Using Analogy to Predict Functional Regions on Genes

Jon A. Pastor
pastor@prc.unisys.com
(215) 648-2769

Kimberle Koile[1]
kkoile@arris.com
(617) 494-0055

G. Christian Overton
overt@prc.unisys.com
(215) 648-2420

Unisys CAIT
P. O. Box 517
Paoli, PA 19301

June 4, 1992

Abstract

We describe a system to support prediction of previously undetected functional regions (features) on genes using a form of reasoning by analogy. Analogy is a powerful tool in the study of biological systems, since analogous components typically have a biological correspondence based on evolution. We capitalize on this correlation between biotaxonomic relatedness and similarity of genetic structure to drive our predictions. By building a classification hierarchy whose members represent classes of related genes, and classifying individual genes in this hierarchy, we can both form theories about the general structural characteristics of classes of genes, and generate hypotheses about the detailed structure of individual genes.

[1]Current address: Arris Pharmaceutical Corporation, 26 Landsdowne Street, Suite 440, Cambridge, MA 02139

1 Introduction

Reasoning by analogy is a fundamental technique in the study of biological systems, from the level of molecular organization through the level of ecosystems. This method is especially powerful because it is grounded in evolutionary theory: all organisms are, to a greater or lesser degree, related genetically. For this reason, similar biological systems are often **homologous** – derived from common ancestors – rather than merely analogous; for example, the eyes of all vertebrates derive from a common ancestor, and are thus homologous, while bird wings and bat wings evolved separately, and are thus merely analogous. Analogical reasoning between homologous systems is exceptionally powerful and accurate, since their similarities have a deeply-rooted biological basis.

We have developed novel analogical reasoning tools to support research into regulation of gene expression, the translation of DNA sequences into proteins. Understanding this process is a challenge that transcends mere scientific curiosity: diseases such as cancer and AIDS are a result of pre-emption of normal genetic controls. The basic mechanisms of genetics – the encoding of instructions in DNA, and the use of these instructions to synthesize the proteins necessary to cellular function – have been understood for some time. Subtler issues relating to the timing and control of this process are, however, more elusive, and are among the most critical areas of research in Molecular Biology today. In particular, it is now possible to read the genetic code, and thus to determine what chemical a given set of instructions will produce, quite rapidly; determining *when* the instructions will be followed, and *at what rate* the product will be synthesized, have proven to be much less tractable, and these are crucial to normal cellular function.

This paper describes a system that uses analogical reasoning to drive the formation of theories about the genetic structure of related organisms, and the generation of hypotheses about previously unrecognized regulatory elements in the genetic program of organisms in which such elements have not yet been identified experimentally. Our methodology is related to, but differs in significant

ways from, several other models of analogical reasoning described in the Machine Learning literature. More importantly, analogical reasoning has typically been applied to areas such as general problem-solving [13], planning [2], classification [24], and diagnosis [15]; although it is a potent tool for biological reasoning, we are unaware of any previous application of a similar methodology to a practical problem in Molecular Biology. In Section 2, we review other approaches to analogical reasoning, and provide a brief introduction to our domain and the data on which we operate. In Section 3, we describe our system in detail, and in Section 4 we present results of our system's operation. Finally, in Section 5, we compare our approach with the other Machine Learning approaches, and with common practice in DNA sequence analysis; assess the system's contributions and limitations; and note directions for future growth.

2 Background

2.1 Related Machine Learning Research

Our methodology shares much with **Case-Based Reasoning (CBR)** [13, 14], **Conceptual Clustering (CC)** [8, 9], and **Concept Formation (CF)** [12].

CBR is a form of reasoning by analogy in which well-characterized "cases", organized and indexed in a case database, are used as templates to reason about the properties of incompletely characterized but (by some measure) similar cases. A typical CBR system proceeds by first ordering the existing cases with respect to their degree of similarity to the new case, then selecting the most similar cases, and finally adapting the information in the known cases to apply to the new case. CBR methodology closely matches the reasoning process often used by biologists in approaching the study of a new biological system: find a well-understood system similar to the new system, hypothesize the existence of features in the new system based on the features of the known system, then design

and perform experiments to test for those features in the new system. As noted in Section 1, the power of this reasoning process lies in the fact that biological systems are related through evolution: analogous systems are often actually homologs – derived from a common ancestral system. This applies at the level of gene systems and metabolic systems, as well as to whole organisms.

In Conceptual Clustering (CC) and Concept Formation (CF), the emphasis is on the use of abstracted information in classes for prediction of values in new instances. A typical CC/CF system builds a class hierarchy using algorithms similar to those used in cluster analysis; CF differs from CC in that CC builds its hierarchy as a "batch" operation, while in CF the hierarchy is built incrementally [12]. Classes contain partial descriptions, sometimes called **images**, corresponding to attribute values shared by instances of the class. If a new instance is sorted into a given class, and that instance lacks values for some attributes in the image stored at the class, the CC/CF system predicts the stored values for that instance.

2.2 A Primer on Genes

The genetic information of higher organisms is contained in molecules of **deoxyribonucleic acid (DNA)**, which can be regarded as strings, up to several billion characters long, formed from the alphabet {**a,c,g,t**}. For our purposes, a **gene** is a DNA substring consisting of a few hundred to a few hundred thousand characters that specifies or "codes for" a biologically functional molecular product—either a protein or a structural **ribonucleic acid (RNA)**. For protein-coding genes, gene **expression** (Figure 1) is a three-step process in which the DNA template is **transcribed** (copied) into RNA, which is then edited and used as a template for synthesis of a protein in a process called **translation**. A protein is composed of a sequence of **amino acids**, and each three-character "word" in the messenger RNA corresponds a particular amino acid; the identity of the amino acids specified by the RNA sequence, and the order in which they appear, determine what

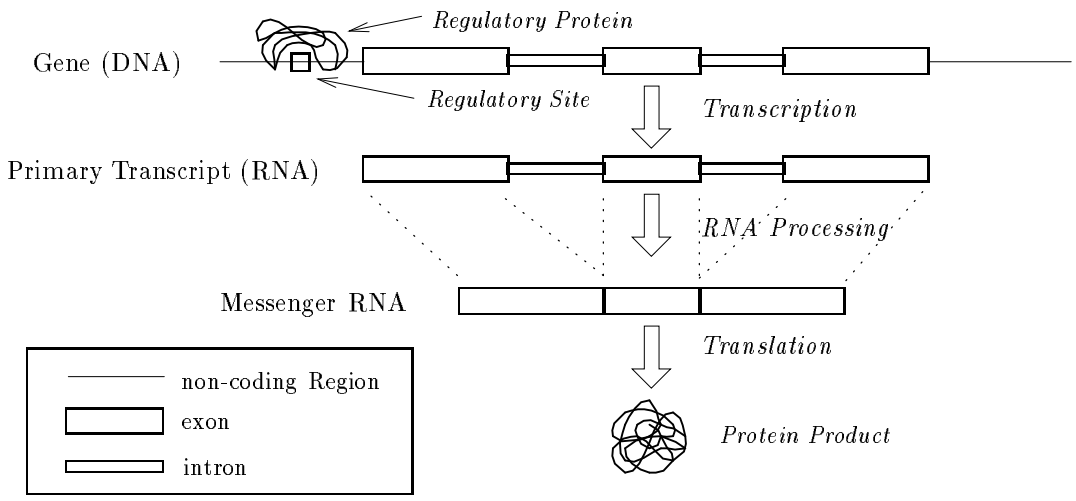

Figure 1: Gene expression for a protein-coding gene

protein is synthesized. DNA subsequences preceding (upstream), following (downstream), and, in some cases, within the coding region are involved in regulating the level and timing of expression of the gene, and thus of its product. These DNA regulatory regions act by binding regulatory proteins in the cellular milieu, and thus provide exquisite control of cellular metabolism and the program of animal and plant development.

Figure 2 is a schematic of a typical animal protein-coding gene, the adult human β-globin gene, showing the organization of its **features** (DNA subsequences of known or presumed biological function). The protein-coding region is contained within the **exon** subsequences of the **primary transcript**, the inital RNA copy of the DNA. **Intron** regions are removed from the primary transcript during an RNA processing step to form the **messenger RNA** which is the actual template for protein synthesis (cf. Figure 1).

In this study, our interest is in the regulatory sequences surrounding the primary transcript. Upstream **promoter** signal regions and downstream **terminator** signal regions (not shown in Figure 2, but sometimes located near the primary transcript/3' flank boundary) respectively act to define the start and stop points of the primary transcript. The class of promoters includes

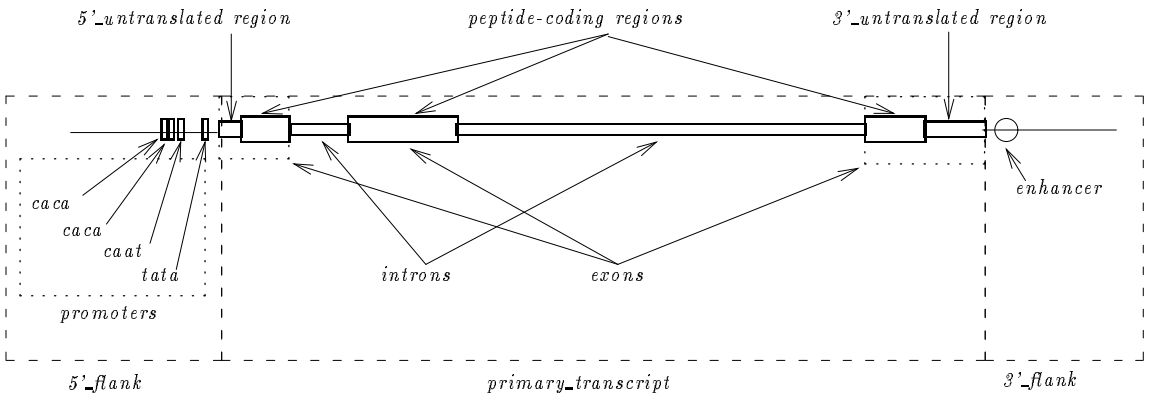

Figure 2: Features of a gene (HUMHBB, the human β-globin gene)

subsequences such as the so-called **TATA, CAAT** and **CACA** boxes, which are relatively well **conserved** (found across a wide range of genes and species), as well as rarer subsequences that restrict gene expression to a specific tissue in an organism. Proper expression of a gene is critically dependent on the organization of the promoter sequences. In addition to the promoter sequences, which are necessarily position-dependent, there are relatively position-independent subsequences, called **enhancers**, that are also tissue-specific. As the name implies, when enhancers are present, the level of expression of the gene is increased, often by as much as two orders of magnitude. The human β-globin gene enhancer is found downstream from the primary transcript. Examples are also known of tissue specific subsequences that are involved in suppression of gene expression (not shown here).

2.3 Source Databaseses

Two primary databases are used by our system: the GenBank biosequence database (**GenBank**) [5], and the PIR protein taxonomy (**PIR**) [7]. GenBank contains detailed information about virtually all genes whose sequences have been determined. Since gene boundaries are typically not known with precision until the DNA sequence has been determined, entries in GenBank do not necessarily list all features of a gene. For this reason, as well as the sheer magnitude of the

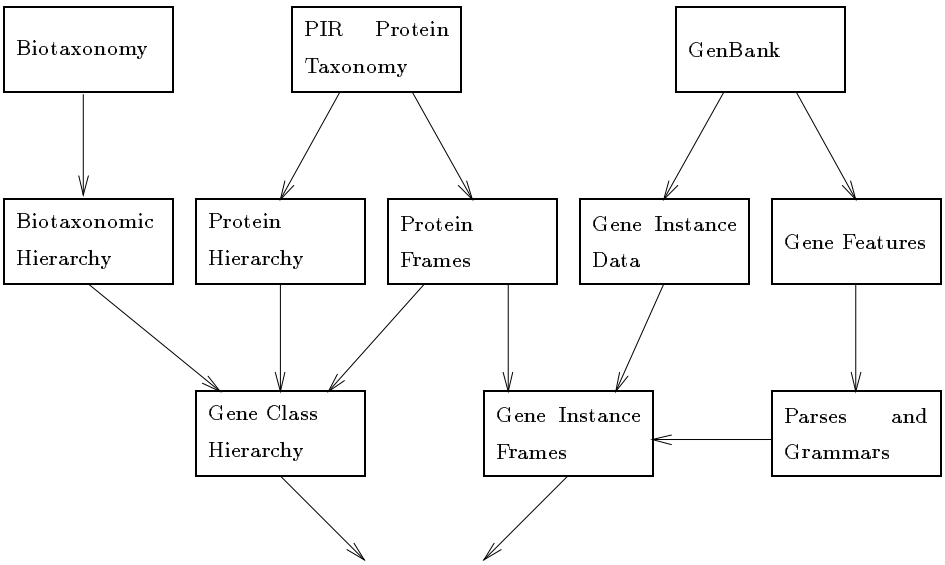

Figure 3: Information flow

database, entries are typically both incomplete and out-of-date; this is one of the problems that our system addresses.

PIR contains detailed information about several thousand protein sequences; in addition, it places each protein in an hierarchy based on genetic distance. Proteins "that can be demonstrated to be evolutionarily related based on their sequences" [7] are hierarchically grouped into superfamilies. Within a superfamily, proteins whose amino-acid sequences differ from one another by less than 50% are grouped in families; those differing by less than 20% are grouped in sub-families; and those differing by less than 5% are grouped under a single entry number.

3 System Description

The overall information flow in our system is shown in Figure 3. First, the source databases – GenBank, the PIR protein taxonomy, and two biotaxonomies – are converted to relational form.

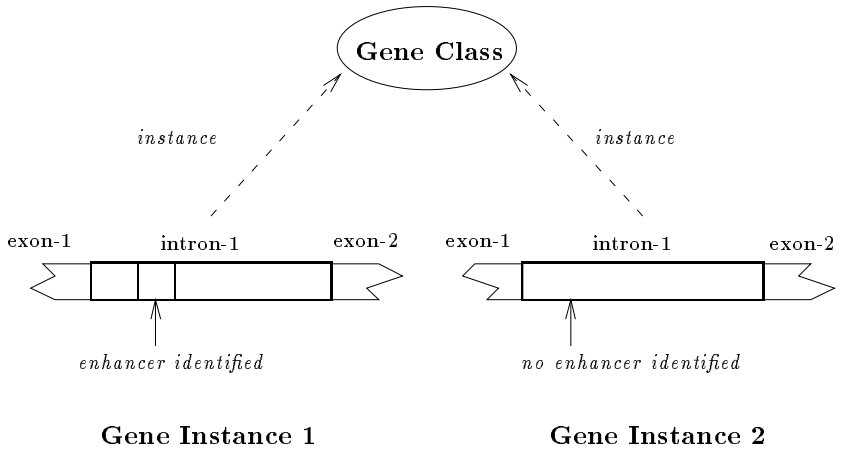

Figure 4: Propagation of features among instances of a class

Next, a classification hierarchy for genes, based on several similarity measures, is constructed, and analogical reasoning is used to support theory formation and hypothesis generation. Both of these activities are guided by classification of new genes into the similarity hierarchy, where the similarity measures are such that it is reasonable to expect structural similarity (preservation of gene features) among members of a similarity class. In Figure 4, for example, it would be reasonable to presume that **Gene Instance 2** has an enhancer in its first intron, despite lack of experimental evidence, since the closely-related **Gene Instance 1** has such an enhancer.

There are thus several distinct stages to the operation of our system:

- preprocessing of source databases
- generation of structural parses and grammars from sets of feature tuples, and creation of frames representing gene instances
- generation of the classification hierarchy, including creation of frames representing gene classes
- theory formation and hypothesis generation
- hypothesis testing

Preprocessing entails much more than simply changing the format of the source data; this will be described in a forthcoming paper. The other stages of our processing are described in the following sections.

3.1 Structural Parse and Grammar Generation

The first significant processing module in our system is a gene structure parser, which takes as input a collection of relational tuples representing features obtained from GenBank for a particular gene, and generates a parse tree representing the structure of the gene (Figure 5a). In addition, it is capable of deducing the existence, location, and size of certain features if they are missing from the GenBank entry; for example, exons are seldom explicitly identified in GenBank. From the structural parse for each gene, a grammar tailored to that parse is constructed, in the form of interval relations (Figure 5b). A frame [18] is then created for each gene instance (Figure 6a), with all available information (including data from GenBank and PIR, the parse, and the grammar) installed in slots of the frame.

We have chosen grammars as the representation for gene structure for several reasons. Grammars are powerful, compact notations that are capable of representing infinitely large languages. Their characteristics and manipulation are well-understood, and it is common to employ the same grammar as both a declarative and a procedural representation of a given language. In our system, the gene structure parser uses a very general grammar procedurally to drive the generation of parses, from which descriptive grammars are then constructed. In addition, grammars can contain meta-rules; in our system, the propagation of grammar elements during both theory formation and hypothesis generation can be controlled by meta-rules where appropriate.

In a conventional grammar, the rule

$$LHS \rightarrow RHS_1, RHS_2, ..., RHS_n$$

states that the nonterminal symbol LHS consists of the symbols RHS_i; it is presumed that RHS_j and RHS_{j+1} are adjacent – that no other symbol can occur between them – and that they do not overlap. In our domain, the RHS elements are gene features, which are frequently *not* adjacent, and may even overlap. (A general discussion of applications of grammars to the analysis of DNA

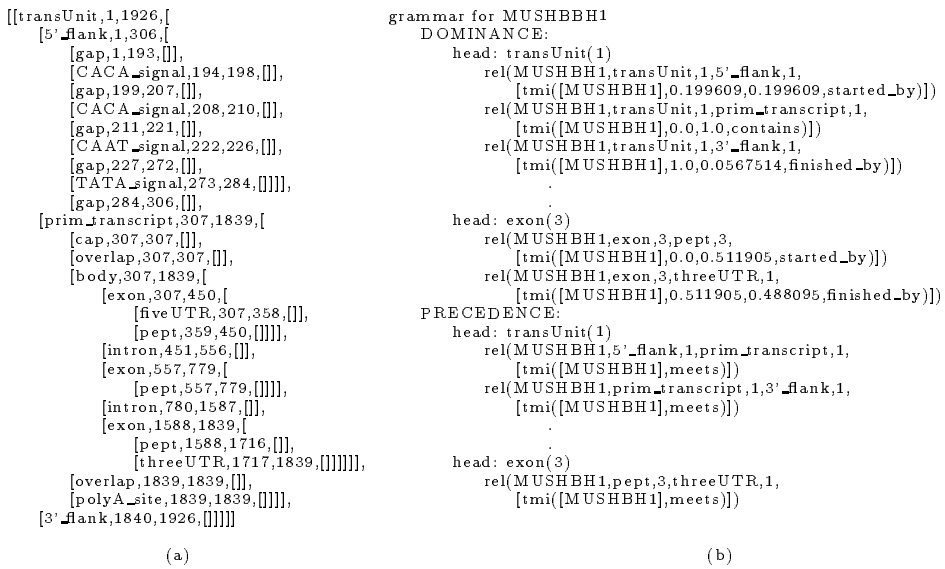

Figure 5: Parse and grammar for MUSHBBH1 (mouse embryonic β-globin gene H1)

can be found in [22].) Furthermore, where the RHS_i of a typical grammar rule may be presumed to be exhaustive, we cannot presume the same of our grammar rules: in fact, our system presumes that undetected features *do* exist among those already known. For these reasons, we have based our grammars on the ID/LP formalism [11], which separates **dominance** (part-whole) and **precedence** (order of parts) information; we have also added explicit information about the interval relationships among features, based on the interval calculus of Allen [1].

Our grammar elements have the form:

```
rel(GeneId,Feature1,Occurrence1,
          Feature2,Occurrence2,TMinfo,Relation)
```

where `Feature1` and `Feature2` are the features whose relationship is described by the clause, `Occurrence1` and `Occurrence2` index multiple occurrences of the same feature (e.g., exon, intron) and `Relation` is either `dominance` or `precedence`; note that in Figure 5b the `Relation`s are not shown. In the example above, dominance relations hold between LHS and each of the RHS_i; precedence relations hold pairwise among the RHS_i. The `TMinfo` field contains information concerning the sources for the grammar element (i.e., the instance(s) from which the relation is derived), its normalized position and size, and the interval relation (**contains**, **meets**, etc.) that holds between

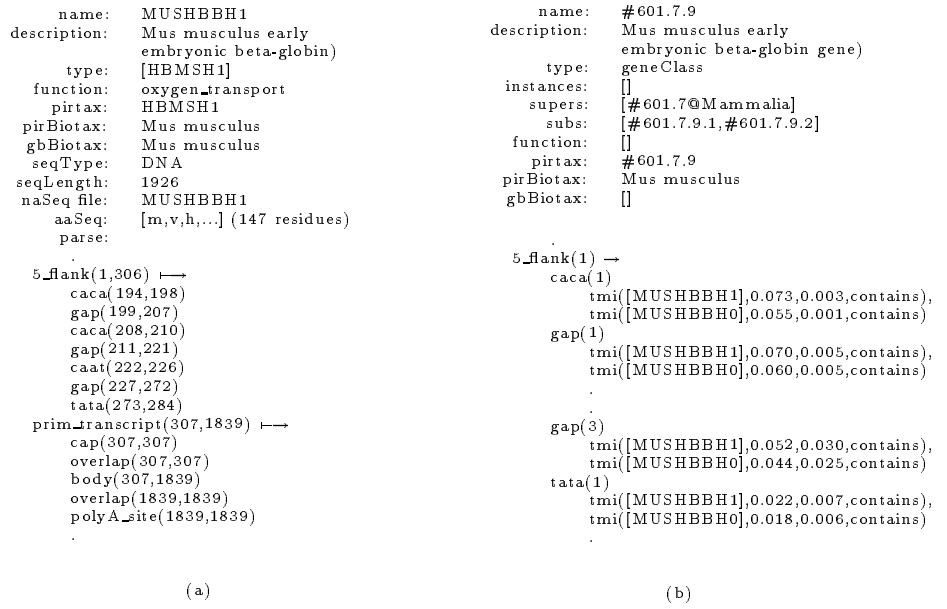

Figure 6: Instance and class frames

the two features. Normalized positions and sizes, for all features but 3' and 5' flanks, are computed relative to the closest containing (dominating) feature; since the flanks can be of arbitrary size, features within them are normalized to the primary_transcript length.

3.2 Classification Hierarchy Generation

The class hierarchy in which genes are organized is structured as a tree of frames, with information about gene classes and instances stored in slots of the appropriate frames. The only attributes presently used in constructing the tree are PIR class and biotaxonomic class; these taxonomic attributes are, in Lebowitz's [17] terms, the "predictive" attributes (the ones upon which predictions are based), while gene features are the "predictable" attributes (the ones predicted). With respect to the taxonomic attributes, the tree is a generalization-specialization hierarchy in which subsumption is strictly enforced, as in frame systems such as KNET [10] and KL-ONE [3]; we are therefore licensed to regard theories housed at a class as being applicable to specializations and instances of that class. This is a crucial point: enforcing subsumption guarantees that the structure of our knowledge base mirrors the organization of the biological knowledge that we are

modeling.

The hierarchy is constructed (Figure 7) by refining the PIR protein taxonomy; at present, the only other distance measure used is biotaxonomic relatedness, but the general algorithm extends in a straightforward fashion to additional dimensions of refinement. The effect of building the hierarchy in this manner is that protein sequence similarity is given highest weight in similarity determinations, followed by biotaxonomic distance; this is an heuristic, developed by the molecular biologist in our group. The hierarchy is realized as a collection of frames representing gene classes (Figure 6b). The algorithm used in generating the hierarchy is shown as Figure 8.

Classification of instances in the gene hierarchy is based strictly on the taxonomic attributes – that is, on genetic distance, as measured by protein-sequence differences, and biotaxonomic relatedness. Each gene instance can be uniquely placed in the class hierarchy, as an instance of the gene class having the same biotaxonomic and PIR classes. Gene instances appear only as leaf nodes and must be instances of exactly one gene class. Given this organization of the knowledge base, features present in any instance of a gene class can reasonably be presumed to be present on other instances of the class, unless this is specifically contradicted.

3.3 Theory Formation and Hypothesis Generation

Once a gene instance is classified, theory formation can begin. Theory formation is implemented by combining gene class grammars. As mentioned in Section 3.1, the RHS of a rule in a conventional grammar is presumed to be a complete enumeration of the components of the LHS. Thus, the two rules

$$A \rightarrow B, C, E$$

and

$$A \rightarrow B, D, E$$

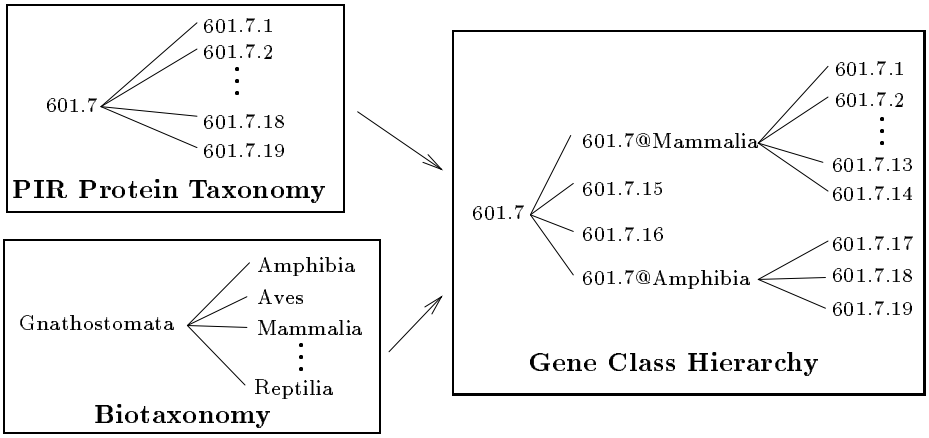

Figure 7: Generation of the classification hierarchy

would be incompatible, and could only be combined by disjunction. The grammar rules in our system, however, are almost always based on incomplete information, given the relative scarcity of data on the features in which we are most interested (i.e., promoters and enhancers). Our fundamental premise is that, between sufficiently similar genes, it is legitimate to *merge* the two rules above into a single rule:

$$A \rightarrow B, \{C, D\}, E$$

where the notation $\{C,D\}$ indicates that the order of C and D is not determined, but that both are present.

In addition to merging grammars during theory formation, our system also builds **consensus sequences** where appropriate. Even features that are relatively well-conserved across species can vary considerably in size and composition, but it is generally possible to capture the commonality or consensus among the various sequences that have been shown empirically to be instances of the same feature. This consensus can take the form of an explicit enumeration of the actual sequences, a regular-expression-like string, or some statistical summary of frequency-of-occurrence of the various characters in the sequence. At present, we are storing consensus sequences as the union of all subsequence strings actually identified in the literature as corresponding to a given

```
procedure label_nodes(node_list)
    for each node in node_list
        if node is not a protein [i.e., node is non-leaf]
            label_nodes(node.children)
            set node's biotaxonomic class to "NONE"
            for each child in node.children
                if node's biotaxonomic class is "NONE"
                        or node's biotaxonomic class is a subclass of child's biotaxonomic class
                    set node's biotaxonomic class equal to child's biotaxonomic class
end

procedure group_nodes(node_list)
    for each node in node_list
        set bio_subs equal to subclasses of node's biotaxonomic class
        for each child in node.children
            for each sub_class in bio_subs
                set sub_class.pir_nodes to "EMPTY"
                if child's biotax is a subclass of sub_class
                    push child onto sub_class.pir_nodes
        for each sub_class in bio_subs
            new_node = create(node)
            for each node_to_reparent in sub_class.pir_nodes
                change node_to_reparent's parentage from node to new_node
            make new_node a child of node
end

procedure generate_class_hierarchy(root_node)
    label_nodes(list(root_node))
    group_nodes(list(root_node))
end
```

Figure 8: Hierarchy generation algorithm (pseudocode)

feature.

Determination of the degree of similarity necessary to permit merging grammars is a critical issue in the operation of our system. Gene classes high in the classification hierarchy correspond to broad groups of genes, with considerable variation in the taxonomic attributes; gene classes at lower levels of the hierarchy represent groups of genes that are much more similar. With respect to an instance being classified, one can – given some criteria for acceptable levels of similarity – locate the highest parent class meeting these criteria; we will refer to this parent class as the **pivot**. The higher the pivot is in the classification hierarchy, the greater the possible distance between instances scoped by the pivot; in our present model, this distance corresponds to evolutionary relatedness.

Choice of the pivot point has two implications, one for the nature of the general grammars stored at gene classes, and the other for the mechanics of analogy searches. With respect to grammar formation at gene classes, the pivot is precisely that point at which merging of grammars

```
function propagate_up(gene_class_or_instance)
    if gene_class_or_instance.parent exceeds criterial distance
        disjoin all grammars between gene_class_or_instance
                        and the root of the gene class hierarchy
        return(gene_class_or_instance)
    else
        merge gene_class_or_instance.grammar with
                        gene_class_or_instance.parent.grammar
        for each feature described in gene_class_or_instance.grammar
                            for which a consensus sequence is stored
            combine gene_class_or_instance.feature.sequence with
                            gene_class_or_instance.parent.feature.sequence
        return(propagate_up(gene_class_or_instance.parent))
end

procedure propagate_down(gene_class)
    for each child in gene_class.children
        for each relation in gene_class.grammar
            if relation is not in child.grammar
                add relation to child.grammar
            if relation describes a feature for which a consensus sequence is stored
                            and consensus sequence stored at gene_class
                                        is broader than that stored at child
                merge child's consensus sequence for feature with gene_class's
        if child is a gene class
            propagate_down(child)
end

procedure form_theory_gen_hypotheses(gene_instance)
    gene_instance.grammar = derive_grammar(gene_instance.parse)
    pivot = propagate_up(gene_instance)
    propagate_down(pivot)
end
```

Figure 9: Theory formation and hypothesis generation algorithm (pseudocode)

ceases to be legitimate, and we must combine grammars by disjunction. With respect to the mechanics of analogy search, tolerating a greater distance corresponds to weakening the criteria for similarity upon which our processing is based. The criterial distance for the pivot point is an explicit parameter, subject to user control. This permits an iterative style of search, providing the ability to initiate search with very strict similarity criteria, which may then be relaxed if necessary. Note that an overly restrictive criterial distance will have the same effect on the behavior of our system as noisy data does on a CC/CF system: while our classification hierarchy will not change, analogy searches will return few, if any, predictions, as though the class hierarchy had been overfit [21] to noisy data.

Hypothesis generation consists of projecting grammar elements from a gene class grammar down onto instances of the corresponding gene class. Grammar elements propagated in this manner to

gene instances correspond to hypothesized features. Positions are projected using the normalized positions stored in grammars.

The algorithm used for both theory formation and hypothesis generation is shown in Figure 9. In practice, hypothesis generation is done dynamically, on demand, thus reducing the need for truth maintenance.

3.4 Hypothesis Testing

Hypotheses specify both the identity and the location of proposed features. Where a feature's location and size are identical in all similar genes, it is possible to give absolute start and end positions; this is typically not the case, and locations can therefore be predicted only relative to other features at the same level of the grammar, with a projected absolute location and size based on normalized location and size. Since both relative and absolute locations are known for all features in gene instances (the former via the grammar, the latter via the parse), some hypotheses can be ruled out; for example, a predicted feature that would overlap a known feature can usually be eliminated as infeasible. In other cases, domain knowledge can be used to support or invalidate an hypothesis. Finally, some hypotheses can be supported or invalidated fairly reliably via sequence analysis; for example, an hypothesized TATA or CAAT signal can be located by searching the sequence in the predicted region for the appropriate string.

All of these techniques notwithstanding, some hypotheses cannot be invalidated by any means short of experimental analysis; in fact, while computational techniques may permit invalidation, true confirmation can *only* be accomplished experimentally. Consequently, unconfirmed hypotheses may persist in the knowledge base for indeterminate periods of time, and may in turn contribute to further hypotheses. For this reason, because of the possibility of changes to the underlying databases (e.g., GenBank updates), and to support explanation of the derivation of hypotheses,

some form of truth maintenance [6] is necessary. Truth maintenance in a database the size of GenBank is somewhat problematic (cf. Section 5), but we are taking pains to preserve sufficient information during theory formation and hypothesis generation to support limited forms of truth maintenance and explanation.

We have used the consensus sequences built during theory formation to confirm predictions (see Section 4). As mentioned in Section 3.3, consensus sequences are stored as the union of all subsequence strings actually occurring in instances of a particular feature in a particular class of genes. During the consensus sequence search, the sequence being searched is scanned for occurrences of any of these subsequences. Matches need not be exact, and partial matches can be ranked by quality; the quality of a match can be measured in various ways, but we use a simple count of the number of matching characters.

4 Results

For our initial implementation, we have restricted our attention to the globin gene family, a group of genes that are functionally similar, and whose (protein) sequences are fairly similar as well. Of all globin genes, we are considering mammalian and avian hemoglobin and myoglobin, with token representation of plant globins (leghemoglobin); all of our testing has been performed on β-/ϵ-globin genes (PIR family 601.7). Our system is tested by removing features from some genes, and then classifying these and other genes and hypothesizing features as described above. The system should hypothesize the removed features for the edited genes, in the appropriate locations.

As mentioned in Section 1, few entries in GenBank have any promoter features listed. In order to obtain test data, we performed a literature search, located publications [4, 23] characterizing promoters in the β-/ϵ-globin family, and manually altered our version of the GenBank database to reflect the added features.

```
          -predicted-  best match   --actual---
feature   start  len   start  len   start  len    target seq
--------- ----- -----  ----- -----  ----- ----- -------------
space(1)    415   306     --    --                      ----
caca(1)     110     5    113     5    113     5         CACCC
gap(1)      105     9     --    --                      ----
caca(2)      96     3     98     4     99     3          CAgA
gap(2)       93    11     --    --                      ----
caat(1)      81     5     85     5     85     5         CCAAT
gap(3)       76    44     --    --                      ----
tata(1)      32    12     34    12     34    12  AAGAATAAAAGG
gap(4)       25    25     --    --                      ----

feature   sources               consensus sequence
--------- --------------------- ----------------------------------
space(1)  [GOTHBBEII,GOTHBBEI]
caca(1)   [GOTHBBEII,GOTHBBEI]  [caccc]
gap(1)    [GOTHBBEII,GOTHBBEI]
caca(2)   [GOTHBBEII,GOTHBBEI]  [caca,cca]
gap(2)    [GOTHBBEII,GOTHBBEI]
caat(1)   [GOTHBBEII,GOTHBBEI]  [ccaat]
gap(3)    [GOTHBBEII,GOTHBBEI]
tata(1)   [GOTHBBEII,GOTHBBEI]  [aagaataaaagg,aggaataaaagg]
gap(4)    [GOTHBBEII,GOTHBBEI]

feature   ranked match list
--------  ----------------------------------------------------
caca(1)   [r(5,CACCC,113),r(5,CACCC,273)]
caca(2)   [r(3,CAgA,98),r(3,CCA,99),r(3,CcCA,100),r(3,CACc,113),...]
caat(1)   [r(5,CCAAT,85)]
tata(1)   [r(12,AAGAATAAAAGG,34)]
```

Figure 10: Prediction of features for mouse β-globin MUSHBBH1

Our results have been quite satisfactory, although limited by the scarcity of data. For example, in Figure 10, predictions for promoters in the mouse β-globin gene MUSHBBH1 based on promoters in goat ϵ-globin genes GOTHBBEI and GOTHBBEII are reasonably accurate, despite the relatively large evolutionary distance between goats and mice.

The first grouping in Figure 10 gives, for each feature:

- its predicted start position (number of characters upstream from the start of the primary transcript) and length;
- the start and length of the best match found by the consensus-sequence search in the actual sequence for MUSHBBH1;
- the start and length of the feature as described in the literature;
- and the sequence identified by the consensus-sequence search.

The second grouping gives, for each feature, the genes from which the feature was identified, and (where appropriate) the consensus sequence for the feature. The final grouping gives, for each feature for which a consensus-sequence search was performed, an ordered list of matches found by the search; the first member of each list is the match that is shown under "best match" in the first grouping. Note that in the case of the second CACA signal, the target sequences "CAgA" and "CCA" were ranked equally, but "CAgA" was preferred, even though it contains one incorrect character; this is because the matching algorithm for the consensus sequence search presently ranks matches by number of matched characters, and weights physical proximity higher than mismatches, and "CAgA" is closer to the predicted position than "CCA".

5 Discussion

Our work exists within two distinct contexts: Artificial Intelligence, primarily the sub-area of Machine Learning; and Molecular Biology. In this section, we compare our approach with other approaches to similar problems, and evaluate our contributions. We then explore the limitations of the system as currently implemented, and suggest areas for future growth.

5.1 Machine Learning

The most significant difference between conventional CBR and our system lies in the fact that gene instances, which might be expected to correspond to cases, are typically so incomplete as to be uninformative individually. The true cases in our system are, in fact, the gene classes, into which information from several gene instances has been merged. In this respect, our approach shares much with Conceptual Clustering (CC) and Concept Formation (CF), in which the emphasis is on the use of abstracted information in classes for prediction of values in new instances. Unlike most CC/CF systems, but like Fisher's Cobweb system [8], ours is polythetic (classifies based on multiple attributes) and does flexible prediction (prediction of multiple attributes).

While the manner in which we use information stored at classes in our hierarchy is similar to that used by CC and CF systems, the manner in which our hierarchy is built differs substantially. As described in Section 3.2, our hierarchy is built statically; class descriptions, our equivalent of images, are then built by placing instances in the hierarchy and inducing generalizations of attribute values. Also, unlike CF systems, classification of instances results not in a restructuring of the class hierarchy, but in modification of the the intensional descriptions of classes. In terms of the sub-processes described by Fisher and Langley [9], our system derives its class hierarchy and classifies instances in a simple, deterministic fashion, but has a relatively sophisticated way of "assigning conceptual descriptions to object classes."

5.2 Molecular Biology

We have made two principal contributions to DNA sequence analysis. First, there is presently no systematic theory describing the general structural characteristtics of classes of genes; we provide a first step toward such a theory. In addition, an entry in GenBank cannot list features (e.g., promoters, enhancers) that had not been characterized at the time of submission of the entry; we

permit identification of such features, so that older entries can be updated. The same mechanisms used to predict new features can also be used to validate data for existing features: GenBank inevitably contains erroneous data, despite conscientious and thorough manual checking. Conflicts between an hypothesized feature and the existing features of a gene may be due to errors in the original entry.

Our approach has several advantages over traditional techniques for locating gene features, which are typically oriented toward identifying and searching for consensus sequences in large populations of genes. These techniques rely on statistical analysis of the sequence database, but since many regulatory features are similar only among related genes, there is considerable variation in the overall population; this variation, coupled with small sample sizes, leads to relatively inaccurate consensus sequences. Our classification-based search strategy, relying on a correlation between biotaxonomic/protein product similarity and structural/nucleotide sequence similarity, dramatically reduces the space of sequences that must be searched. Where traditional methods derive consensus sequences based on all examples of a particular feature, without respect to their biological sources, we are able to obtain relatively accurate consensus sequences, since restricting the population to a set of related genes reduces the intrinsic variation in the set of signals to be averaged.

5.3 Limitations of the Current System

Some aspects of the present system are based on simplifying assumptions. For example, the manner in which evidence is combined in the present system is very crude: all features scoped by the pivot are given equal weight. Since the strength of an hypothesis can be measured by the distance between the source of the appropriate grammar element and the gene instance to which the hypothesis applies, there are obvious ways in which combination of evidence can be improved. Also, our classification hierarchy considers only two dimensions on which similarity can be measured, and we

have adopted a relatively simple measure for the criterial distance discussed in Section 3.3.

In addition, due the number of genes and features stored, and the consequent size of the databases involved, full implementation of an adequate truth maintenance system will require the use of secondary storage. Furthermore, while information is stored that could provide the basis for an explanation capability, no such feature has been implemented.

Finally, only a few heuristics for hypothesis generation and pruning have been incorporated; this is likely to be a major area of expansion.

5.4 Future Directions

As mentioned above, one area for substantial expansion will be in truth maintenance. In addition, our system must be extended to handle explicit negative evidence, since experimental evidence *against* the presence of a particular feature must be recorded, and must unequivocally dominate any evidence in favor of the feature.

Another area of expansion is detection of second- and higher-order patterns (e.g., correlations between two distinct patterns). Conventional statistical techniques lack the ability to model higher-order patterns in the sequences being analyzed. Neural networks, with their ability to extract patterns from multidimensional data, might be applied to some advantage in this area. They have been used in several other DNA-analysis applications, such as prediction of splice junctions in mRNA [16, 19], location of translational initiation sites in bacterial genomes [25, 26], prediction of secondary structure from mRNA [20], and identification of coding regions in prokaryotic and eukaryotic genomes [16].

References

[1] James F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, Nov 1983.

[2] Kevin D. Ashley and Ewina L. Rissland. Compare and Contrast, A Test of Expertise. In Janet Kolodner, editor, *Proceedings of the DARPA Case-Based Reasoning Workshop*, pages 31–36. DARPA, May 1988.

[3] R. J. Brachman and J. G. Schmolze. An Overview of the KL-ONE Knowledge Representation System. *Cognitive Science*, 9:171–216, 1985.

[4] Francis S. Collins and Sherman M. Weissman. The Molecular Genetics of Human Hemoglobin. *Progress in Nucleic Acid Research*, 31:315–462, 1984.

[5] DDBJ, EMBL Data Library, and GenBank Staffs. The DDBJ/EMBL/GenBank Feature Table Definition, Version 1.01. Los Alamos National Laboratory Report, Sept 1987.

[6] J. Doyle. A Truth Maintenance System. *Artificial Intelligence*, 12(3), 1979.

[7] W. C. Barker et. al. Protein Sequence Database of the Protein Identification Resource (PIR). CODATA Exchange Format Specification Document CXFSD-0687, 1987.

[8] Douglas H. Fisher. Noise-Tolerant Conceptual Clustering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 825–830. Morgan Kaufmann, August 1989.

[9] Douglas H. Fisher and Pat Langley. Approaches to Conceptual Clustering. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 691–697. Morgan Kaufmann, 1985.

[10] Michael W. Freeman. KNET: An Extended SI-Net Formalism for Knowledge Representation Systems. Technical Report TR 80-1, Burroughs Corporation, January 1980.

[11] Gerald Gazdar, Ewan Klein, and Geoffrey K. Pullum. *General Phrase Structured Grammar.* Basil Blackwell, Oxford, U.K., 1985.

[12] John A. Gennari, Pat Langley, and Doug Fisher. Models of Incremental Concept Formation. *Artificial Intelligence*, 40:11–61, 1989.

[13] Janet L. Kolodner. Experiential Processes in Natural Problem Solving. Technical Report GIT-ICS-85/23, School of Information and Computer Science, Georgia Institute of Technology, Atlanta, GA 30332, 1985.

[14] Janet L. Kolodner, Robert L. Simpson, Jr., and Katia Sycara-Cyranski. A Process Model of Case-Based Reasoning in Problem Solving. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 284–290. Morgan Kaufmann, 1985.

[15] Phyllis Koton. Using Experience in Learning and Problem Solving. Technical Report MIT/LCS/TR-441, Massachusetts Institute of Technology, March 1989.

[16] Alan S. Lapedes. Applications of Neural Networks and Other Machine Learning Algorithms to DNA Sequence Analysis. In G. Bell and T. Marr, editors, *Computers and DNA*, Reading, MA, 1990. Santa Fe Institute, Addison-Wesley.

[17] Michael Lebowitz. Generalization from Natural Language Text. *Cognitive Science*, 7(1):1–40, 1983.

[18] Marvin Minsky. A Framework for Representing Knowledge. MIT AI Lab Memo 306, MIT, 1974.

[19] K. Nakata, M. Kanehisa, and D. DeLisi. Prediction of splice junctions in mRNA sequences. *Nucleic Acids Research*, 13:5327–5340, 1985.

[20] Ning Qian and T. J. Sejnowski. Predicting the secondary structure of globular proteins using neural network models. *Journal of Molecular Biology*, 202:865–884, 1988.

[21] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1:81–106, 1986.

[22] D. B. Searls. Investigating the Linguistics of DNA with Definite Clause Grammars. In E. Lusk and R. Overbeek, editors, *Logic Programming: Proceedings of the North American Conference, Vol. 1*, volume 1, pages 189–208. MIT Press, 1989.

[23] George Stamatoyannopoulos, Arthur W. Nienhuis, Philip Leder, and Philip W. Majerus. *The Molecular Basis of Blood Diseases*. W.B. Saunders Company, Philadelphia, PA, 1987.

[24] Robert E. Stepp III and Ryszard S. Michalski. Conceptual Clustering: Inventing Goal-Oriented Classifications of Structured Objects. In Ryszard S. Michalski, Jaime G. Carbonell, and Tom M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 2, chapter 17, pages 471–498. Morgan Kaufmann, 1986.

[25] G.D. Stormo, T.D. Schneider, and L.M. Gold. Characterization of translational initiation sites in E. coli. *Nucleic Acids Research*, 10:2971–2996, 1982.

[26] G.D. Stormo, T.D. Schneider, L.M. Gold, and A. Ehrenfeucht. Use of the 'perceptron' algorithm to distinguish translational initiation sites in E. coli. *Nucleic Acids Research*, 10:2997–3010, 1982.