activity of the protein within a living organism. Thus, effectively all of the information about the biological structure, function, and activity of a protein is contained (albeit hidden) in its primary sequence (i.e., the linear sequence of letters).

SWISS-PROT is a massive, systematically-collected, periodically-reviewed, annotated database of protein sequences that is maintained by the University of Geneva and the European Molecular Biology Laboratory (Bairoch and Boeckmann 1991). Release 30 of SWISS-PROT (October 1994) contains 14,147,368 amino acid residues from 40,292 sequences from hundreds of different species. The Human Genome Project and other research efforts in molecular biology are rapidly increasing the number of entries in SWISS-PROT and other databases of protein and DNA sequences. Automated analysis techniques (such as those of machine learning) may prove necessary for analyzing this accumulating data.

Most proteins appear in many different species; however, the primary sequences of the "same" protein in two different species are usually not identical. For one thing, the primary sequences usually differ slightly in length. Moreover, even after using an alignment algorithm (e.g., Smith and Waterman 1981) to align such sequences, the residues found at a particular aligned position often still differ. The reason is that only relatively small subsequences of the overall sequence are responsible for the biological function and activity of the protein. Even when one finds this subsequence, only a few (sometimes none) of the residues of the subsequence are actually identical (that is, *conserved*). The reason is that, over millions of years, evolution substitutes chemically-similar residues into a protein sequence. Moreover, evolution often substitutes dissimilar residues at non-critical positions and even changes the sequence length by inserting or deleting residues.

Sometimes, amidst all the differences, it is possible to identify certain high specificity, high sensitivity patterns (called *motifs*, *sites*, *signatures*, or *fingerprints*) in a set of sequences for biologically similar proteins. If a motif is defined well, it will detect a biologically-important common property. The residues in such motifs often prove to be directly responsible for the essential function and activity of the protein.

PROSITE is a database of biologically meaningful patterns found in protein sequences (Bairoch and Bucher 1994). Release 12 of PROSITE (June 1994) contains 1,029 different motifs. Since the intended primary purpose of PROSITE is to detect families of proteins in computerized databases, a motif is included in PROSITE if it detects most (preferably all) sequences that have a particular biological property (i.e., has few false negatives), while detecting few (preferably zero) unrelated sequences (i.e., has few false positives).

2.1 The D-E-A-D Box Family of Proteins

In the "Birth of the D-E-A-D box," Linder and his colleagues (1989) described a family of proteins (called *helicases*) involved in the unwinding of the double helix of the DNA molecule during the replication of DNA (Chang, Arenas, and Abelson 1990; Dorer, Christensen, and Johnson 1990; Hodgman 1988). This family of proteins gets its name from the fact that the amino acid residues D (aspartic acid), E (glutamic acid), A (alanine), and D appear, in that order, at the core of one of the biologically critical subsequences of this protein. There are 34 proteins from this family among the 40,292 proteins appearing in Release 30 of SWISS-PROT. Proteins of this family can be detected effectively (but not perfectly) by the following motif (called ATP_HELICASE_1) of length nine that was included by Amos Bairoch at the University of Geneva in the PROSITE database:

[LIVM]-[LIVM]-D-E-A-D-x-[LIVM]-[LIVM].

In interpreting this expression, the first pair of square brackets indicates that the first residue of the nine is to be chosen from the set consisting of the amino acid residues

L, I, V, and M. The second pair of square brackets indicates that the second residue is chosen (independently from the first) from the same set of four possibilities. Then, the third, fourth, fifth, and sixth residues must be D, E, A, and D, respectively. The X in the motif indicates that the seventh residue can be any of the 20 possible amino acid residues. The eighth and ninth residues are chosen from the same set of four, namely L, I, V, and M.

D and E are negatively charged and hence hydrophilic (water-loving) at normal pH values. A is small, uncharged, hydrophobic (water-hating). L (leucine), I (isoleucine), V (valine), or M (methionine) are moderately-sized, uncharged, and hydrophobic. Thus, ignoring the X, this motif calls for three hydrophilic residues and one hydrophobic residue accompanied, on each side, by two moderately-sized hydrophobic residues.

The above PROSITE expression detects any of $4^4 \infty 20 =$ 5,120 different possible sequences of length nine (out of approximately $5 \infty 10^{11}$ possible sequences of length nine). When SWISS-PROT is searched using the above PROSITE expression, there are 34 true positives, 14,147,333 true negatives (among the 40,292 proteins), 1 false positive, and 0 false negatives. This corresponds to a correlation coefficient (Matthews 1975), *C*, of 0.99. The number of PROSITE expressions (composed of disjunctions such as shown above) covering exactly nine positions is $(2^{20})^9 \sim 10^{54}$. Since the length of an expression that capable of detecting a particular family of proteins is, in actual practice, not known in advance, the search space of the motif discovery problem is considerably larger than 10^{54}.

2.2 The Manganese Superoxide Dismutase Family of Proteins

The oxygen radicals that are normally produced in living cells have been implicated in many degenerative processes, including cancer and aging. Proteins belonging to the manganese superoxide dismutase family prevent oxidative damage to DNA and other molecules by catalyzing the conversion of these toxic superoxide radicals to oxygen and hydrogen peroxide (Ludwig et al. 1991; Stoddard, Ringe, and Petsko 1990; Bannister, Bannister, and Rotilio 1987). The four ligands of the manganese atom are conserved in all the known sequences of the manganese superoxide dismutase family. Amos Bairoch selected a short conserved region that includes two of the four ligands, namely one D (aspartic acid) and one nearby H (histidine), to create the following motif of length eight (called SOD_MN) for detecting proteins belonging to this family:

D-x-W-E-H-[STA]-[FY][FY].

For example, in human manganese superoxide dismutase (whose length is 198), the above motif correctly identifies the protein as belonging to this family because residues 159 to 166 of this protein are

DVWEHAYY.

When it is tested against all of SWISS-PROT, the above motif scored 40 true positives, 14,147,328 true negatives, and 0 false positive and 0 false negatives (for a correlation of 1.00).

The Protein Data Bank (PDB) maintained by the Brookhaven National Laboratory in Upton, New York (Bernstein et al. 1977) is the worldwide computerized repository of the three-dimensional coordinates of the atomic structure of proteins. Proteins from the PDB can be interactively displayed by making a three-dimensional *kinemage* of the protein using the PREKIN software and viewing the kinemage with the MAGE software (Richardson and Richardson 1992). Figure 1 shows residues 159 to 166 of human manganese superoxide dismutase as well as the histidines at positions 26 and 74 of the protein sequence. The manganese is ligated by Asp 159, His 163, His 26, and His 74.

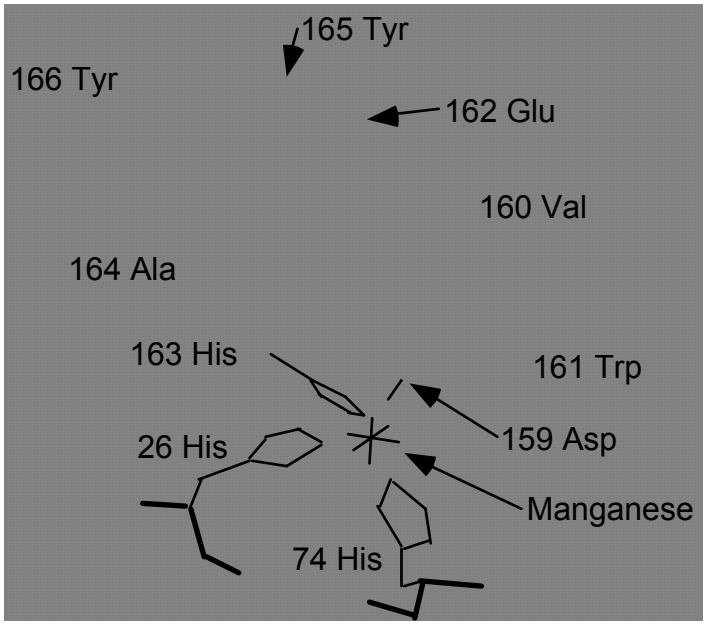

Figure 1 Active site of human manganese superoxide dismutase.

3. BACKGROUND ON GENETIC PROGRAMMING

John Holland's pioneering *Adaptation in Natural and Artificial Systems* described how the evolutionary process in nature can be applied to solving problems using what is now called the *genetic algorithm* (Holland 1975).

Genetic programming is an extension of the genetic algorithm in which the genetic population consists of computer programs. *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (Koza 1992) provides evidence that genetic programming can solve, or approximately solve, a variety of problems from a variety of fields. Recent work on genetic programming is found in Kinnear 1994. A videotape visualization of a number of applications of

genetic programming can be found in Koza and Rice (1992) and Koza (1994). The sequence of work-performing steps of the to-be-evolved programs are not specified in advance by the user. Instead, the sequence of steps is evolved as a result of the competitive and selective pressures of the evolutionary process and the recombinative role of crossover.

When humans write programs, they use subroutines to exploit, *by reuse*, the regularities, symmetries, homogeneities, similarities, patterns, and modularities of problem environments. *Genetic Programming II: Automatic Discovery of Reusable Programs* (Koza 1994a) extends genetic programming to evolve multi-part programs consisting of a main program and one or more reusable hierarchically-callable subprograms.

The PROSITE language has no facility for defining and using subroutines. For example, the D-E-A-D box motif found in the PROSITE database contains a repeatedly used subexpression consisting of four moderately-sized, uncharged, hydrophobic residues (L, I, V, and M). Similarly, the manganese superoxide dismutase motif contains a repeatedly used subexpression consisting of Y and F. Because of this manifest modularity, genetic programming with automatically defined functions may be appropriate for evolving a motif-detecting program for the these families of proteins.

Genetic programming with automatically defined functions may be a promising approach to this problem because when it was applied to the problem of identifying transmembrane domains in proteins, the results were slightly better than previous human-written algorithms (Koza 1994c). Genetic programjming has also been used to identify omega loops in proteins (Koza 1994c), to predict whether a residue in a protein sequence is in an α-helix (Handley 1993a, 1994a), to predict the degree to which a protein sequence is exposed to solvent (Handley 1994b), to predict whether or not a nucleic acid sequence is an *E. coli* promoter region (Handley 1995a); to predict whether or not a 60-base DNA sequence contains a centrally-located splice site (Handley 1995b); and to classify a nucleic acid subsequence as being an intron or exon (Handley 1995c).

4. PREPARATORY STEPS FOR THE D-E-A-D BOX PROBLEM

In applying genetic programming with automatic function definition to a problem, there are six major preparatory steps, namely determining

(1) the set of terminals for each branch,
(2) the set of primitive functions for each branch,
(3) the fitness measure for evaluating how well a program does at solving the problem,
(4) the parameters and variables for controlling the run,
(5) the criterion for terminating a run and designating the result, and
(6) the architecture of the overall multi-part program.

4.1 Terminal Set, Function Set, and Architecture

After analyzing the problem, it seems reasonable that the ingredients of the to-be-evolved computer programs should include 20 zero-argument logical functions capable of interrogating the current position of a protein sequence. For example, `(A?)` is the zero-argument residue-detecting function that returns 1 if the current residue in the sequence is alanine (A) but otherwise returning 0.

The square brackets used in PROSITE expressions are used to form sets of residues. The two-argument disjunctive function `OR` can be used to dynamically define disjunctions of the values returned by the 20 residue-detecting functions.

If the overall architecture of the yet-to-be-evolved motif-detecting program uses automatically defined functions to organize amino acid residues into subsets, then the result-producing branch can be used to perform some operations to reach a conclusion. The two-argument conjunctive function `AND` corresponds to the dash of the PROSITE language; it is used to advance to next position of protein sequence. This suggests an overall architecture consisting of several (say, two) automatically defined functions and one result-producing branch.

Specifically, the terminal set, $\mathcal{T}_{adf}$, for the two function-defining branches (`ADF0` and `ADF1`) contains the 20 zero-argument residue-detecting functions. That is,

$$\mathcal{T}_{adf} = \{(A?), (C?), (D?), \ldots, (Y?)\}.$$

The function set, $\mathcal{F}_{adf}$, for the two function-defining branches is

$$\mathcal{F}_{adf} = \{OR\}.$$

The terminal set, $\mathcal{T}_{rpb}$, for the result-producing branch includes the now-defined ADFs, `ADF0` and `ADF1`, and the 20 zero-argument residue-detecting functions.

$$\mathcal{T}_{rpb} = \{ADF0, ADF1, (A?), (C?), (D?), \ldots, (Y?)\}.$$

The function set, $\mathcal{F}_{rpb}$, for the result-producing branch is

$$\mathcal{F}_{rpb} = \{AND\}.$$

Programs consist of two function-defining branches (`ADF0` and `ADF1`) composed of functions from $\mathcal{F}_{adf}$ and terminals from $\mathcal{T}_{adf}$, as well as one result-producing branch composed of functions from $\mathcal{F}_{rpb}$ and terminals from $\mathcal{T}_{rpb}$. Note that we do not prespecify the length of the motif that is to be evolved. Both the size and content of each branch of the multi-part program is to be evolved by genetic programming.

If the result-producing branch of an overall program returns a logically true value at a particular position in a protein sequence, that position will be identified as the beginning of an occurrence of the motif; otherwise that position will be classified negatively. If a program

examines a residue that is beyond the C-terminal (end) of the protein, the position will be classified negatively.

4.2Fitness Measure

The fitness measure determines how well a particular genetically-evolved motif-detecting program predicts whether a particular amino acid residue is the beginning of a motif. Correlation is appropriate as a measure of raw fitness for genetic programming in a two-way classification problem.

Fitness is measured over a number of trials called *fitness cases*. A set of in-sample fitness cases (i.e., the training set) is used to measure the fitness of programs during the evolutionary process. The fitness cases for this problem are individual amino acid residues of proteins. The single residue at the start of the occurrence of the motif is a positive fitness case and all other residues are negative fitness cases.

For example, for the D-E-A-D box problem, the set of in-sample fitness cases contained all the residues of 26 of the 34 proteins in SWISS-PROT. Because of the rarity of the motif among the 14,147,368 residues in SWISS-PROT and in order to save computer time, we extracted 210 30-residue fragments from proteins that did not belong to the D-E-A-D box family, did not contain the D-E-A-D box motif, but did contain a sizable partial match with this motif (such as all `X-X-D-E-A-D-X-X-X` or `V-X-X-EAD-X-X-X` that are not in the D-E-A-D box family). When we were done, the in-sample fitness cases consisted of 19,200 amino acid residues from 236 proteins (26 residues being positive instances and l9,174 being negative instances).

After a motif-detecting program is evolved using the in-sample fitness cases, the question arises as to how well it generalizes to previously unseen, different fitness cases from the same problem environment. A set of out-of-sample fitness cases consisting of 5,605 amino acid residues from 53 proteins (8 residues being positive instances and 5,597 being negative instances) was used to validate the performance of a genetically-evolved motif-detecting program. For reference, when the D-E-A-D box motif found in the PROSITE database is tested against the out-of-sample fitness cases, there are 8 true positives, 5,596 true negatives, 1 false positive, and 0 false negatives (for a correlation of 0.94). When it is tested against all of SWISS-PROT, it scored 34 true positives, 14,147,333 true negatives, and 1 false positive and 0 false negatives (for a correlation of 0.99).

4.3Parameters

A population size, M, of 256,000 was used. The targeted maximum number of generations, G, was set at 201 (although every run we made yielded a solution long before generation 200). The maximum size (i.e., number of functions and terminals in the work-performing parts of each branch) was 50 points per branch. Minor parameters were chosen as in Koza 1994a.

4.4Termination Criterion and Result Designation

The termination criterion for any one run of this problem is the emergence of an evolved program with an in-sample correlation of 1.00 on the in-sample fitness cases. That program is designated as the result of the run.

4.5Jury Method for Creating a Consensus Motif

Because of the intended purpose of PROSITE (which is purposely oriented toward overfitted descriptions) and because the 1,029 PROSITE motifs partition the existing databases into relatively small subsets, there is a poverty of instances of any given motif in the database. The difficulties of evolving a motif from such an impoverished database can be compensated for by using a jury (Rost and Sander 1993) of evolved results having an in-sample correlation of 1.00. A unanimous decision by a jury of 12 evolved results on new, previously unseen, out-of-sample fitness cases was required in order to classify a position of a protein sequence as the beginning of the motif. Otherwise, the position was classified negatively.

5. IMPLEMENTATION OF PARALLEL GENETIC PROGRAMMING

The problem (written in ANSI C) was run on a home-built medium-grained parallel computer system consisting of 64 INMOS transputers (housed on Transtech 4 megabyte TRAMs) arranged in a toroidal mesh with a host PC 486 type computer (running Windows). The so-called *distributed genetic algorithm* or *island model* for parallelization (Tanese 1989, Goldberg l989) was used. That is, subpopulations (called *demes* in Sewell Wright 1943) were situated at the processing nodes of the system. Population size was Q = 4,000 at each of the D = 64 demes. The initial random subpopulations were created locally at each processing node. Generations were run asynchronously on each node. After a generation of genetic operations was performed locally on each node, four boatloads, each consisting of B = 8% (the migration rate) of the subpopulation (selected on the basis of fitness) were dispatched to each of the four toroidally adjacent nodes. Details are in Koza and Andre 1995.

6. RESULTS FOR THE D-E-A-D BOX FAMILY

```
IN ONE RUN, THE BEST MOTIF-DETECTING
PROGRAM FOR THE D-E-A-D BOX FAMILY
AMONG THE 256,000 RANDOM PROGRAMS OF
GENERATION 0 SCORED 20 TRUE POSITIVES,
19,152 TRUE NEGATIVES, 22 FALSE
POSITIVES, AND 6 FALSE NEGATIVES ON THE
IN-SAMPLE FITNESS CASES (FOR A
CORRELATION OF 0.60) AND IS SHOWN
BELOW:

(PROGN (DEFUN ADF0 ()
  (values (OR (L?) (N?))))
      (DEFUN ADF1 ()
  (values (OR (R?) (V?))))
```

```
(VALUES (AND (V?) (AND (L?)
(D?)))))
```

Note that although we program genetic programming in C, LISP is used to present evolved programs since it highlights the program's tree structure. This best-of-generation program defines the motif, `V-L-D`, of length three. This motif admits no alternatives in any of its three positions and ignores the subsets defined by its automatically defined functions.

In subsequent generations, the programs in the population became more complex and their fitness improved apace. The length of the motifs started to increase (sometimes beyond the length of the D-E-A-D box motif in PROSITE). The result-producing branches started to refer to one or both of its automatically defined functions. The automatically defined functions started to be used two or more times.

On generation 42 of one run, the best-of-generation program (shown below) scored 26 true positives, 19,174 true negatives, and 0 false positives and 0 false negatives (for an in-sample correlation of 1.00):

```
(PROGN (DEFUN ADF0 ()
  (values (OR (OR (OR (OR (w?)(m?))(OR
    (c?)(a?)))(m?))(OR (OR (OR (OR (w?)(m?))(OR
    (m?)(i?)))(l?))(i?)))))
      (DEFUN ADF1 ()
  (values (OR (OR (OR (a?) (e?))(OR (OR (OR (OR
    (a?)(k?))(OR (v?)(n?)))(OR (OR (e?)(r?))(OR (OR
    (a?)(w?))(OR (k?)(q?)))))(OR (v?)(k?))))(OR (OR
    (i?)(c?))(a?)))))
      (VALUES (AND (AND (AND (AND (AND
      (AND (AND (ADF1) (ADF0)) ( AND
      (D? ) (E? ))) (ADF0)) (D? ))
      (ADF1)) (ADF0)) (ADF0)))
```

This best-of-generation program defines the following motif of length nine:

[VIAEKNRWQC]-[LIMCAW]-D-E-[LIMCAW]-D-
[RNEKVIAWQC]-[LIMCAW]-[LIMCAW]

Note that the common sub-expression `[LIMCAW]` defined in `ADF0` is used a total of four times in the above overall expression. When tested on the 5,605 out-of-sample fitness cases, this expression scored 8 true positives, 5,597 true negatives, and 0 false positives and 0 false negatives (for a correlation of 1.00). When tested against SWISS-PROT, this program scored 34 true positives, 14,147,328 true negatives, and 6 false positives, and 0 false negatives (for a correlation of 0.92).

In another run, the best-of-generation program from generation 64 had an in-sample correlation of 1.00 and defined the following motif of length ten:

[FVIAC]-[LIM EQDNRSK]-D-E-[AFVIC]-D-
[LIMEQDNRSK]-[LIMEQDNRSK] -[LIMEQDNRSK]-
[LIMEQDNRSK]

In other runs, 10 additional evolved programs each had an in-sample correlation of 1.00. These 12 results participated in a jury that created a genetically evolved consensus motif of length 10 (shown below) that scored 26 true positives, 19,174 true negatives, 0 false positives, and 0 false negatives (for a correlation of 1.00):

[IV]-[lim]-D-E-[AI]-D-[rnek]-[lim]-[lim]-[limeqdnrsk]

Note that `[LIM]` is used three times in this expression. When tested on the 5,605 out-of-sample fitness cases, this expression scored 8 true positives, 5,597 true negatives, and 0 false positives and 0 false negatives (for a correlation of 1.00). When tested against SWISS-PROT, this program scored 34 true positives, 14,147,334 true negatives, and 0 false positives, and 0 false negatives (for a correlation of 1.00).

Thus, the genetically evolved consensus motif created by the jury scored slightly better than the human-written motif found in the PROSITE database on the problem of detecting the D-E-A-D box family of proteins.

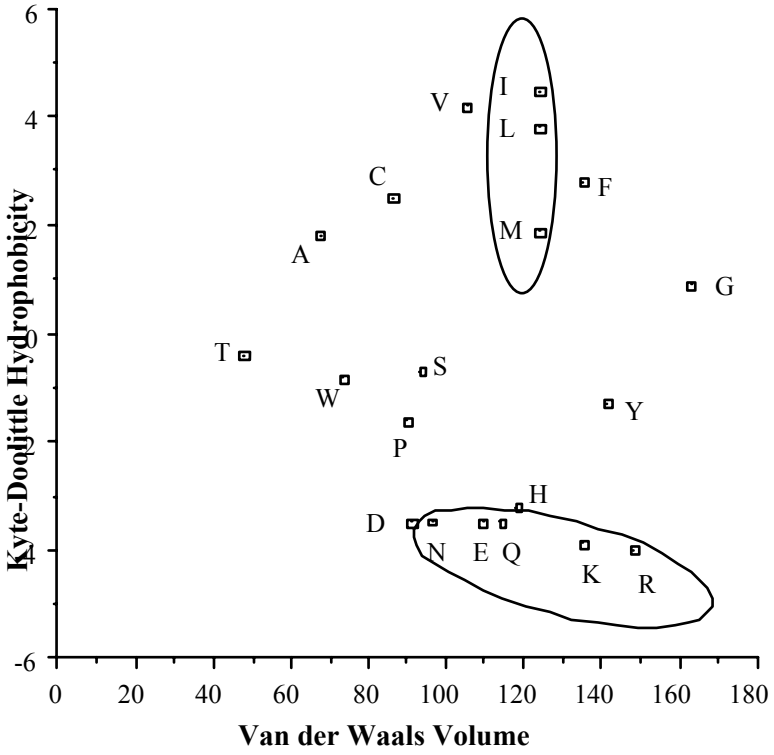

Figure 2 Scatter diagram of hydrophobicity and volume.

Recalling that the motif found in the PROSITE database for the D-E-A-D box family is
[LIVM]-[LIVM]-D-E-A-D-x-[LIVM]-[LIVM],
one can see that the genetically evolved consensus motif differs in the following ways from the motif found in the PROSITE database.

First, position 7 of the motif has a definite character. The X in position 7 of the PROSITE motif is replaced by [RNEK] in the consensus motif. Figure 2 is a scatter diagram relating the Van der Waals volume (Creighton 1993) and the hydrophobicity values (Kyte and Doolittle 1982) of the 20 amino acids. In this figure, R, N, E, and K are located in the same general area (circled in the lower right) indicating that they are all highly hydrophilic and bulky. Second, the [LIVM] in positions 2, 8, and 9 of the PROSITE motif is replaced by the somewhat more precise [LIM] in the consensus motif. As can be seen in the circled area at the top right of the figure, residues I, L, and M have virtually identical volumes whereas V has a different volume.

Third, the [LIVM] in position 1 of the PROSITE motif is replaced by the somewhat more precise [IV] in the consensus motif.

Fourth, the genetically evolved consensus motif specifies that position 10 (beyond the last position specified by the PROSITE motif) contains [LIMRNEKQDS].

7. RESULTS FOR THE MANGANESE SUPEROXIDE DISMUTASE FAMILY

The in-sample fitness cases for the problem of detecting the manganese superoxide dismutase family of proteins consisted of 13,518 amino acid residues from 270 proteins (30 residues being positive instances and 13,488 being negative instances). The out-of-sample fitness cases were constructed using the same approach as described above and consisted of 3,280 residues from 53 proteins (10 residues being positive instances and 3,270 being negative instances).

When the manganese superoxide dismutase motif found in the PROSITE database is tested against the out-of-sample fitness cases, there are 10 true positives, 3,270 true negatives, 0 false positive, and 0 false negatives (for a correlation of 1.00). When it is tested against all of SWISS-PROT, it scores a correlation of 1.00.

Three automatically defined functions were used on this problem.

The genetically evolved motifs participated in a jury that created the following consensus motif of length nine that had an in-sample correlation of 1.00:
D–[VAML]–W–E–H–[SA]–[YFH]–[YFAHS]–[YFADHLIS]

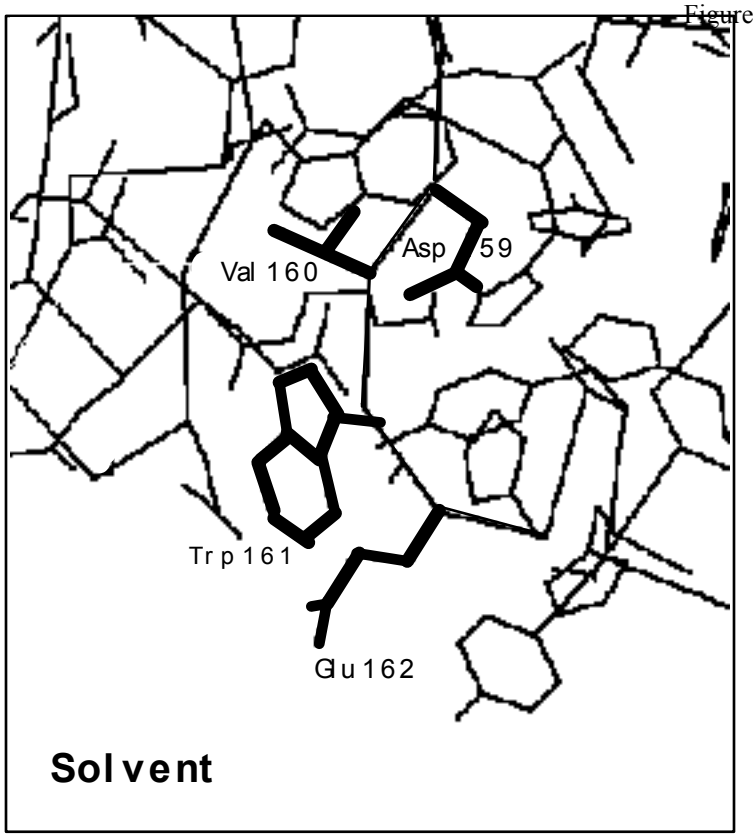

Figure 3 Section of protein showing that position 160 is buried.

When tested on the 3,280 out-of-sample fitness cases, this expression had an out-of-sample correlation of 1.00. When tested against SWISS-PROT, this program scored 40 true positives, 14,147,328 true negatives, and 0 false positives, and 0 false negatives (for a correlation of 1.00). That is, the genetically evolved consensus motif created by the jury scored as well as the human-written motif found in the PROSITE database on the problem of detecting the manganese superoxide dismutase family of proteins.

The motif found in the PROSITE database for the manganese superoxide dismutase family is
D-x-W-E-H-[STA]-[FY][FY].
The genetically evolved consensus motif differs in the following ways from the motif found in the PROSITE database.

First, the X in position 2 of the PROSITE motif is replaced by the set of hydrophobic residues [VAML]. As can be seen from figure 3, position 2 of the motif (i.e., position 160 of the protein) is buried (in contrast to, for example, electrically charged Glu 162 which is exposed to the solvent). Thus, it is reasonable that whatever appears at position 2 should be hydrophobic.

Second, the [STA] in position 6 of the PROSITE motif is replaced by the somewhat more precise [SA] in the consensus motif.

Third, the genetically evolved consensus motif specifies that position 9 (beyond the last position specified by the PROSITE motif) contains [YFADHLIS].

8. CONCLUSIONS

Genetic programming was successfully used to create motifs for the D-E-A-D box family of proteins and the manganese superoxide dismutase family. Both motifs were evolved without prespecifying their length. Both evolved motifs employed automatically defined functions to capture the repeated use of a common subexpression. When tested against the SWISS-PROT database of proteins, the two genetically evolved consensus motifs detect the two families either as well, or slightly better than, the comparable human-written motifs found in the PROSITE database.

9. FUTURE WORK

Work is underway to detect patterns in protein sequences using grammars that are more complex than PROSITE expressions.

References

Anfinsen, C. B. 1973. Principles that govern the folding of protein chains. *Science* 81: 223-230.

Bairoch, A. and Boeckmann, B. 1991. The SWISS-PROT protein sequence data bank: current status. *Nucleic Acids Research* 22(17) 3578–3580.

Bairoch, Amos and Bucher, Philipp. 1994. PROSITE: Recent developments. *Nucleic Acids Research* 22(17) 3583-3589.

Bannister Joe V., Bannister, William.H., and Rotilio Giuseppe. 1987. Aspects of the structure, functions, and applications of superoxide dismutase. CRC Critical Review of Biochemistry 22:111-154.

Bernstein, F. C., Koetzle, T. F., Williams, G. J. B., Meyer, E.J., Jr., Brice, M. D., Rodgets, J. R., Kennard, O. Shimamouchi, T., and Tasumi, M. 1977. The protein data bank: A computer based archival file for macromolecular structures. *Journal of Molecular Biology*. 112: 535-542.

Chang, Tien-Hsien, Arenas, Jaime, and Abelson, John. 1990. Identification of five putative yeast RNA helicase genes. *Proceedings of the National Academy of Sciences U.S.A.* 87:1571–1575.

Creighton, T. E. 1993.*Proteins: Structures and Molecular Properties*. Second Edition. W. H. Freeman.

Dorer, D. R., Christensen, A. C., and Johnson, D. H. 1990. A novel RNA helicase gene tightly linked to the *Triplo-lethal* locus of *Drosophila*. *Nucleic Acids Research* 18(18): 5489–5495.

Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley.

Handley, Simon. 1993a. Automated learning of a detector for α-helices in protein sequences via genetic programming. In Forrest, Stephanie (editor). *Proceedings of the Fifth International Conference on Genetic Algorithms*. San Mateo, CA: Morgan Kaufmann Publishers Inc. Pages 271-278.

Handley, Simon. 1994a. Automated learning of a detector for the cores of α-helices in protein sequences via genetic programming. *Proceedings of the First IEEE Conference on Evolutionary Computation.* IEEE Press. Volume I. Pages 474–479.

Handley, Simon. 1994b. The prediction of the degree of exposure to solvent of amino acid residues via genetic programming. In Altman, Russ, Brutlag, Douglas, Karp, Peter, Lathrop, Richard, and Searls, David (editors). *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*. Menlo Park, CA: AAAI Press. 1994. Pages 156–159.

Handley, Simon. 1995a. Predicting whether or not a nucleic acid sequence is an *E. coli* promoter region using genetic programming. **In Press.**

Handley, Simon. 1995b. Predicting whether or not a 60-base DNA sequence contains a centrally-located splice site using genetic programming. **In Press.**

Handley, Simon, 1995c. Classifying nucleic acid subsequences as introns or exons using genetic programming. **In Press.**

Hodgman, T. C. 1988. A new superfamily of replicative proteins. *Nature* 333:2 2–23 and Errata at Nature 333: 578-578.

Holland, John H. 1975. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Ann Arbor, MI: University of Michigan Press. The second edition is currently available from The MIT Press 1992.

Kinnear, Kenneth E. Jr. (editor). 1994. *Advances in Genetic Programming*. Cambridge, MA: The MIT Press.

Koza, John R. 1989. Hierarchical genetic algorithms operating on populations of computer programs. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann. Volume I. Pages 768-774.

Koza, John R. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: The MIT Press.

Koza, John R. 1994a. *Genetic Programming II: Automatic Discovery of Reusable Programs.* Cambridge, MA: The MIT Press.

Koza, John R. 1994b. *Genetic Programming II Videotape: The Next Generation*. Cambridge, MA: The MIT Press.

Koza, John R. 1994c. Evolution of a computer program for classifying protein segments as transmembrane domains using genetic programming. In Altman, Russ, Brutlag, Douglas, Karp, Peter, Lathrop, Richard, and Searls, David (editors). *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*. Menlo Park, CA: AAAI Press. 1994. Pages 244–252.

Koza, John R. and Andre, David. 1995. Parallel Genetic Programming on a Network of Transputers. Stanford University Computer Science Department technical report STAN-CS-TR-95-1542. January 30, 1995.

Koza, John R., and Rice, James P. 1992 .*Genetic Programming: The Movie*. Cambridge, MA: The MIT Press.

Kyte, J. and Doolittle, R. 1982. A simple method for displaying the hydropathic character of proteins. *Journal of Molecular Biology*. 157:105-132.

Linder, P., Lasko, P., Ashburner, M., Leroy, P., Nielsen, P. J., Nishi, J., Schneir, J., and Slonimski, P. P. 1989. Birth of the D-E-A-D box. *Nature* 337: 121–122.

Ludwig, M. I., Metzger, A. I., Pattridge, R. A., and Stallings, W. C. 1991. Manganese superoxide dismutase from *Thermus thermophilus*: A structural model refined at 1.8 Å resolution. *Journal of Molecular Biology* 219: 335-358.

Matthews, B. W. 1975. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochemica et Biophysica Acta*. 405:442-451.

Richardson, D. C. and Richardson, J. S. 1992. The kinemage: A tool for scientific communication. *Protein Science* 1(1) 3–9.

Rost, B. and Sander, C. 1993. Prediction of protein secondary structure at better than 70% accuracy. *Journal of Molecular Biology*. 232: 584–599.

Smith, T. F. and Waterman, M. S. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology*. Volume 147. Pages 195-197.

Stoddard, B. I., Ringe, D., and Petsko, G. A. 1990. The structure of iron superoxide dismutase from *Pseudomonas ovalis* complexed with the inhibitor azide. *Protein Engineering* 4: 113–199.

Stryer, Lubert. 1988. *Biochemistry*. W. H. Freeman. Third Edition.

Tanese, Reiko. 1989. *Distributed Genetic Algorithm for Function Optimization.* PhD. dissertation. Department of Electrical Engineering and Computer Science. University of Michigan.

Wright, Sewell. 1943. *Genetics* 28. Page 114.