Time for AI and Society

PROCEEDINGS OF THE AISB'00 SYMPOSIUM ON ARTIFICIAL INTELLIGENCE IN BIOINFORMATICS

17th-20th April, 2000
University of Birmingham

AISB'00 Convention

17th-20th April 2000

University of Birmingham
England

Proceedings of the AISB'00 Symposium on

Artificial Intelligence in Bioinformatics

Published by

The Society for the Study of Artificial Intelligence and the Simulation of Behaviour

United Kingdom

http://www.cogs.susx.ac.uk/aisb/

ISBN 1 902956 12 X

Printed at the University of Birmingham, Edgbaston, Birmingham B15 2TT, England.

Contents

The AISB'00 Convention

The millennial nature of current year, and the fact that it is also the University of Birmingham's centennial year, made it timely to have the focus of this year's Convention be the question of interactions between AI and society. These interactions include not just the benefits or drawbacks of AI for society at large, but also the less obvious but increasingly examined ways in which consideration of society can contribute to AI. The latter type of contribution is most obviously on the topic of societies of intelligent artificial (and human) agents. But another aspect is the increasing feeling in many quarters that what has traditionally been regarded as cognition of a single agent is in reality partly a social phenomenon or product.

The seven symposia that largely constitute the Convention represent various ways in which society and AI can contribute to or otherwise affect each other. The topics of the symposia are as follows: Starting from Society: The Application of Social Analogies to Computational Systems; AI Planning and Intelligent Agents; Artificial Intelligence in Bioinformatics; How to Design a Functioning Mind; Creative and Cultural Aspects of AI and Cognitive Science; Artificial Intelligence and Legal Reasoning; and Artificial Intelligence, Ethics and (Quasi-)Human Rights. The Proceedings of each symposium is a separate document, published by AISB. Lists of presenters, together with abstracts, can be found at the convention website, at http://www.cs.bham.ac.uk/~mgl/aisb/.

The symposia are complemented by four plenary invited talks from internationally eminent AI researchers: Alan Bundy ("what is a proof?"- on the sociological aspects of the notion of proof); Geoffrey Hinton ("how to train a community of stochastic generative models"); Marvin Minsky ("an architecture for a society of mind"); and Aaron Sloman ("from intelligent organisms to intelligent social systems: how evolution of meta-management supports social/ cultural advances"). The abstracts for these talks can be found at the convention website.

We would like to thank all who have helped us in the organization, development and conduct of the convention, and especially: various officials at the University of Birmingham, for their efficient help with general conference organization; the Birmingham Convention and Visitor Bureau for their ready help with accommodation arrangements, including their provision of special hotel rates for all University of Birmingham events in the current year; Sammy Snow in the School of Computer Science at the university for her secretarial and event-arranging skills; technical staff in the School for help with various arrangements; several research students for their volunteered assistance; the Centre for Educational Technology and Distance Learning at the university for hosting visits by convention delegates; the symposium authors for contributing papers; the Committee of the AISB for their suggestions and guidance; Geraint Wiggins for advice based on and material relating to AISB'99; the invited speakers for the donation of their time and effort; the symposium chairs and programme committees for their hard work and inspirational ideas; the Institue for Electrical Engineers for their sponsorship; and the Engineering and Physical Sciences Research Council for a valuable grant.

John Barnden & Mark Lee

Artificial Intelligence in Bioinformatics

Andrew C.R. Martin*; Dave W. Corne†

*Parallel Emergent & Distributed Architectures Laboratory, Dept. of Computer Science,
University of Reading, PO Box 225, Whiteknights, Reading RG6 6AY, UK;
D.W.Corne@reading.ac.uk
†Division of Cell & Molecular Biology, School of Animal and Microbial Sciences,
University of Reading, PO Box 228, Whiteknights, Reading RG6 6AJ, UK;
A.C.R.Martin@reading.ac.uk

The term 'bioinformatics' has been around for several years to refer solely to analysis of protein and DNA sequence data. In the last 6 years, the science has been broadened to encompass every aspect of computational biology from DNA sequence and evolution analysis through amino acid sequence analysis (general analysis, looking for patterns, motifs and fingerprints or looking for distant evolutionary relationships) to protein structure analysis and prediction. It also covers such diverse topics as automated data collection, text analysis and data storage.

As such, bioinformatics is at the boundary between biology and computer science. Most of the tools used routinely have been developed by scientists from a biological sciences background. For example, the developers of the first application of dynamic programming to protein sequence alignment were clinicians (Needleman & Wunsch (1970) "A general method applicable to the search for similarities in the amino acid sequence of two proteins", *J. Mol. Biol.* **48**: 443–453). They had no computer science training and came up with the dynamic programming method independently as a way of performing global alignment between two sequences. The method was later formalised and adapted to local alignment by Smith and Waterman (1981, "Identification of common molecular subsequences", *J. Mol. Biol.* **147**: 195–197). However, the two disciplines are now starting to realise that they can learn a lot from eachother and are starting to share their knowledge more intimately.

The amount of biological data available is increasing exponentially. In particular, the volume of sequence data (DNA and protein) and of protein structural data doubles approximately every 18 months. Thus it is becoming more and more important that rigorous computational methods are available to store, process, clean and analyse these data. In addition, this presents vast opportunities for the application of a wide range of techniques from computer science including artificial intelligence.

This symposium focuses on the application of artificial intelligence and related techniques to bioinformatics issues. The range of papers presented illustrates the broad diversity of techniques being applied: support vector machines, genetic algorithms, constraint programming, inductive logic programming, neural networks, database analysis, information extraction and information agents. The applications are equally diverse: drug design, drug target identification, protein structure comparison, genome annotion, protein sequence annotation, data extraction from the literature, structural analysis and determination of protein structure folding signatures.

The programme committee are thanked for their valuable services: Chris Cannings (University of Sheffield, UK), Thomas Dandekar (EMBL, Heidelberg, Germany), Alex Gammerman (Royal Holloway and Bedford New College, UK), David Gilbert (Computer Science, City University, UK), Arun Holden (School of Biomedical Sciences, University of Leeds, UK), David Jones (Biochemistry, Brunel University, UK), Nigel Martin (Computer Science, Birkbeck College, UK), Graham Megson (School of Computer Science, Cybernetics and Electronic Engineering, University of Reading, UK), Ray Paton (Computational Biology Group, University of Liverpool, UK), Shail Patel (Unilever PLC, UK), Vic Rayward-Smith (School of Information Systems. University of East Anglia, UK), Martin Reese (University of California, Berkeley, USA), Rob Russell (Smith Kline Beecham, UK), Richard Sibly (School of Animal and Microbal Sciences, University of Reading, UK), Richard Tateson (British Telecommunications PLC, UK), David Westhead (Biochemistry, University of Leeds, UK),

Drug Design by Machine Learning: Support Vector Machines for Pharmaceutical Data Analysis

Robert Burbidge; Matthew Trotter; Sean Holden; Bernard Buxton
University College London; Gower Street, London WC1E 6BT
robert.burbidge@ptp.sira.co.uk; m.trotter@cs.ucl.ac.uk; s.holden@cs.ucl.ac.uk; b.buxton@cs.ucl.ac.uk

Abstract

In this paper, we show that the support vector machine (SVM) classification algorithm, a recent development from the machine learning community, proves its potential for QSAR analysis. In a benchmark test, the SVM is compared to several machine learning techniques currently used in the field. The classification task involves predicting the reduction of dihydrofolate reductase by pyrimidines, using data obtained from the UCI machine learning repository. Three artificial neural networks, a radial basis function network, a C5.0 decision tree and a nearest neighbour classifier are all outperformed by the SVM. The SVM is significantly better than all of these, bar a manually capacity-controlled neural network, which takes considerably longer to train.

1 Introduction

A recent classification algorithm from the machine learning community is introduced and applied to a well-known problem in the field of drug discovery. As a control, the algorithm is compared to several intelligent classification techniques that are currently used to tackle the problem. In this paper we first describe QSAR analysis, a technique used by pharmaceuticals companies in the drug discovery process. Following this, we introduce the support vector machine (SVM), a new learning algorithm for classification. After a brief theoretical argument on the advantages of using SVMs for QSAR analysis, we present empirical evidence for this approach. Finally, we make some conclusions.

2 QSAR Analysis

Quantitative structure-activity relationship (QSAR) analysis represents an essential part of the drug discovery process. It also presents an extremely challenging problem to the field of Intelligent Systems and one that, if solved successfully, has the potential to provide significant economic benefit. In this paper, we present evidence that a recent state-of-the-art machine learning technique considerably outperforms several of its competitors when applied to such problems.

The underlying assumption behind QSAR analysis is that the variation of biological activity within a group of compounds can be correlated with the variation of their respective structural and chemical features. That is, there exists a rule or function that predicts a molecule's activity from the values of its physicochemical descriptors. The aim of QSAR analysis is to discover such general rules and equations. Activities of interest include chemical reactivity, biological activity and toxicity. In this paper, we are concerned with predicting biological activity for drug design. Typically we measure or calculate the descriptors of a finite number of compounds and measure the activity of interest. A priori knowledge about the underlying chemistry may also be considered. The aim is that the rules discovered should be successful in predicting the properties of previously unseen compounds. Solving QSAR problems where data sets contain few rows (compounds) and many columns (descriptors) is very difficult with standard statistical approaches. Large data sets of high dimensionality, which describe highly non-linear relationships between structure and activity, pose similar problems.

QSAR analysis is becoming increasingly important in automated pharmaceutical production processes. New compounds emerging from the production lines must be screened for their potential use (measured by chemical or biological activity in some assay) in future products. The capacity of the production lines is increasing through developments in robot technology and pharmaceutical methods. QSAR analysis forms an essential part of the overall screening process, in which new compounds are tested against structural models to determine their potential activity or otherwise. As demand for new pharmaceutical products increases, along with competition within the industry, companies require increased screening throughput and accuracy. Mistakes made at the screening stage are reflected in process inefficiencies and subsequent capital losses to the company involved.

Artificial intelligence techniques have been applied to QSAR analysis since the late 1980s, mainly in response to increased accuracy demands. Intelligent classification techniques, including neural networks (Devillers, 1999b), genetic algorithms (Devillers, 1999a) and decision trees, have come to the fore. The problem of combining high classification accuracy with informative results has also been tackled via the use of hybrid and prior knowledge techniques, such as expert systems (Gini et al., 1998). Machine learning techniques have, in general, offered greater accuracy than have their statistical forebears, but not without accompanying problems for the QSAR analyst to consider. Neural networks, for example, offer high accuracy in most cases but can suffer from over-fitting the training data (Manallack and Livingstone, 1999). Other problems with the use of neural networks concern the reproducibility of results, owing largely to set-up and stopping criteria, and lack of information regarding the classification produced (Manallack and Livingstone, 1999). Genetic algorithms may also suffer from their stochastic nature, in that results may be hard to reproduce and the resulting classification may not be optimal (Goldberg, 1989). Decision trees offer a large amount of information regarding their decisions, in the form of predictive rules, but occasionally struggle to provide the accuracy supplied by more powerful, but less informative, techniques. Owing to the reasons outlined above, there is a continuing need for the application of more accurate and informative classification techniques to QSAR analysis.

3 Support Vector Machines

The general problem of machine learning is to search a, usually very large, space of potential hypotheses to determine the one that will best fit the data and any prior knowledge (Mitchell, 1997). The data may be observed or unobserved or a combination of the two. In the case of classification problems we are presented with examples generated from some real world phenomenon. Each example belongs to one of a fixed number of classes, possibly including the null class. We assume that the examples are independently and identically distributed (i.i.d.). The task is to learn an hypothesis based on this data and any prior knowledge that correctly predicts the labels of previously unseen data generated i.i.d. from this phenomenon. That is we aim to minimize the generalization error. Normally we randomly partition the data into a training set and a test set. The hypothesis is learned using the training data. An unbiased estimate of the generalization error is then given by the error on the test set. To reduce the variance in the estimate this can be repeated several times and the results averaged over the different partitions (cross-validation).

Classifiers typically learn by empirical risk minimization (ERM) (Vapnik, 1998), that is they search for the hypothesis with the lowest error on the training set. Unfortunately, this approach is doomed to failure without some sort of capacity control. To see this, consider a very expressive hypothesis space. If the data are noisy, which is true of most real world applications, then the ERM learner will choose an hypothesis that accurately models the data and the noise. Such a hypothesis will perform badly on unseen data. To overcome this we limit the expressiveness of the hypothesis space. This is achieved by adding a regularization term that penalizes complex models (Ripley, 1996). The other main problem with conventional techniques is that there are usually numerous parameters. These are generally set using a separate validation set of data or by ad hoc heuristics. Thirdly, many algorithms converge only to locally optimal hypotheses within their search space, as opposed to the global optimum.

The support vector machine (SVM) is a relatively recent addition to the toolbox of the data-mining practitioner. Support vector machines are based on the structural risk minimization principle (SRM) (Vapnik, 1979) from computational learning theory. SVMs construct a hyperplane that separates two classes (this can be extended to multi-class problems). Separating the classes with a large margin minimizes a bound on the expected generalization error. By searching for large margin hyperplanes the SVM is limiting the complexity of the hypothesis space, measured in terms of the VC-dimension (Vapnik and Chervonenkis, 1974). This is also intuitively reasonable since, given two separable classes, we would like to construct an hyperplane that is in the centre of the separating band. In the case of non-separable classes we simply minimize the number of misclassifications whilst maximizing the margin with respect to the correctly classified examples (this introduces the only free parameter, C, of the SVM, see below). The hyperplane output by the SVM is given as an expansion on a small number of training points known as support vectors (SV). The SVs are closest to the hyperplane and intuitively correspond to those points that are hardest to classify. To train an SVM requires solving a large-scale quadratic programming (QP) problem. The solution to the QP problem is the global optimum and can be found quickly using techniques from mathematical programming (Burges, 1998).

SVMs are very powerful learners. Since the training data only appear in scalar products we can use a Mercer kernel (Mercer, 1909) to learn an hyperplane in a very high (even uncountable) dimensional space. SVMs can thus be used to learn two-layer, sigmoid neural networks, radial basis function (RBF) networks (see Bishop, 1995, for a description of these techniques) and polynomial classifiers among many others. In the separable case SVMs are fully automatic in that they need no fine-tuning. The VC-dimension can be used to select the optimal parameter settings without expensive cross-validation. Moreover, SVMs are relatively insensitive to variation in the parameters and are not prone to overfitting when, for example, using high degree polynomial kernels.

4 Why Should SVMs Work for QSAR Analysis?

When learning QSARs the algorithm must deal with high dimensional feature spaces. The SVM uses VC-theory to avoid over-fitting and hence has the potential to deal with a large number of features, even in the low throughput case where there are few training examples. This is apparent from the fact that SVMs typically learn in very high dimensional spaces without over-fitting. As mentioned in section 2, QSARs are typically highly non-linear. Learning a high degree polynomial classifier by regression is time-consuming and ad hoc, since we must decide on which cross-products to include and tune the degree of the polynomial. Using an SVM, we can learn a high degree polynomial classifier and be confident that it will not over-fit. Furthermore, the degree of the polynomial can be chosen using the VC dimension. A third point is that SVMs do not make any assumptions about correlations between the features, as opposed to techniques that assume statistical independence. Since SVMs are robust learners they are not as badly affected by noise as other classification algorithms. This is particularly important in high throughput screening where there may be a significant amount of noise in the data labels.

5 Problem Description

The data used in this experiment were obtained from the UCI Data Repository (Blake and Merz, 1998) and are described in (King et al., 1992). The problem is to learn a binary relationship on the biological activity of trimethoprim analogues. The biological activity is measured as $\log(1/K_i)$, where K_i is the equilibrium constant for the association of the drug to dihydrofolate reductase. Each drug has three positions of possible substitution. For each substitution position there are nine descriptors: polarity, size, flexibility, hydrogen-bond donor, hydrogen-bond acceptor, π donor, π acceptor, polarizability and σ effect. Each of the twenty-four non-hydrogen substituents was given an integer value for each of these properties (see King et al., 1992, for details); lack of a substitution is indicated by nine -1s. This gives twenty-seven integer attributes for each drug.

The task is to learn the relationship $\text{great}(d_n, d_m)$ which states that drug no. n has a higher activity than drug no. m. Each instance consists of two drugs, giving fifty-four attributes in total, and a label 'true' or 'false', indicating the value of the relationship great(). Each instance in the data is followed by its inverse, for example if the first instance represents $\text{great}(d_2, d_1) = \text{true}$ then the second instance represents $\text{great}(d_1, d_2) = \text{false}$. This produces a two-class classification problem where each class is equally represented (hence the misclassification costs are equal). The data are partitioned into a five-fold cross-validation series, with each fold consisting of approximately 1800 training examples and 1000 test examples. In learning the relationship great() we can rank the compounds in terms of their activity without going to the effort of predicting that activity. This is in contrast to the more general and harder problem of learning a set of rules or a regression equation to predict activity. The obvious limitation of this set up is that $\text{great}(d_n, d_m)$ is meaningless when $n = m$, or when the two drugs have the same activity.

6 Results

A variety of standard classification algorithms were trained and tested on each cross-validation fold. Using the software package Clementine 5.0 (http://www.spss.com) a C5.0 decision tree, Gaussian RBF network and two- and three-layer sigmoid neural networks were trained. A 1-nearest neighbour (1-NN) classifier was also implemented using Matlab 5.3 (http://www.mathworks.com). An SVM was used to learn a Gaussian RBF classifier, the SVM was implemented with the software package SVM^{light} (Joachims, 1999, http://www-ai.informatik.uni-dortmund.de/thorsten/svm_light.html). The parameters for the various algorithms were chosen as follows. For C5.0, Clementine defaults were used (extensive validation showed that these were very close to optimal settings for this problem). For the neural network and RBF network, Clementine defaults were again used and capacity control was manually tuned by adjusting the number of hidden nodes. The number and location of the centres of the RBFs was found automatically by Clementine using k-means clustering. For the nearest neighbour classifier Euclidean distance was used. For the SVM the width of the Gaussian RBFs was chosen as that which minimized an estimate of the VC-dimension. The parameter C in the SVM, which controls the error-margin tradeoff, was set at 100 (this is set by observing the number of SVs in the solution).

The test accuracies, averaged over the five cross-validation folds, are shown in table 1. Using 50% of the available training data to track generalization error, an automatically pruned neural network achieved 85.56% and a dynamically grown three-layer network achieved 85.27%. Thus, the SVM classification is significantly better ($p < 0.01$) than all of the existing techniques bar the capacity controlled neural network. C5.0 is quickest to train, followed by the SVM. The neural networks used require at least an order of magnitude greater training time. The nearest neighbour classifier requires no training but classification time is two orders of magnitude greater than for the other models.

Classifier	Test Acc %	Time/s
SVM-RBF	87.33	77.4
1-NN	83.62	382
NN (manual)	86.97	2110
RBF	78.76	418
C5.0	81.30	4.40

Table 1: Test accuracies and computational times; time includes training and test time but not I/O.

7 Conclusion

The support vector machine has been introduced as a robust and highly accurate intelligent classification technique, well suited to QSAR analysis. On a real QSAR analysis problem the SVM outperforms several of the most frequently used machine learning techniques. The neural networks achieve similar performance to that of the SVM but require an order of magnitude longer training times and manual capacity control. This becomes increasingly significant when learning QSARs on large numbers of compounds. Other techniques, including an RBF network, a C5.0 decision tree and a nearest neighbour classifier, all fall considerably short of the SVM's performance. The SVM is an automated and efficient deterministic learning algorithm, providing the benefit of reproducible and verifiable results.

The evidence presented in this paper suggests that the SVM is a data-mining tool with great potential for QSAR analysis.

Acknowledgements

This research has been undertaken within the INTErSECT Faraday Partnership managed by Sira Ltd and the National Physical Laboratory, and has been supported by the Engineering and Physical Sciences Research Council (EPSRC), SmithKline Beecham, Glaxo Wellcome, SPSS and Sira. The data mining software package Clementine has been provided by SPSS.

Robert Burbidge is an associate of the Postgraduate Training Partnership established between Sira Ltd and University College London. Postgraduate Training Partnerships are a joint initiative of the Department of Trade and Industry and EPSRC.

References

C. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, 1995.

C. L. Blake and C. J. Merz. UCI repository of machine learning databases. 1998. URL http://www.ics.uci.edu/.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):1–47, 1998.

J. Devillers. *Genetic Algorithms in Molecular Modeling*. Academic Press, 1999a.

J. Devillers. *Neural Networks and Drug Design*. Academic Press, 1999b.

Gini, Testaguzza, Benefenati, and Todeschini. Hybrid toxicology expert system: architecture and implementation of a multi-domain hybrid expert system for toxicology. *Chemometrics and Intelligent Laboratory Systems*, 43:135–145, 1998.

D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

T. Joa chims. Making large-scale svm learning practical. In C. Burges B. Scholkopf and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT Press, 1999.

R. D. King, S. Muggleton, R. A. Lewis, and M. J. E. Sternberg. Drug design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogues binding to dihydrofolate reductase. *Proc. Natl. Acad. Sci. USA*, 89:11322–11326, 1992.

D. T. Manallack and D. J. Livingstone. Neural networks in drug discovery: have they lived up to their promise? *Eur. J. Med. Chem.*, 34:95–208, 1999.

J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions Royal Society London A*, 209:415–446, 1909.

T. Mitchell. *Machine Learning*. McGraw-Hill International, 1997.

B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, 1996.

V. Vapnik. *Estimation of Dependencies Based on Empirical Data*. Nauka, Moscow, 1979.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

DEFINING MEDICAL PROTEIN TARGETS: CHALLENGES FOR GENETIC ALGORITHMS, EXPERT KNOWLEDGE AND ARTIFICIAL INTELLIGENCE

Thomas Dandekar; Fuli Du; Steffen Schmidt
EMBL, Heidelberg and university of Heidelberg,
Postfach 102209, D-69012 Heidelberg, Germany;
dandekar@embl-heidelberg.de

Abstract

Exploiting the rapid increase in available sequence data, definition of medical relevant protein targets is improved by a combination of (i) differential genome analysis (target list) and (ii) structure prediction (individual target picture). Fast sequence comparisons, data mining and genetic algorithms further enhance these procedures. *Mycobacterium tuberculosis* polyphosphate glucokinase is chosen as an application example.

1 Introduction

Here we present some of the challenges and problems for bioinformatics in the steps towards the identification of a potential medical protein target from an available more or less complete genome sequence. The steps will be illustrated referring to examples and techniques from our laboratory. We will first review and describe steps in identifying a potential interesting target from a genome sequence. Next a particular target found (example: polyphosphate glucokinase from *Mycobacterium tuberculosis*) is examined in its structure applying the genetic algorithm. Techniques from artificial intelligence (AI) become also more and more important in protein target identification. On the other hand, several steps are not yet easy automised or decided and recognised by AI systems and computers without any human intervention.

2. Materials and Methods

2.1. Differential genome analysis

Differential genome analysis was carried out extensively using sequence comparison methods as described previously (Huynen et al., 1998). Additional programs for protein and domain identification included PSI-BLAST (Altschul et al., 1997) and SMART (Schulz et al., 1998), and specific PERL and awk programs and routines.

2.2. Secondary structure prediction

Secondary structure prediction was achieved using available secondary structure prediction programs. Profile based neural networks are used by Rost's program PHD (Rost, 1996). In contrast, Frishman and Argos' program PREDATOR (Frishman and Argos, 1997) utilises local pairwise alignment of the sequence to be predicted with each related sequence. Levin's program SIMPA96 assigns secondary structure comparing the blosum62 similarity scores of best matching fragments in a database of known structures (Levin, 1997).

2.3. Genetic algorithm protein folding simulations.

The protein main chain (N,C_α,C' and O) was modelled in grid free simulations using internal co-ordinates and a set of seven standard conformations to assign ϕ and ψ values to the backbone (Rooman et al., 1991). The conformations of all residues along the amino acid sequence were successively collected together and decoded from a long bit-string (a "chromosome"). Starting from a population of random bit-strings, the quality of each encoded structure was judged by a fitness function composed of rewards and punishments. Five structure parameters, suitably weighted (further details in Dandekar and Argos, 1994, 1996) were calculated and summed up to judge structural fitness, briefly
1) the total scatter of all residues around the common centre of mass is calculated;
2) the distribution of hydrophobic residues around the centre of mass (same centre as in 1) is considered;
3) main chain van-der-Waals atomic overlaps are punished;
4) secondary structures are promoted co-operatively with different functions depending on whether they occur in a region predicted by secondary structure prediction programs or not;
5) the formation of hydrogen bonds in ß-strands, formation of ß-sheets and reverse turns is measured by different subroutines;

2.4. Genetic algorithm simulation conditions

High quality bit-strings (after a random start) were selected preferentially as parents and mutated (one bit per string per generation) and recombined through cross-over (probability of recombination is 0.2 per bit string per generation and occurs at exactly one equivalent site chosen at random on each of the parental chromosome pairs) to yield the next parental generation of folds. A positive constant keeps the population of prediction trials richer since low fitness individuals may also survive. Simulations were run over many generations to allow convergence (the product of population and generation equalled at least 4 x 10^5, corresponding to a processing time for main chain simulation runs of 20 minutes on a VAX 7620 for a 46-residue protein). The best fold comparing the fittest individuals from ten runs yielded the prediction in these simulations.

3. Results and Discussion

3.1. Differential genome analysis

Large scale sequencing projects yield a huge amount of sequencing data. Even the complete genome sequence from a number of prokaryotic organisms has been obtained and after yeast also the first genomic overviews from eukaryotes such as *Caenorhabiditis elegans* and *Drosophila* become available. In the following we will concentrate on a simpler, but severely pathogenic organism, *Mycobacterium tuberculosis*.
Detection of open reading frames is even in prokaryotes a non-trivial exercise involving detection software such as GENEMARK (Lukashin and Borodovsky, 1998). Such programs are constantly improved, for instance by applying hidden markov chain models but they doubtless require further improvement for the challenging task of gene detection, in particular in eukaryotes, including more AI methods.

However, for the purpose described here, protein target identification, we will focus on the next step, differential genome analysis, as this method allows us to tackle the following recognition task: Which proteins are specific for this organism, which are shared with several other species and which proteins are general and wide-spread ? Sorting the genome encoded proteins in this way allows in a first order approximation to define pharmacological targets: Genome specific enzymes may be pharmacological targeted without hurting other organisms, in particular the human patient if the genome analysed is from a pathogen. Furthermore, genes shared only among pathogens give a further clue to identify new pathogenicity factors.

These questions can be answered by a two step procedure, first the group of genes one is interested in (or even a whole genome) is compared by fast, automated sequence comparison procedures with the corresponding genes from other genomes.

Three different types of genes can be distinguished after being compared to a given query sequence: Genes from other species can either be (i) strongly similar and probably encode a protein with the same function (orthologue), (ii) be related but perhaps only in certain domains or parts of the complete sequence (a "homologue") or (iii) are not significantly related. Relatedness is established by significant e-values ($p > 0.001$) in sequence comparisons, orthology by the stricter criterion of highest reciprocal relatedness among all proteins from a whole genome. For this typical algorithms such as BLAST, FASTA and PSI-BLAST are available (Bork et al., 1998). In particular, with application specific software for large scale sequence to sequence comparisons the speed up obtained is large enough to compare even complete genomes amongst each other (Huynen et al., 1998).

3.2. Lists of potential targets

The genes encoding orthologous protein with the same function we next classify according to Venn diagrams. Different categories are identified in this way such as organism specific genes (e.g. our example kinase), genes shared between several pathogenic species (e.g. host interaction factors), between most bacteria (e.g. ribosomal proteins) or between patient and parasite (e.g. triosephosphate isomerase). This Venn classification can be automated using awk scripts and perl programs.

Currently we are investigating more advanced ways to cluster and sort genes. This includes different data mining subroutines which in addition to direct sequence similarity (orthologous genes, detected by the programs mentioned above) interpret and start to understand similar functionality. This can be achieved by combining clusters according to sequence similarity with perl programs linking and sorting genes by functional patterns identified in the description line comparing genes from different species. Thus, referring to the specific example explained here, the polyphosphate glucokinase from *Mycobacterium tuberculosis* (E.C. 2.7.1.63) can be brought and grouped into the context of other, not functional identical but related enzyme activities for instance hexokinases and glucokinases from different organisms.

3.3. further identification tools

The challenging recognition problem of species specific versus wide-spread, common genes can thus at least in a first level approximation be automated using a two step procedure and by excessive use (computer time demanding cross-comparisons) of fast comparison routines on each step. On the other hand several additional steps necessary to derive suitable targets we have not yet codified. For instance, expert knowledge is required to understand and derive detailed medical implications from the different lists and genes compiled in the above mentioned way so that attention can be focused on the biological most interesting or medical most promising targets.

However, expert systems may after sufficient training and development also perform such further sorting steps. As one step towards this goal we are currently concerned with deriving more general rules how related sequences may be recognised and can be grouped. Simple rules such as the percentage of sequence identity multiplied by the length of the identity stretch are combined with enzyme specific data strings (such as "kinase") and recognition motifs (e.g. from the database PROSITE) to allow a far more specific recognition and classification of related enzyme activities and with the potential for large scale up.

3.4. target structure analysis

We will now for the second part of our results focus on the structure analysis of the polyphosphate glucokinase from *Mycobacterium tuberculosis*. The gene and its encoded enzyme are a specific adaptation of *Mycobacterium tuberculosis* (see set 1 above, species specific adaptations). It catalyses the reaction: phosphate(n) + D-glucose <-> phosphate(n-1) + D-glucose phosphate.

As polyphosphate glucokinase connects carbohydrate metabolism and energy utilisation it may be considered part of a supportive or backup pathway for *Mycobacterium tuberculosis* not present in human beings and presents in this way a potential interesting pharmacological target. As a first step for drug design a picture of its three dimensional structure is helpful, e.g. to define hydrophobic pockets or size and shape of its catalytic cavity.

Recognition of the three dimensional structure of a protein is again a challenge for AI. Many different methods for the prediction of three dimensional protein folds have been devised (reviewed e.g. in Dandekar and König, 1997), including different methods from AI research such as the use of neural networks (e.g. Jones, 1999; Norton et al., 1998; Mandal et al., 1996) and genetic algorithms (reviewed e.g. in Clark and Westhead, 1996). We will present here own results applying the latter technique. These search strategies are robust global optimizers and have been applied for numerous optimisation tasks in engineering and computing as well as to build classifier systems and to perform other tasks in AI research (Goldberg, 1989). Since several years (Dandekar and Argos, 1992; König and Dandekar, 1999) we are applying genetic algorithms for protein fold prediction from sequence and secondary structure.

Regarding the polyphosphate glucokinase from *Mycobacterium tuberculosis*, a first check concerned the question whether the fold can be recognised by simpler techniques. In particular it was tested whether it can be linked via sequence homologies to a known three dimensional protein structure, for instance applying the program PSI-BLAST (Altschul et al., 1997). In the example there is not yet a homologous structure available, similar a program specifically geared to recognise and analyse protein domains ("SMART", Schulz et al., 1998) fails to recognise a known domain. In such cases, use of the genetic algorithm to derive a fold prediction is advantageous. Starting from sequence and a combined secondary structure prediction (using the programs PHD, SIMPA96 and PREDATOR, see Materials and Methods) the genetic algorithm calculates from this start information a prediction of the protein fold for the polyphosphate glucokinase from *M.tuberculosis*.

Different protein structure selection criteria (see Materials and Methods) are implemented to select structures by mutation and recombination of solution trials during several hundred generations structures which close to optimal fulfil the combined structure selection criteria. To achieve fold predictions close to the native structure of the protein appropriate criteria and suitable respective weights for the fitness function had to be carefully selected. Furthermore, an extensive test battery of structures with known three dimensional fold was used in numerous test runs to judge the outcome of the selection and how it operated on different sequences and protein topologies (see Materials and Methods; further details in Dandekar and Argos, 1994; Dandekar and Argos, 1996).

The globular fold obtained by the genetic algorithm simulation is a prediction of the complete fold for that protein. However, in our experience, as this protein structure is larger than 100 amino acids (in total 265 amino acids) and contains many different secondary structure elements this presents only a rough first order approximation of the fold (RMSD error in parts of the fold still higher than 5 Å). Nevertheless, the respective topology of important residues (e.g. the catalytic centre) is available from the model structure and sufficiently detailed to plan molecular biology and structure probing experiments (König and Dandekar, 1999). In particular, after several cycles of model building and experiment such a model becomes accurate enough (Saxena et al., 1997) that pharmacological intervention strategies can be examined.

In the concrete example this could be inhibitors of *M.tuberculosis* polyphosphate glucokinase targeting its catalytic centre, however, with low affinity to the standard glucokinases and hexokinases present in man, especially in liver (main organ where glucokinase is present), erythrocytes and brain (glycolysis critical for energy supply in these organs).

4. Conclusion

We present here our two part strategy to identify and analyse potential pharmacological targets from genome sequences using *Mycobacterium tuberculosis* as an example. Both parts involve different algorithms and programs from biocomputing and exploit advantageously concepts from AI, notably concerning complex recognition tasks.

In the first task, the identification of a new target, we achieve this by a combination of sequence to sequence comparisons, data mining and motif recognition, for the second task, three dimensional fold recognition, we exploit genetic algorithm simulations; polyphosphate glucokinase is yielded as a potential medical target.

Nevertheless, for both tasks, additional improvements will be made possible from advances in AI research, in particular to further reduce the requirement for human intervention (notably in target recognition) and experimental data (notably in structure prediction).

Acknowledgements

Support of research in our laboratory by DFG and BMBF is gratefully acknowledged.

References

S.F. Altschul, T.L. Madden, A.A. Schaffer, J. Zhang, Z. Zhang, W. Miller and D.J. Lipman. Gapped Blast and PSI-Blast, a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389-3402, 1997

P. Bork, T. Dandekar, Y. Diaz-Lazcoz, F. Eisenhaber, M. Huynen and Y. Yuan. Predicting function: From genes to genomes and back. *J.Mol.Biol.* 283:707-725, 1998

D.E. Clark and D.R. Westhead. Evolutionary algorithms in computer-aided molecular design.*J. Comp.-Aided Mol. Design* 10:337-358, 1996

T. Dandekar and P. Argos. Potential of genetic algorithms in protein folding and protein engineering simulations. *Protein engineering* 5:637-645, 1992

T. Dandekar and P. Argos. Folding the main chain of small proteins with the genetic algorithm. *J.Mol.Biol.* 236: 844-861, 1994

T. Dandekar and P. Argos. Identifying the tertiary fold of small proteins with different topologies from sequence and secondary structure using the genetic algorithm and extended criteria specific for strand regions. *J.Mol.Biol.* 256:645-660, 1996

T. Dandekar and R. König. Computational methods for the prediction of protein folds. *Biochimica et Biophysica Acta* 1343:1-15, 1997

D. Frishman and P. Argos. Seventy-five percent accuracy in protein secondary structure prediction. *Proteins* 27:329-335, 1997

D. Goldberg *Genetic algorithms in search, optimization and machine learning*. Addison Wesley, Massachusetts, 1989

M.A. Huynen, T. Dandekar and P. Bork. Genomics: Differential genome analysis applied to the species specific features of *Helicobacter pylori. FEBS Lett.* 426:1-5, 1998

D.T. Jones. GenTHREADER: An efficient and reliable fold recognition method for genomic sequences. *J.Mol.Biol.* 287:797-815, 1999

R. Koenig and T. Dandekar. Refined genetic algorithm simulations to model proteins. *J. Mol. Model.* 5:317-324, 1999

J.M. Levin.Exploring the limits of nearest neighbour secondary structure prediction. *Protein Eng.* 10:771-776, 1997

A.V. Lukashin and M. Borodovsky. GeneMark.hmm: new solutions for gene finding. *Nucleic Acid Res.* 26: 1107-1115, 1998

C. Mandal, B.D. Kingery, J.M. Anchin, S. Sybramaniam and D.S. Linthicum. ARBGEN: A knowledge-based automated approach for antibody structure modeling. *Nat. Biotechnol* 14:323-328, 1996

D.D. Norton, D.S. Dwyer,D.S. and M.I. Muggeridge. Use of a neural network secondary structure prediction to define targets for mutagenesis of herpes simplex virus glycoprotein B. *Virus Res.* 55:37-48, 1998.

M.J. Rooman, J.-P. Kocher and S.J. Wodak. Prediction of protein backbone conformation based on seven structure assignments *J. Mol. Biol.* 221:961-979, 1991.

B. Rost. PHD: predicting one-dimensional protein structure by profile-based neural networks. *Methods Enzymol.* 266:525-539, 1996

P. Saxena, I. Whang, Y. Voziyanov, C. Harkey, P. Argos, M. Jayaram and T. Dandekar. Probing Flp: A new approach to analyze the structure of a DNA recognizing protein by combining the genetic algorithm, mutagenesis and non-canonical DNA target sites. Biochim. Biophys. Acta 1340, 187-204, 1997

J. Schultz, F. Milpetz, P. Bork and C.P. Ponting. SMART, a simple modular architecture research tool: Identification of signalling domains. *Proc. Natl. Acad. Sci. USA*, 95:5857-5864, 1998

Topology–based protein structure comparison using a pattern discovery technique

David Gilbert[1,2] ; David Westhead[3] ; Juris Viksna[1,4] ; Janet Thornton[2,5,6]
[1] Department of Computer Science; City University, Northampton Square, London EC1V 0HB, UK
[2] European Institute of Bioinformatics; Hinxton, Cambridge CB10 1SD, UK
[3] School of Biochemistry and Molecular Biology; University of Leeds, Leeds LS2
[4] Institute of Mathematics and Computer Science; University of Latvia, Riga LV-1459, Latvia
[5] Department of Biochemistry; University College, London WC1E 6BT, UK
[6] Crystallography Dept.; Birkbeck College, London WC1E 7HX, UK
[1]drg@soi.city.ac.uk ; [3]westhead@bmb.leeds.ac.uk ; [4]jviksna@cclu.lv ; [5]thornton@biochem.ucl.ac.uk

Abstract

We describe the design and implementation of a fast topology–based method for protein structure comparison. The approach uses the TOPS topological representation of protein structure, aligning two structures using a common discovered pattern and generating measure of distance derived from an insert score. Heavy use is made of a constraint-based pattern matching algorithm for TOPS diagrams that we have designed. The system is maintained at the European Bioinformatics Institute and is available over the Web via the at tops.ebi.ac.uk/tops. Users submit a structure description in Protein Data Bank (PDB) format and can compare it with structures in the entire PDB or a representative subset of protein domains, receiving the results by email.

Keywords: structure comparison, constraints, pattern matching, pattern discovery, protein motifs, protein topology.

1 Introduction

An understanding of the similarities and differences between protein structures is very important for the study of the relationship between sequence, structure and function, and for the analysis of possible evolutionary relationships. This has lead to the need for computational methods of structure comparison; furthermore, the rapid increase in the size of structural databases means that techniques to compare a given structure with member of such a database should be fast.

Various structure comparison methods have emerged, ranging from those which make detailed geometrical comparisons of backbone coordinates Taylor and Orengo (1989), through methods using vector approximations to secondary structure elements, or SSEs, Mitchell et al. (1989); Grindley et al. (1993); Artymuik et al. (1994), and finishing with methods based on highly simplified models of structure Koch et al. (1996); Koch and Lengauer (1997); Tsukamoto et al. (1997). These latter methods typically consider a sequence of SSEs, along with relationships like spatial adjacency within the fold and approximate orientation, neglecting details like lengths and structures of loops, and the lengths of the secondary structure elements themselves. This type of description of a protein structure is commonly known as a 'topological' description.

The topological description has the advantage of simplicity, which makes it possible to implement very fast comparison algorithms. Further, by neglecting many of the details which typically vary between related structures, like lengths and structures of loops, and exact lengths, spatial positions and orientations of SSEs, it has the potential to detect more distant structural relationships than could be found by methods based on more geometrical descriptions. On the other hand, its disadvantages are that there may be structures which, although related at the topological level, are very different from a geometric point of view, and have no meaningful biological relationship.

2 TOPS diagrams and patterns

TOPS cartoons were originally drawn manually Sternberg and Thornton (1977) and comprise graphical representations of secondary structure elements (SSEs), their relative orientations and some indication of spatial adjacency. Subsequently a richer representation of the topological structure has been devised Flores et al. (1994); Westhead et al. (1999, 1998), termed a TOPS *diagram*, which includes information about hydrogen bonding between strands and chirality connections between SSEs; this representation is used to automatically produce graphical cartoons.

We have previously described in detail our formal representation of TOPS diagrams and patterns as graphs, and the design of a fast pattern matching program Gilbert et al. (1999). In this paper we describe a pattern discovery algorithm for TOPS diagrams and show how we use it to structurally align diagrams and compute a comparison measure.

TOPS diagrams In TOPS diagrams (for example the diagram for 2bop in Figure 1), strands are represented by triangles and helices by circles, connected in a sequence from the amino (N) terminus to the carboxy (C) terminus. SSEs are considered to have a direction of 'up' or 'down', implied in the way the connecting lines to the symbols are drawn: connections drawn to the edge of a symbol imply connection to the base and those drawn to the centre imply connection to the top, and the direction is that taken by the protein chain from N to C terminus. The direction information is duplicated for strands: upward pointing triangles have the direction 'up' and downward pointing ones the direction 'down'. The existence of hydrogen bond ladders between a pair of strands is indicated by a single H-bond in the TOPS representation, labelled as being parallel or anti-parallel, according to the relative directions of the two strands that it joins. In addition, TOPS diagrams also represent the chiralities of connections between connections between two parallel strands within the same sheet and connections between long parallel helices. A more detailed description of TOPS diagrams can be found in Gilbert et al. (1999).

More formally, a TOPS diagram is a triple (S, H, C) where $S = S_1, \ldots, S_k$ is a sequence of length k of secondary structure elements (SSEs) and H and C are relations over the SSEs, called respectively H-bonds and chiralities. In this description an H-bond constraint refers to a ladder of individual hydrogen bonds between adjacent strands in a sheet. We will later refer to the *length* of a diagram as the length of the sequence S.

In our formalism an SSE is a character from the alphabet $\{\alpha, \beta\}$ standing for helix and strand respectively. Since each SSE in a TOPS diagram is associated with a direction *up* or *down* we associate a direction symbol, $+$ or $-$, with each letter of our alphabet, giving $\{\alpha_+, \alpha_-, \beta_+, \beta_-\}$.

Both H-bonds and chiralities are symmetric relations (non-directed arcs in the graph). An H-bond constrains the types of the two SSE's involved to be strands, and each bond is associated with a relative direction $\delta \in \{P, A\}$, indicating whether the bond is between parallel or anti-parallel strands. Chiralities are associated with handedness $\chi \in \{L, R\}$ (left and right respectively), and only occur between pairs of SSEs of the same type. We denote the H-bond relationship between two SSEs S_i and S_j by (S_i, δ, S_j) and a chirality relationship by (S_i, χ, S_j).

The formal definition of a TOPS diagram $D = (S, H_d, C_d)$, given $\Sigma = \{\alpha_+, \alpha_-, \beta_+, \beta_-\}$, is
$S = (S_1, \ldots, S_k),\ S_i \in \Sigma$
$H_d = \{(S_i, \delta, S_j) | S_i, S_j \in \{\beta_+, \beta_-\},\ \delta = P \leftrightarrow S_i = S_j,\ \delta = A \leftrightarrow S_i \neq S_j\}$
$C_d = \{(S_i, \chi, S_j) | S_i, S_j \in \Sigma,\ \chi \in \{R, L, \}\}$

As an example, consider the TOPS diagram for 2bop in Figure 1; we can 'stretch out' this diagram to give a linear form, as shown in Figure 3, and represent it formally as 2bop = (S, H, C), where
$S = (\beta_{+_1}, \alpha_{-_2}, \alpha_{-_3}, \beta_{+_4}, \beta_{+_5}, \beta_{-_6}, \alpha_{+_7}, \beta_{-_8})$
$H = \{(\beta_{+_1}, A, \beta_{-_6}), (\beta_{+_1}, A, \beta_{-_8}),$
$(\beta_{+_4}, A, \beta_{-_6}), (\beta_{+_5}, A, \beta_{-_6})\}$
$C = \{(\beta_{+_1}, R, \beta_{+_4}), (\beta_{-_6}, R, \beta_{-_8})\}$

TOPS patterns A TOPS *pattern* (or *motif*) is similar to a TOPS diagram, but is a generalisation which describes several diagrams conforming to some common topological characteristics. This generalisation is achieved by specifying the insertion of SSEs (and any associated H-bond and chiralities) into the sequence of secondary structure elements; indeed a diagram is just a pattern where no inserts are permitted. The length of an insert is constrained to be within the range of the lengths of the sequences that can be inserted. A TOPS pattern is thus a triple, similar to that of a TOPS diagram; in this case, however, we refer to the sequence of SSEs with inserts permitted as *T-pattern*. The inserts are similar to wild cards with length constraints; we extend the definition of TOPS patterns given in Gilbert et al. (1999) to permit such wild cards before the beginning of, and after the end of the sequence of SSEs.

Formally a TOPS pattern is a triple (T, H, C) where T (referred to as a T-pattern) is a sequence $(n_0, m_0) - V_1 - (n_1, m_1) - V_2 - \ldots - (n_{k-1}, m_{k-1}) - V_k - (n_k, m_k)$ comprising secondary structure elements indicated by V_i and between each of these an insert description, as well as an insert description (n_0, m_0) before V_1 and also an insert (n_k, m_k) after V_k. Each insert description is a pair (n, m) where n stands for the minimum and m for the maximum number of SSEs which can be inserted at that position. The range of n and m is from zero to the largest number of SSE's in any TOPS diagram (approximately 60). H are H-bonds and C are chiralities, just as in the diagrams. Since TOPS diagrams exhibit rotational invariances of 180° about the x and y-axes, we associate a *direction variable*, $\oplus$ or $\ominus$ with each SSE in a pattern P s.t. they satisfy the constraint
$\forall \oplus, \ominus \in P : \mathit{opp}(\oplus, \ominus) \leftrightarrow (\oplus = + \wedge \ominus = -) \vee (\oplus = - \wedge \ominus = +)$

The formal definition of a TOPS diagram pattern $P = (T, H_p, C_p)$, $\forall \oplus, \ominus \in P : \mathit{opp}(\oplus, \ominus)$, given $\Sigma = \{\alpha_\oplus, \alpha_\ominus, \beta_\oplus, \beta_\ominus\}$ is:
$T = (n_0, m_0) - V_1 - (n_1, m_1) - V_2 - \ldots - (n_{k-1}, m_{k-1}) - V_k - (n_k, m_k)$, $V_j \in \Sigma, n_j \leq m_j$
$H_p = \{(S_i, \delta, S_j) | S_i, S_j \in \{\beta_\oplus, \beta_\ominus\},\ \delta = P \leftrightarrow S_i = S_j,\ \delta = A \leftrightarrow S_i \neq S_j\}$
$C_p = \{(S_i, \chi, S_j) | \chi \in \{R, L, \},\ S_i, S_j \in \Sigma\}$

For example a TOPS pattern which describes plaits, of which 2bop is an instance, is given by Plait = (V, H, C), where
$V = ((0, \mathbf{N}) - \beta_{\oplus_1} - (0, \mathbf{N}) - \alpha_{\ominus_2} - (0, \mathbf{N}) - \beta_{\oplus_3} - (0, \mathbf{N}) - \beta_{\ominus_4} - (0, \mathbf{N}) - \alpha_{\oplus_5} - (0, \mathbf{N}) - \beta_{\ominus_6} - (0, \mathbf{N}))$
$H = \{(\beta_{\oplus_1}, A, \beta_{\ominus_4}), (\beta_{\oplus_1}, A, \beta_{\ominus_6}), (\beta_{\oplus_3}, A, \beta_{\ominus_4})\}$
$C = \{(\beta_{\oplus_1}, R, \beta_{\oplus_3}), (\beta_{\ominus_4}, R, \beta_{\ominus_6})\})$

Figures 2 and 4 illustrate this in non-linear and linear form respectively.

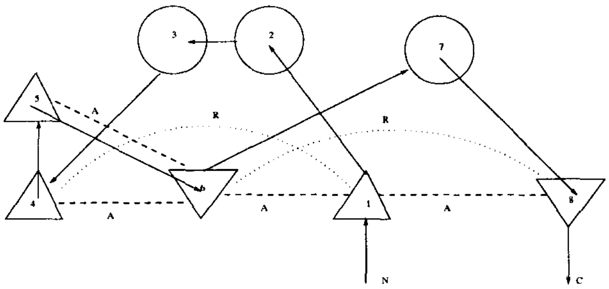


Figure 1: TOPS diagram for 2bop

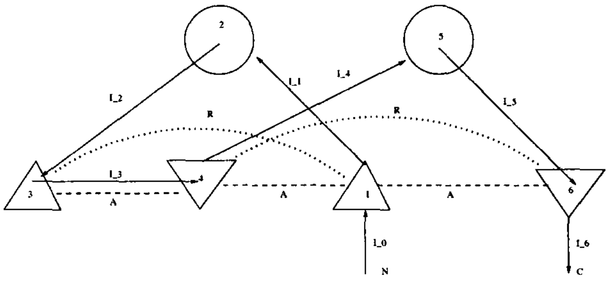


Figure 2: TOPS diagram for the plait motif

3 Methods

We have designed a measure to compare the similarity between two TOPS diagrams, in order to be able to perform structure comparison at the topological level. Our method works by performing a structural alignment of the SSEs of the diagrams and computing a score based on an edit distance over aligned blocks of SSEs plus contributions from the H-bond and chirality sets of the diagrams. In order to perform the alignment we use a least general common pattern generated by a pattern discovery technique which we have designed; this in turn makes heavy use of our constraint-based pattern matching method for TOPS diagrams.

3.1 Pattern discovery for TOPS diagrams

Pattern discovery for sequences is a well-established technique Brazma et al. (1998) which could be applied to TOPS diagrams and patterns as follows. The first, "pattern driven" (PD) is based on enumerating candidate patterns in a given solution space and picking out the ones with high fitness; the second, "diagram driven" (DD) comprises algorithms that try to find patterns by comparing given diagrams and looking for local similarities between them. In the equivalent of DD for sequences, an algorithm may be based on constructing a local multiple alignment

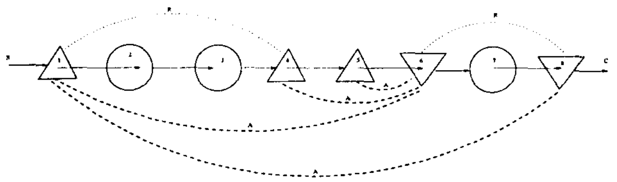


Figure 3: Linearised TOPS diagram for 2bop

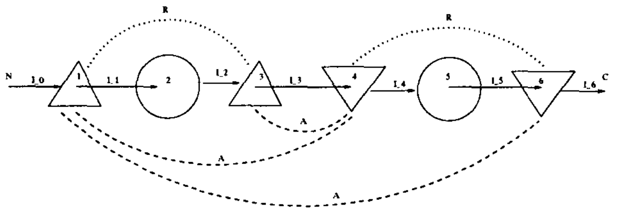


Figure 4: Linearised TOPS diagram for the plait motif

of given sequences and then extracting the patterns from the alignment by combining the segments common to most of the sequences.

Essentially the difference between pattern discovery for sequences and TOPS diagrams is that techniques for the former assume that the grammar of the former is regular whilst that of the latter is context–sensitive due to the fact that H-bond and chirality arcs may cross (i.e. they describe a "copy language"). Thus in a naive version of a PD approach for TOPS diagrams not only would we have to enumerate an exponentially large number of patterns comprising not only all the possible combinations of the SSEs (and their orientations) in a pattern of length k, but also all the possible H-bond and chirality connections over them.

Our algorithm discovers patterns of H-bonds (and chiralities) based on the properties of sheets for TOPS diagrams; we also derive T-patterns, i.e. the associated sequences of SSEs and insert sizes. Briefly, the algorithm attempts to discover a new sheet by finding, common to all the target set of diagrams, a (fresh) pair of strands, sharing an H-bond with a particular direction. Then it attempts to extend the sheet by repeatedly inserting a fresh strand which is H-bonded to one of the existing strands in the (current) sheet. The algorithm then finds all further H-bonds between all the members of the current sheet. The entire process is repeated until no more sheets can be discovered; any chirality arcs between the H-bonds in the pattern are then discovered by a similar process. The numbers of inserts between each strand in the pattern are then computed for all the patterns in the learning set, and the minimum and maximum size of the gaps in the corresponding insert positions in the pattern are thus found, and combined with the SSE sequence to give the T-pattern. The result is the least general common TOPS pattern characterising the target set of protein descriptions.

Naive insertion of a new SSE into an existing sequence of SSEs is expensive: consider the case when the existing sequence is of length 2. The new H-bond can be inserted at the beginning of the sequence, at the end of the sequence or between the existing two SSEs. Moreover, a new H-bond must be discovered between the new SSE and one of the existing SSEs in the sequence. We use a 'seed' derived from one of the target set of diagrams in order to give the insertion point: the H-bond pattern is extended in one diagram first by selecting one of the remaining bonds from the diagram H-bond set; if this fails to give a pattern which matches the other diagram, then an alternative bond is selected.

An alternative approach would be to adapt that of Koch et al Koch et al. (1996), which constructs an edge product graph for two graphs and then employs Bron and Kerbosch's algorithm Bron and Kerbosch (1973) which enumerates all the maximal cliques in the graph. Although Koch et al improve Bron and Kerbosch's algorithm by restricting the search process to cliques representing connected substructures, they determine common substructures in more than two topology graphs by forming the intersections between all substructures of all cliques resulting from a pairwise comparison.

The worst-time complexity for the learning algorithm based on repeated matching is approximately $O(k * n^n)$, where k is the number of sequences, and n the number of secondary structures (helices and strands) in a sequence. The maximal clique method has complexity $O((n^k/c_k)!)$ (with little information about c_k, except $c_k \geq 1$) for the same n and k. These are approximations assuming that number of nodes is approximately the same as the number of edges — this is more or less true in TOPS. In terms of implementation, the clique algorithm (for $k = 2$) tends to be slower (up to 10 times) in comparison with the repeated matching algorithm, although it sometimes produces better results.

We use a variant of the repeated matching algorithm to discover common patterns in all-α domains, where patterns of chirality arcs are discovered (stage ??), and stage ?? is omitted.

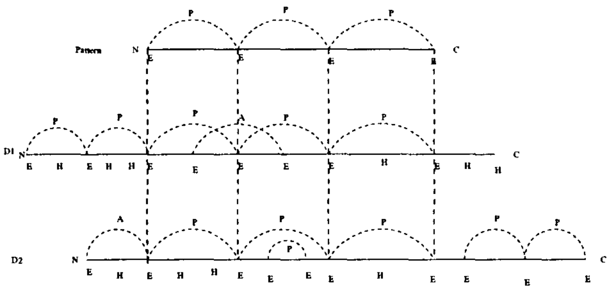


Figure 5: Making an alignment

Distance measure

Given two TOPS diagrams $D1 = (S1, H1, C1)$, $D2 = (S2, H2, C2)$ and a least general common pattern $P = (SP, HP, CP)$, we can make a structural alignment of S1 and S2 by matching P with $D1$ and $D2$. If $length(SP) = N$, then there are $N + 1$ insert positions in the pattern, corresponding to $N + 1$ blocks of unaligned SSEs in $D1$ and $S2$. An example is illustrated in Figure 5, where aligned blocks in $S1$ and $S2$ are indicated by $S1_1 \ldots S1_5$ and $S2_1 \ldots S2_5$ respectively.

The distance measure M between $D1$ and $D2$ is given by the normalised sum of the edit distances (Levenshtein (1965)) of all the blocks plus a contribution from the extra (when compared with the pattern) H-bonds and chiralities in the diagrams.

We have evaluated our method by performing a pairwise comparison of 1396 domains from the SCOP PDB40d database Murzin et al. (1995) and computed the error versus coverage data using the SCOP numbers as an indication of structural homology. Two domains are defined as homologous if at least their first three SCOP numbers are identical; the domains are non-homologous if only their first SCOP numbers are identical. Matches between domains with with only the first two SCOP numbers identical are ignored (not performed) since the SCOP hierarchy does not differentiate homologous and non-homologous pairs at this level. Coverage versus error results are given in Figure 6.

Times per comparison pair are typically 30–400ms on average (DEC Alpha).

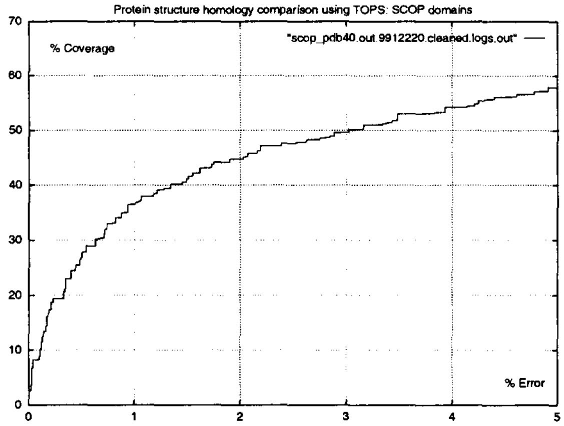


Figure 6: Coverage vs error

System availability: structure comparison server

The comparison system can be used via the Web at tops.ebi.ac.uk/tops. Target structures can be compared against either a database of TOPS diagrams corresponding to all the domains currently in the PDB (currently over 24000 domains) or with a representative subset (the TOPS Atlas Westhead et al. (1998)), based on clustering structures in the structural databank Bernstein et al. (1977); Abola et al. (1987) using the standard single linkage clustering algorithm at 95% sequence similarity, and containing to date over 3000 members.

Users upload a target structure description in PDB format, select a database against which to compare, and enter their email address in order to receive the result. The target description is first analysed using the DSSP program Kabsch and Sander (1983) which locates SSEs and atomic hydrogen bonds. The TOPS program Flores et al. (1994); Westhead et al. (1999) uses this information in a topological analysis which includes analysis of connection chirality; the resulting file is then translated into a TOPS diagram in logic programming format by a compiler we have written in clp(FD) Codognet and Diaz (1996). The comparison is then performed off-line, the result of each comparison comprising the distance measure, the name of the domain compared, and its hierarchic classification according to the CATH system developed at UCL Orengo et al. (1997). The output is sorted by distance from the target protein, and returned to the user by email. Users may also request the output for each comparison to be annotated with the numbers of the corresponding residues and also the common discovered pattern.

The system is fast; a comparison of one structure against the entire PDB (15000 domains) takes from under 10 minutes to 1 hour or more on a DEC Alpha, depending on the complexity of the structure submitted.

4 Conclusions

Although our pattern discovery algorithm produces the richest patterns over α–β domains, when both H-bond and chirality connections can be discovered, it also discovers patterns of H-bonds for all-β domains and patterns of chiralities for all-α domains. However, the null pattern will be discovered when comparing two all-α domains with no chirality information, and thus in this case neither an alignment no a meaningful comparison measure can be computed. The accuracy of the system as measured by coverage against error falls in between those for a well-performing atom-coordinate approach (ranging from 60% coverage at 1% error to 78% coverage at 5% error) and sequence-based approaches (ranging from 16% coverage at 1% error to 18% coverage at 5% error).

A disadvantage of the topological approach is that no RMSD output can be made - the best that can be done is to return the numbers of the matching residues of the matching SSEs, which is not a one to one relationship between residues, but rather between between SSEs which are potentially of different lengths. However, an advantage of our pattern–based declarative approach is that the patterns can be returned to the user - these contain more information than is conveyed by the comparison score alone, for example that both pattern contained a complete barrel.

Finally, our pattern discovery algorithm can be used to make multiple alignments of TOPS structures, since it is is linear time in the number of members of the target set.

Acknowledgements

The authors wish to thank Alvis Brazma of the EBI for his invaluable suggestions, Dan Hatton for his painstaking testing, and the CATH group at UCL for their help in using CATH.

David Gilbert has been partially supported by an EPSRC Visiting Fellowship at the European Bioinformatics Institute whilst on sabbatical from City University.

References

E. E. Abola, F. C. Bernstein, S. H. Bryant, T. F. Koetzle, and J. Weng. Protein Data Bank. In F. H. Allen, G. Bergerhoff, and R. Sievers, editors, *Crystallographic Databases-Information Content, Software Systems, Scientific Applications*, pages 107–132. Data Commission of the International Union of Crystallography, Bonn/Cambridge/Chester, 1987.

P.J. Artymuik, H.M. Grindley, A.R. Poirette, D.W. Rice, E.C. Ujah, and P. Willett. Identification of β-Sheet motifs, of ψ-loops, and of patters of amino acid residues in three dimensional protein structures using a subgraph isomorphism algorithm. *J. Chem. Inf. Comput. Sci.*, 34:54–62, 1994.

F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, E. F. Meyer, Jr., M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi. The Protein Data Bank: a Computer–based Archival File for Macromolecular Structures. *Journal of Molecular Biology*, 112:535–542, 1977.

A. Brazma, I. Jonassen, I Eidhammer, and D. R. Gilbert. Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, 5(2):277–303, 1998.

C. Bron and J. Kerbosch. Algorithm 457: Finding All Cliques of an Uundirected Graph. *CACM*, 16 (9):575–577, 1973.

P. Codognet and D. Diaz. Compiling constraints in clp(FD). *Journal of Logic Programming*, 27(3): 185–226, 1996.

T.P. Flores, D.M. Moss, and J.M. Thornton. An algorithm for automatically generating protein topology cartoons. *Protein Engineering*, 7(1):31–37, 1994.

D. R. Gilbert, D. R. Westhead, N. Nagano, and J. M. Thornton. Motif–based searching in tops protein topology databases. *Bioinformatics*, 15(4):317–326, 1999.

H.M. Grindley, P.J. Artymuik, D.W. Rice, and P. Willett. Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm. *Journal of Molecular Biology*, 229(707–721), 1993.

W. Kabsch and C. Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

I. Koch and T. Lengauer. Detection of distant structural similarities in a set of proteins using a fast graph-based method. In T. Gaasterland et al, editor, *Proceedings of the 5th International Conference on Intelligent Systems in Molecular Biology*, pages 167–178. AAAI Press, 1997.

I. Koch, T. Lengauer, and E. Wanke. An algorithm for finding maximal common subtopologies in a set of protein structures. *Journal of Computational Biology*, 3(2):289–306, 1996.

V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady Akademii nauk SSSR (in Russian)*, 163(4):845–848, 1965. Also in Cybernetics and Control Theory, vol 10, no. 8, pp 707–710, 1996.

E.M. Mitchell, P.J. Artymuik, D.W. Rice, and P. Willett. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *Journal of Molecular Biology*, 212:151–166, 1989.

A.G. Murzin, S.E. Brenner, T. Hubbard, and C. Chotia. **scop**: a structural classification of proteins database for the investigation of sequences and structures. *Journal of Molecular Biology*, 247: 536–540, 1995.

C. A. Orengo, A. D. Michie, D. T. Jones, M. B. Swindells, and J. M. Thornton. CATH – a hirearchic classification of protein domain structures. *Structure*, 5(8):1093–1108, 1997.

M.J.E. Sternberg and J.M. Thornton. On the conformation of proteins: The handedness of the connection between parallel beta strands. *Journal of Molecular Biology*, 110:269–283, 1977.

W.R. Taylor and C.A. Orengo. Protein structure alignment. *Journal of Molecular Biology*, 208:1–22, 1989.

Y. Tsukamoto, K. Takiguchi, K. Satou, T. Furuichi, E. Takagi, and S. Kuhara. Application of a deductive database system to search for topological and similar three-dimensional structures in protein. *CABIOS*, 13(2):183–190, 1997.

D. R. Westhead, D. C. Hutton, and J. M. Thornton. An atlas of protein toplogy cartoons available on the World Wide Web. *Trends in Biochemical Sciences*, 23, 1998.

D. R. Westhead, T. W. F. Slidel, T. P. J. Flores, and J. M. Thornton. Protein structural topology: automated analysis and diagrammatic representation. *Protein Science*, 8(4):897–904, 1999.

Automatically Extracting Enzyme Interaction and Protein Structure Information from Biological Science Journal Articles

Kevin Humphreys ; George Demetriou ; Robert Gaizauskas
Department of Computer Science, University of Sheffield
Regent Court, Portobello Street
Sheffield S1 4DP UK
{kwh ; demetri ; robertg}@dcs.shef.ac.uk

Abstract

With the explosive growth of scientific literature in the area of molecular biology, the need to automatically process and extract information from on-line text sources has become increasingly important. In this paper we consider the application of Information Extraction (IE) technology to the extraction of factual information from biological journal papers. IE has proved successful at extracting information primarily from newswire texts and primarily in domains concerned with human activity as demonstrated by the systems that took part in the U.S. DARPA Message Understanding Conferences (MUCs). We describe how an information extraction system designed to participate in the MUC exercises has been modified for two bioinformatics applications: EMPathIE, concerned with enzyme and metabolic pathways; and PASTA, concerned with protein structure. The progress so far provides convincing grounds for believing that IE techniques will deliver novel and effective ways for the extraction of information from unstructured text sources in the pursuit of knowledge in the biological domain.

1 Introduction

Information Extraction (IE) may be defined as the activity of extracting details of predefined classes of entities and relationships from natural language texts and placing this information into a structured representation called a *template* (Cowie and Lehnert, 1996; Gaizauskas and Wilks, 1998). The prototypical IE tasks are those defined by the U.S. DARPA-sponsored Message Understanding Conferences (MUCs), requiring the filling of a complex template from newswire texts on subjects such as joint venture announcements, management succession events, or rocket launchings (Def, 1995, 1998). While the performance of current technology is not yet at human levels overall, it is approaching human levels for some component tasks (e.g. the recognition and classification of named entities in text) and is at a level at which comparable technologies, such as information retrieval and machine translation, have found useful application. IE is particularly relevant where large volumes of text make human analysis infeasible, where template-oriented information seeking is appropriate (i.e. where there is a relatively stable information need and a set of texts in a relatively narrow domain), where conventional information retrieval technology is inadequate, and where some error can be tolerated.

One area where we believe these criteria are met, and where IE techniques have as yet been applied only in a limited way (though see Fukuda et al. (1998); Rindflesh et al. (2000); Thomas et al. (2000)), is the construction of databases of scientific information from journal articles, for use by researchers in molecular biology. The explosive growth of textual material in this area means that no one can keep up with what is being published. Conventional retrieval technology returns both too little, because of the complex, non-standardised terminology in the area, and too much, because what is sought is not whole texts in which key terms appear, but facts buried in the texts. Further, useful templates can be defined for some scientific tasks. For example, scientists working on drug discovery have an ongoing interest in reactions catalysed by enzymes in metabolic pathways. These reactions may be viewed as a class of events, like corporate management succession events, in which various classes of entities (enzymes, compounds) with attributes (names, concentrations) are related by participating in the event in specific roles (substrate, catalyst, product). Finally, some error can be tolerated in these applications, because scientists can verify the information against the source texts – the technology serves to assist, not to replace, investigation.

Thus, we believe automatically extracting information from scientific journal papers is an important and feasible application of IE techniques. It is also interesting from the perspective of IE research because it extends IE to *domains* and to *text genres* where it has never been applied before. To date most IE applications have been to domains of human activity, predominately economic activity, and have involved newswire texts which have a characteristic lexis, structure and length. Applying IE to scientific journal papers in the area of molecular biology means a radical shift of subject domain away from the world of people, companies, products and places that have largely figured in previous applications. It also means dealing with a text genre in which there is a vast and complex technical vocabulary, where the texts are structured into subsections dealing with method, results, and discussion, and where the texts are much longer. These differences all pose tough challenges for IE techniques as developed so far: can they be applied successfully in this area?

In this paper we describe the use of the technology developed through MUC evaluations in two bioinformatics applications. The next section describes the general functionality of an IE system, and section 3 then describes the two specific applications on which we are working: extraction of information about enzymes and metabolic pathways and extraction of information about protein structure, in both cases from scientific abstracts and journal papers. Section 4 describes the principle processing stages and techniques of our system, and section 5 presents evaluations of the system's performance. While much further refinement of the system for both applications is possible, indications are that IE can indeed be successfully applied to the task of extracting information from scientific journal papers.

2 Information Extraction Technology

The most recent MUC evaluation (MUC-7, (Def, 1998)) specified five separate component tasks, which illustrate the main functional capabilities of current IE systems:

1. *Named Entity recognition* requires the recognition and classification of named entities such as organisations, persons, locations, dates and monetary amounts.
2. *Coreference resolution* requires the identification of expressions in the text that refer to the same object, set or activity. These include variant forms of name expression (*Ford Motor Company ... Ford*), definite noun phrases and their antecedents (*Ford ... the American car manufacturer*), and pronouns and their antecedents (*President Clinton... he*). Coreference relations are only marked between certain syntactic classes of expressions (noun phrases and pronouns) and a relatively constrained class of relationships to mark is specified, with clarifications provided with respect to bound anaphors, apposition, predicate nominals, types and tokens, functions and function values, and metonymy.
3. *Template Element filling* requires the filling of small scale templates (slot-filler structures) for specified classes of entity in the texts, such as organisations, persons, certain artifacts, and locations, with slots such as name (plus name variants), description as supplied in the text, and subtype.
4. *Template Relation filling* requires filling a two slot template representing a binary relation with pointers to template elements standing in the relation. For example, a template relation of `employee_of` containing slots for a person and organisation is filled whenever a text makes clear that a particular person is employed by a particular organisation. Other relations are `product_of` and `location_of`.
5. *Scenario Template filling* requires the detection of relations between template elements as participants in a particular type of event, or scenario (rocket launches for MUC-7), and the construction of an object-oriented structure recording the entities and various details of the relation.

Systems are evaluated on each of these tasks as follows. Each task is precisely specified by means of a task definition document. Human annotators are then given these definitions and use them to produce by hand the 'correct' results for each of the tasks – filled templates or texts tagged with name classes or coreference relations (these results are called *answer keys*). The participating systems are then run and their results, called *system responses*, are automatically scored against the answer keys. Chief metrics are *precision* – percentage of the system's output which is correct (i.e. occurs in the answer key) – and *recall* – percentage of the correct answer which occurs in the system's output.

State-of-the-art (MUC-7) results for these five tasks are as follows (in the form recall/precision): named entity – 92/95; coreference – 56/69; template element – 86/87; template relation – 67/86; scenario template 42/65.

3 Two Bioinformatics Applications of Information Extraction

We are currently investigating the use of IE for two separate bioinformatics research projects. The Enzyme and Metabolic Pathways Information Extraction (EMPathIE) project aims to extract details of enzyme reactions from articles in the journals *Biochimica et Biophysica Acta* and *FEMS Microbiology Letters*. The utility for biological researchers of a database of enzyme reactions lies in the ability to search for potential sequences of reactions, where the products of one reaction match the requirements of another. Such sequences form metabolic pathways, the identification of which can suggest potential sites for the application of drugs to affect a particular end result. Typically, journal articles in this domain describe details of a single enzyme reaction, often with little indication of related reactions and which pathways the reaction may be part of. Only by combining details from several articles can potential pathways be identified.

The Protein Active Site Template Acquisition (PASTA) project aims to extract information concerning the roles of amino acids in protein molecules, and to create a database of protein active sites from both scientific journal abstracts and full articles. The motivation for the PASTA project stems from the need to extract and rationalise information in the protein structure literature. New protein structures are being reported at very high rates and the number of co-ordinate sets (currently about 12000) in the Protein Data Bank (PDB) (Bernstein et al., 1977) can be expected to increase ten-fold in the next five years. The full evaluation of the results of protein structure comparisons often requires the investigation of extensive literature references, to determine, for instance, whether an amino acid has been reported as present in a particular region of a protein, whether it is highly conserved, implicated in catalysis, and so on. When working with several different structures, it is frequently necessary to go through a large number of scientific articles in order to discover any functional or structural equivalences between residues or groups of residues. Computational methods that can extract information directly from these articles would be very useful to biologists in comparison classification work and to those engaged in modelling studies.

The following section describes the EMPathIE and PASTA tasks, including the intended extraction results from documents containing text such as that shown in Figure 1.

> *Results: We have determined the crystal structure of a triacylglycerol lipase from Pseudomonas cepacia (Pet) in the absence of a bound inhibitor using X-ray crystallography. The structure shows the lipase to contain an alpha/beta-hydrolase fold and a catalytic triad comprising of residues Ser87, His286 and Asp264. The enzyme shares several structural features with homologous lipases from Pseudomonas glumae (PgL) and Chromobacterium viscosum (CvL), including a calcium-binding site. The present structure of Pet reveals a highly open conformation with a solvent-accessible active site, This is in contrast to the structures of PgL and Pet in which the active site is buried under a closed or partially opened 'lid', respectively.*

Figure 1: Sample Text Fragment from a Scientific Paper in Molecular Biology

3.1 EMPathIE

One of the inspirations for the Enzyme and Metabolic Pathways application was the existence of a manually constructed database for the same application. The EMP database (Selkov et al., 1996) contains over 20,000 records of enzyme reactions, collected from journal articles published since 1964. That such a database has been constructed and is widely used demonstrates the utility of the application. EMPathie aims to extract only a key subset of the fields found in the EMP database records.

The main fields required in a record of an enzyme reaction are: the enzyme name, with an enzyme classification (EC) number, if available, the organism from which the enzyme was extracted, any known pathway in which the reaction occurs, compounds involved in the reaction, with their roles classified as either substrate (input), product (output), activator, inhibitor, cofactor or buffer, and any compounds known not to be involved in the reaction, with their roles classified as either non-substrate or non-product.

The template definitions include three Template Elements: *enzyme*, *organism* and *compound*, a single Template Relation: *source*, relating *enzyme* and *organism* elements, and a Scenario Template for the specific metabolic pathway task. The Scenario Template describes a pathway involving one or more interactions, each of which is a reaction between an enzyme and one or more participants, possibly under certain constraints. A manually produced sample Scenario Template is shown here, taken from an article on 'isocitrate lyase activity' in FEMS Microbiology Letters.

```
<ENZYME-1> :=
  NAME: isocitrate lyase
  EC_CODE: 4.1.3.1

<ORGANISM-1> :=
  NAME: Haloferax volcanii
  STRAIN: ATCC 29605
  GENUS: halophilic Archaea

<COMPOUND-1> :=
  NAME: phenylhydrazone

<COMPOUND-2> :=
  NAME: KCl

<SOURCE-1> :=
  ENZYME: <ENZYME-1>
  ORGANISM: <ORGANISM-1>

<PATHWAY-1> :=
  NAME: glyoxylate cycle
  INTERACTION: <INTERACTION-1>

<INTERACTION-1> :=
  ENZYME: <ENZYME-1>
  PARTICIPANTS: <PARTICIPANT-1>
                <PARTICIPANT-2>

<PARTICIPANT-1> :=
  COMPOUND: <COMPOUND-1>
  TYPE: Product
  TEMPERATURE: 35C

<PARTICIPANT-2> :=
  COMPOUND: <COMPOUND-2>
  TYPE: Activator
  CONCENTRATION: 1.75 M
```

This template describes a single interaction found to be part of the metabolic pathway known as the *glyoxylate cycle*, where the interaction is between the enzyme *isocitrate lyase* and two other participants. The first participant is the compound `glyoxylate phenylhydrazone`, which has the role of a *product* of the interaction at a temperature of 35C. The second is the compound *KCl*, which has the role of an *activator* at a concentration of 1.75M.

The template design follows closely the MUC-style IE template, and is richer than the EMP database record format in terms of making relationships between entities explicit. However, most of the slot values can still be mapped back to the EMP format to allow an automatic evaluation of system output against the manually constructed EMP resource.

3.2 PASTA

The entities to be extracted for the PASTA task include proteins, amino acid residues, species, types of structural characteristics (secondary structure, quaternary structure), active sites, other (probably less important) regions, chains and interactions (hydrogen bonds, disulphide bonds etc.) In collaboration with molecular biologists we have designed a template to capture protein structure information, a fragment of which, filled with information extracted from the text in Figure 1, is shown below:

```
<RESIDUE-str97-521>:=
  RESIDUE_TYPE:   SERINE
  RESIDUE_NO:     "87"
  IN_PROTEIN:     <PROTEIN-str97-521>
  SITE/FUNCTION:  "active site"
                  "catalytic"
                  "interfacial activation"
                  "calcium-binding site"
  SECOND_STRUCT:  alpha-helix
  REGION:         'lid'
  ARTICLE:        <ARTICLE-str97-521>

<PROTEIN-str97-521>:=
  NAME:           "Triacylglycerol lipase"
  SCOP_CLASS:     "Lipase"
  PDB_CODE:       1LGY
  IN_SPECIES:     <SPECIES-str97-521>

<SPECIES-str97-521>:=
  NAME:           "Pseudomonas cepacia"
  NAME_TYPE:      SCIENTIFIC
```

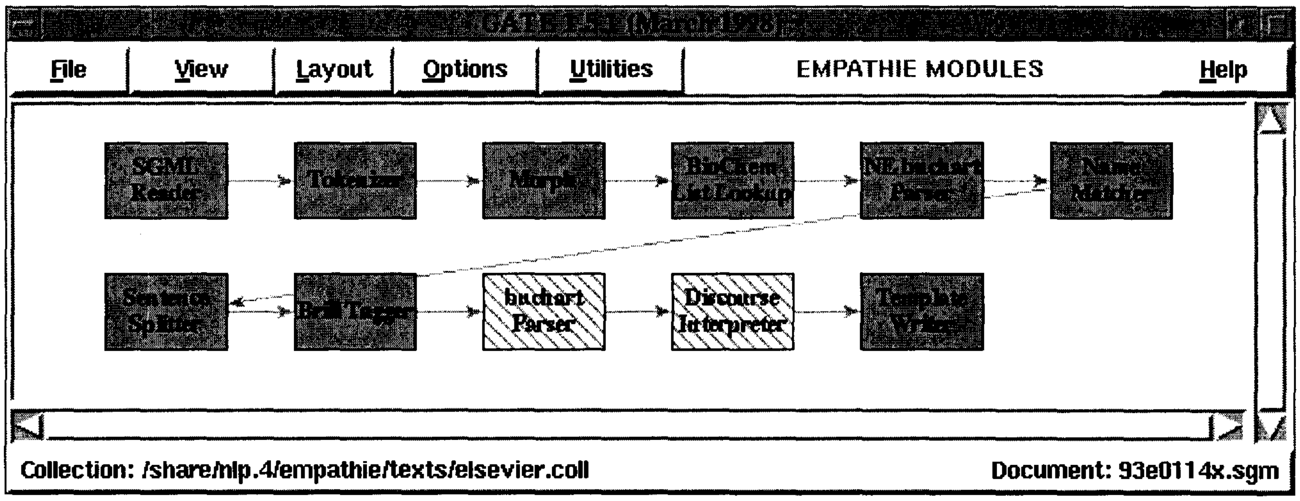
The residue information contains slots that describe the structural characteristics of the particular protein (e.g. SECONDARY structure, REGION) and the importance of the residue in the structure (e.g. SITE/FUNCTION). Other slots serve as pointers, linking different template objects together to represent relational information between entities (e.g. the IN_PROTEIN and IN_SPECIES slots). Further Template Relations can also be defined to link proteins or residues with structural equivalence.

4 The EMPathIE and PASTA Systems

The IE systems developed to carry out the EMPathIE and PASTA tasks are both derived from the Large Scale Information Extraction (LaSIE) system, a general purpose IE system, under development at Sheffield since 1994 (Gaizauskas et al., 1995; Humphreys et al., 1998). One of several dozen systems designed to take part in the MUC evaluations over the years, the LaSIE system more or less fits the description of a generic IE system (Hobbs, 1993). LaSIE is neither as 'deep' as some earlier IE systems that attempted full syntactic, semantic and discourse processing (Hobbs, 1991) nor as 'shallow' as some recent systems that use finite state pattern matching techniques to map directly from source texts to target templates (Appelt et al., 1995). The processing modules which make up the EMPathIE system are shown in figure 2, within the GATE development environment (Cunningham et al., 1997). The PASTA system is similar and reuses several modules, within the same environment. The architecture of the original LaSIE system has been substantially rearranged for its use in the biochemical domain, mainly to allow the reuse of general English processing modules, such as the part-of-speech tagger and the phrasal parser, without special retraining or adaptation to allow for the domain-specific terminology. This has resulted in an independent terminology identification subsystem, postponing general syntactic analysis until an attempt to identify terms has been made. In general, the original LaSIE system modules, developed for newswire applications, have been reused, but with various modifications resulting from specific features of the texts, as described in the following. Both systems have a pipeline architecture consisting of four principal stages, described in the following sections: *text preprocessing* (SGML/structure analysis, tokenisation), *lexical and terminological processing* (terminology lexicons, morphological analysis, terminology grammars), *parsing and semantic interpretation* (sentence boundary detection, part-of-speech tagging, phrasal grammars, semantic interpretation), and *discourse interpretation* (coreference resolution, domain modelling).

4.1 Text Preprocessing

Scientific articles typically have a rigid structure, including abstract, introduction, method and materials, results, and discussion sections, and for particular applications certain sections can be targeted for detailed analysis while others can be skipped completely. Where articles are available in SGML with a DTD, an initial module is used to identify particular markup, specified in a configuration file, for use by subsequent modules. Where articles are in plain text, an initial 'sectioniser' module is used to identify and classify significant sections using sets of regular expressions. Both the SGML and sectioniser modules may specify that certain text regions are to be ex-



Collection: /share/nlp.4/empathie/texts/elsevier.coll

Document: 93e0114x.sgm

Figure 2: EMPathIE system modules within GATE

cluded from any subsequent processing, avoiding detailed processing of apparently irrelevant text, especially within the discourse interpretation stage where coreference resolution is a relatively expensive operation.

The tokenisation of the input needs to identify tokens within compound names, such as abbreviations like *NaCl*, where *Na* and *Cl* need to be matched separately in the lexical lookup stage to avoid listing all possible sequences explicitly. The tokenisation module must therefore make as few assumptions as possible about the input, proposing minimal tokens which may be recombined in subsequent stages.

4.2 Lexical and Terminological Processing

The main information sources used for terminology identification in the biochemical domain are: case-insensitive terminology lexicons, listing component terms of various categories; morphological cues, mainly standard biochemical suffixes; and hand-constructed grammar rules for each terminology class. For example, the enzyme name *mannitol-1-phosphate 5-dehydrogenase* would be recognised firstly by the classification of *mannitol* as a potential compound modifier, and *phosphate* as a compound, both by being matched in the terminology lexicon. Morphological analysis would then suggest *dehydrogenase* as a potential enzyme head, due to its suffix *-ase*, and then grammar rules would apply to combine the enzyme head with a known compound and modifier which can play the role of enzyme modifier.

The biochemical terminology lexicons, acquired from various publicly available resources, have been structured to distinguish various term components, rather than complete terms, which are then assembled by grammar rules. Resources such as the SWISS-PROT list of official enzyme names were manually split into separate lists of component terms, based purely on their apparent syntactic structure rather than any expert knowledge of whatever semantic structure the names reflect. Corresponding grammar rules were then added to recombine the components. Of course, lists of complete multi-word terms can also be used directly in the lexicons, but the rule-based approach has the advantage of being able to recognise novel combinations, not explicitly present in the term lists, and avoids reliance on the accuracy and completeness of available terminology resources. Component terms may also play multiple roles in different terminology classes, for instance amino acid names may be components of both protein and enzyme names, as well as terms in their own right, but the rule-based approach to terminology recognition means they only need to be listed in a single terminology category. The total number of terminology lexicon entries for the biochemical terms is thus comparable to other domains, with approximately 25,000 component terms at present in 52 categories.

4.3 Parsing and Semantic Interpretation

The syntactic processing modules treat any terms recognised in the previous stage as non-decomposable units, with a syntactic role of proper noun. The sentence splitting module cannot therefore propose sentence boundaries within a preclassified term. Similarly, the part-of-speech tagger only attempts to assign tags to tokens which are not part of proposed terms, and the phrasal parser treats terms as preparsed noun phrases. Of course, this approach does not necessarily assume the terminology recognition subsystem to be fully complete and correct, and subsequent syntactic or semantic context can still be used to reclassify or remove proposed terms. In particular, tokens which are constituents of terms proposed but not classified by the NE subsystem, i.e. potential but unknown NEs, are passed to the tagger and phrasal parser as normal, but the potential term is passed to the parser in addition, as a proper name, to allow the phrasal grammar to determine the best analysis. If the unclassified NE is retained after phrasal parsing, it may be classified within the discourse interpreter, using its semantic context or as a result of being coreferred with an entity of a known class.

The phrasal grammar includes compositional semantic rules, which are used to construct a semantic representation of the 'best', possibly partial, parse of each sentence. This predicate logic-like representation is passed on as input to the discourse interpretation stage.

4.4 Discourse Interpretation

The discourse interpreter adds the semantic representation of each sentence to a predefined *domain model*, made up of an ontology, or concept hierarchy, plus inheritable properties and inference rules associated with concepts. The domain model is gradually populated with instances of concepts from the text to become a *discourse model*. A powerful coreference mechanism attempts to merge each newly introduced instance with an existing one, subject to various syntactic and semantic constraints. Inference rules of particular instance types may then fire to hypothesise the existence of instances required to fill a template (e.g. an organism with a *source* relation to an enzyme), and the coreference mechanism will then attempt to resolve the hypothesised instances with actual instances from the text.

The template writer module reads off the required information from the final discourse model and formats it as in the template specification.

Initial domain models for the EMPathIE and PASTA tasks have been manually constructed directly from the template definition. This involves the addition of concept nodes to the system's semantic network for each of the entities required in the template, with subhierarchies for possible subtypes, as required. Property types are added for each of the template slots (e.g. `concentration`, `temperature`), and consequence rules added to hypothesise instances for each slot of a template entity, from an appropriate textual trigger. The Discourse Interpreter's general coreference mechanism is then used to attempt to resolve hypothesised instances with instances mentioned in the text. Subsequent refinement of these models will involve extending the concept subhierarchies and the addition of coreference constraints on the hypothesised instances, based on available training data.

5 Results and Evaluation

5.1 Evaluation

Currently, a complete prototype EMPathIE system exists which can produce filled templates as specified above. This prototype has been developed by concentrating on the full texts of six journal papers (the *development* corpus) and evaluated against a corpus of a further seven journal papers (the *evaluation* corpus). Filled templates for all thirteen of these journal papers were produced by trained biochemists highlighting key entities on paper copies of the texts and adding marginal notes where necessary to specify compound roles in interactions and any additional slot values such as concentration, temperature, etc. The annotations were translated to template format by the system developer (with the system frozen before evaluation texts were seen), but some degree of subjective interpretation was required in this process. The annotation would therefore probably be difficult to reproduce without a detailed task specification document, which would be aided by inter-annotator agreement studies to highlight areas of ambiguity in the task definition. However, the current templates at least have the advantage of being produced with some degree of consistency by the developer alone, and so do allow a useful measure of the system's accuracy.

Overall template filling results are shown in Table 1. The columns show: the number of items the system correctly identified (CORrect), the number of items where the system response and the answer key differed (INCorrect), the number of items the system missed (MISsing), the number the system spuriously proposed (SPUrious) and the standard metrics of RECall and PREcision, discussed in section 2 above. Here "items" refers to filled slot occurrences in the templates. Scoring proceeds by first aligning template objects in the system response with objects in the answer key and then counting the number of matching slot fills in the aligned objects (see Def (1998) for details).

Test Set	COR	INC	MIS	SPU	REC	PRE
Dev	150	121	330	61	25	45
Eval	213	193	518	93	23	43

Table 1: Initial Template results for EMPathIE

In addition to evaluating the template filling capabilities of the prototype we have evaluated its performance at correctly identifying and classifying term classes in the texts (this corresponds to the MUC named entity task). To do this six of the seven evaluation corpus articles were manually annotated for eleven terminology or named entity classes. The results are shown in Table 2 [1].

The PASTA system has been implemented as far as the terminology recognition stage. Preliminary template design, as indicated above, has been carried out, and we are starting to build a domain model. A corpus of 52 abstracts of journal articles has been manually annotated with terminology classes, by the system developer with

[1] In calculating both EMPathie and PASTA terminology results we have used a weak criterion of correctness whereby a response is correct if its type matches the type of the answer key and its text extent matches a substring of the key's extent. Insisting on the stronger matching criterion of strict string identity lowers recall and precision scores by approximately 4 % overall

Name_Type	COR	INC	MIS	SPU	REC	PRE
compound	100	6	156	3	38	92
element	22	0	17	0	56	100
enzyme	136	0	2	13	99	91
gene	0	0	2	0	0	0
genus	15	0	0	9	100	63
location	11	0	3	10	79	52
measure	157	0	49	11	76	93
organism	59	0	26	23	69	72
organizatio	8	2	7	4	47	57
pathway	0	0	10	1	0	0
person	7	1	13	1	33	78
TOTALS	515	9	285	75	64	86

Table 2: Initial Named Entity results for EMPathIE

the assistance of a molecular biologist, to allow an automatic evaluation of the PASTA terminology system using the MUC scoring software. Table 3 shows some preliminary results for the main terminology classes.

Name_Type	COR	INC	MIS	SPU	REC	PRE
protein	358	0	52	12	87	97
species	111	0	22	3	83	97
residue	175	0	4	13	98	93
site	53	0	34	10	61	84
region	19	0	24	0	44	100
2_struct	78	0	1	1	99	99
sup_struct	84	0	0	5	100	94
4_struct	115	0	5	3	96	97
chain	27	0	12	0	69	100
base	38	0	0	1	100	97
atom	42	0	2	10	95	81
non_protein	107	0	0	21	100	84
interaction	10	0	3	1	77	91
TOTALS	1217	0	159	80	88	94

Table 3: Initial Named Entity results for PASTA

5.2 Discussion

It should be stressed that these evaluation results are very preliminary, and we would expect them to improve substantially with further development.

The overall EMPathIE template filling precision scores for both the development and evaluation sets are very close to the score of the LaSIE system in the MUC-7 evaluation (42%). Recall is noticeably lower however (47% in MUC-7), but this is certainly affected by the limited amount of training data available, giving a much smaller set of key words and phrases to use as cues for template fills. Also, it is clear that the EMPathIE task requires much more specialist domain-specific knowledge than the MUC tasks, which typically require only general knowledge of companies and business procedures. The EMPathIE task, as the process of manually filling the templates has demonstrated, can only be performed with the use of detailed domain knowledge, very little of which has been incorporated into the system. For example, a single mention of 'cyanide' in one of the evaluation texts causes its entry as an 'inhibitor' in the manually filled template, though no explicit information in the text would allow it to be classified as such. Only domain-specific knowledge that cyanide is usually an inhibitor allows it to be classified in this case. Such cases are missed completely by the system because the specific knowledge required has not been entered, mainly due to the fact that the developer is not an expert in the domain.

Further consultation with experts would allow more domain-specific information to be entered, improving recall in particular. With this, and a more extensive training set, it should be entirely possible for system performance on the EMPathIE task to equal the best MUC-7 scores (48% recall, 68% precision, from different systems).

The terminology recognition results are more encouraging, and compare favourably with MUC named entity results, particularly the PASTA results. It should be noted that both the EMPathIE and PASTA terminology recognition tasks require the recognition of a considerably broader class of terms than the MUC named entity task and that considerably smaller sets of training data were available. The discrepancy between the EMPathIE and PASTA results on this task can probably be explained by the fact that there was in fact no training data available specifically for the EMPathIE task before the evaluation was carried out, only the informal feedback of biologists looking at system output. Furthermore, the annotation of texts for the EMPathIE terminology task was carried out by a larger group of people than carried out the PASTA annotation task and without a formal annotation specification. Thus, this annotated data is almost certainly less consistently annotated and the results should therefore be interpreted with some caution.

6 Conclusion

Between these two projects much of the low-level work of moving IE systems into the new domain of molecular biology and the new text genre of journal papers has been carried out. We have generalised our software to cope with longer, multi-sectioned articles with embedded SGML; we have generalised tokenisation routines to cope with scientific nomenclature and terminology recognition procedures to deal with a broad range of molecular biological terminology. All of this work is reusable by any IE application in the area of molecular biology.

In addition we have made good progress in designing template elements, template relations, and scenario templates whose utility is attested by working molecular biologists and in adapting our IE software to fill these templates. Preliminary evaluations demonstrate the difficulty of the task, but results are encouraging, and the steps to take to improve performance straightforward. Thus, we are optimistic that IE techniques will deliver novel and effective ways for scientists to make use of the core literature which defines their disciplines.

Acknowledgements

EMPathIE is a 1.5 year research project in collaboration with, and funded by, GlaxoWellcome plc and Elsevier Science. The authors would like to thank Dr. Charlie Hodgman of GlaxoWellcome for supplying domain expertise and Elsevier for supplying electronic copy of relevant journals. PASTA is funded under the UK BBSRC-EPRSC BioInformatics Programme (BIF08754) and is a collaboration between the Departments of Computer Science, Information Studies and Molecular Biology and Biotechnology at the University of Sheffield. The authors would like to thank Dr. Peter Artymiuk of the University of Sheffield for supplying his expertise in protein structures.

References

D. Appelt, J. Hobbs, J. Bear, D. Israel, M. Kameyama, A. Kehler, D. Martin, K. Myers, and M. Tyson. SRI International FASTUS system: MUC-6 Test Results and Analysis. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)* Def (1995), pages 237–248.

F. Bernstein, T. Koetzle, G. Williams, E.J. Meyer, M. Brice, J. Rodgers, O. Kennard, M. Shimanouchi, and M. Tasumi. The protein data bank: A computer-based archival file formacromolecular structures. *Journal of Molecular Biology*, (112):535–542, 1977.

J. Cowie and W. Lehnert. Information extraction. *Communications of the ACM*, 39(1):80–91, 1996.

H. Cunningham, K. Humphreys, R. Gaizauskas, and Y. Wilks. Software Infrastructure for Natural Language Processing. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP-97)*, pages 237–244, 1997. Available as `http://xxx.lanl.gov/ps/9702005`.

Defense Advanced Research Projects Agency. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*. Morgan Kaufmann, 1995.

Defense Advanced Research Projects Agency. *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, 1998. Available at `http://www.saic.com`.

K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. Information extraction: Identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing '98 (PSB'98)*, pages 707–718, Hawaii, 1998.

R. Gaizauskas, T. Wakao, K Humphreys, H. Cunningham, and Y. Wilks. Description of the LaSIE system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)* Def (1995), pages 207–220.

R. Gaizauskas and Y. Wilks. Information Extraction: Beyond Document Retrieval. *Journal of Documentation*, 54(1):70–105, 1998.

J.R. Hobbs. Description of the TACITUS system as used for MUC-3. In *Proceedings of the Third Message Understanding Conference MUC-3*, pages 200–206. Morgan Kaufmann, 1991.

J.R. Hobbs. The generic information extraction system. In *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, pages 87–91. Morgan Kaufman, 1993.

K. Humphreys, R. Gaizauskas, S. Azzam, C Huyck, B. Mitchell, H. Cunningham, and Y. Wilks. Description of the LaSIE-II system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)* Def (1998). Available at `http://www.saic.com`.

T. Rindflesh, L. Tanabe, J. Weinstein, and L. Hunter. Edgar: Extraction of drugs, genes and relations from the biomedical literature. In *Proceedings of the Pacific Symposium on Biocomputing '2000 (PSB'2000)*, pages 517–528, Hawaii, 2000.

E. Selkov, S. Basmanova, T. Gaasterland, I. Goryanin, Y. Gretchkni, N. Meltsev, V. Nenashev, R. Overbeek, E. Panyushkina, L. Pronevitch, E. Selkov, and I. Yunis. The metabolic pathway collection from EMP: The enzymes and metabolic pathways database. *Nucleic Acids Res.*, (24):26–28, 1996.

J. Thomas, D. Milward, C. Ouzounis, S. Pulman, and M. Carroll. Automatic extraction of protein interactions from scientific abstracts. In *Proceedings of the Pacific Symposium on Biocomputing '2000 (PSB'2000)*, pages 541–551, Hawaii, 2000.

PREDICTING FUNCTION FROM STRUCTURE: EXAMPLES OF THE SERINE PROTEASE INHIBITOR CANONICAL LOOP CONFORMATION FOUND IN EXTRACELLULAR PROTEINS

Richard M. Jackson [1*] and Robert B. Russell[2]

[1] Department of Biochemistry and Molecular Biology, University College, Gower Street, London
WC1E 6BT, UK.
email jackson@biochem.ucl.ac.uk;

[2] Bioinformatics Research Group, SmithKline Beecham Pharmaceuticals Research and Development
New Frontiers Science Park (North), Harlow, Essex CM19 5AW, UK.
email russelr1@mh.uk.sbphrd.com;

Abstract

Methods for the prediction of protein function from structure are of growing importance in the age of structural genomics. Here we focus on the problem of identifying sites of potential serine protease inhibitor interactions on the surface of proteins of known structure. Given that there is no sequence conservation within canonical loops from different inhibitor families we first compare representative loops to all fragments of equal length among proteins of known structure by calculating main-chain RMS deviation. Fragments with RMS deviation below a certain threshold (hits) are removed if residues have solvent accessibilities appreciably lower than those observed in the search structure. These remaining hits are further filtered to remove those occurring largely within secondary structure elements. Likely functional significance is restricted further by considering only extracellular protein domains. By comparing different canonical loop structures to the protein structure database we show that the method was able to detect previously known inhibitors. In addition, we discuss potentially new canonical loop structures found in secreted hydrolases, toxins, viral proteins, cytokines and other proteins. We discuss the possible functional significance of several of the examples found, and comment on implications for the prediction of function from protein 3D structure.

1 Introduction

Recent years have seen a dramatic increase in the number of known protein three-dimensional structures. The speed with which protein structures are now determined means that structures can often be solved prior to any knowledge of function (e.g. Yang et al, 1998). These occurrences make methods that predict details of protein function from structure of great importance, particularly in light of the projects now underway to solve structures with the aim of determining function (e.g. see Orengo et al., 1999; references therein).

There are several means to predict functional details from structure. Proteins that adopt a similar fold, even in the absence of any sequence similarity, frequently perform similar functions (Murzin et al., 1995; Orengo et al., 1997), and even when functions are different, certain folds show a preference for binding site location (or supersites; Russell et. al., 1998). If overall structure similarity fails to provide functional information, or if a protein adopts a new fold, possible functional inferences may come from other methods. For example, functional details can be predicted by the analysis of protein surfaces for possible binding clefts (Orengo et al., 1999), from the detection of recurring side-chain patterns (Russell, 1998; Artymiuk et al 1994), or the recurrence of short motifs (even in different folds) that can indicate functional details such as the phosphate binding P-loop (Swindells, 1993) or DNA binding helix-turn-helix (Doherty et al.,1996). Typically, these motifs or patterns contain key residue identities that are critical to a particular function (e.g. Glycine residues in the P-loop, or the Asp/His/Ser catalytic triad).

A different phenomenon is observed within the canonical serine protease inhibitor loops. For these loops, a common function (namely serine protease inhibition) is maintained despite no similarity in amino acid sequence, and despite the fact that the loops occur in clearly non-homologous protein structures (i.e. those adopting different folds). The inhibitors have representatives in the all-α, all-β, α + β, and Small protein classes, representing nine fold families and 12 superfamilies in total (according to the SCOP classification Murzin et al. 1995). Rather than being conferred by a specific combination of side-chains, the inhibitory function of the loop appears to be due to a highly constrained main-chain conformation (Laskowski & Kato, 1980). Bode & Huber (1992) define the characteristic substrate-like canonical loop as involving six binding residues P3-P2-P1-P1'-P2'-P3' (nomenclature from Schechter & Berger, 1967) where the P1-P1' peptide bond represents the site of proteolytic cleavage. When analysed in different crystallisation states, complexation states and by NMR, the motif consists of a main-chain conformation of polyproline II at P2 and P1', an approximate 3_{10}-helix at P1 and parallel β-strand at P2' and P3'. This common motif is presumed to mimic the productive bound substrate conformation, being an intrinsic property of the inhibitors (Bode & Huber, 1992).

Analysis of protein-protein interaction energies in a series of non-homologous serine protease-canonical inhibitor and antibody-antigen complexes showed that the two most commonly studied models for protein-protein molecular recognition interact by a fundamentally different mechanism in terms of main-chain and side-chain contributions (Jackson, 1999). The energetics of protease-inhibitor interactions are dominated by the main-chain to main-chain interaction, whilst the antibody-antigen recognition is largely determined by side-chain interactions. This may reflect the differing roles played by the two classes of protein. The serine protease inhibitors bind tightly to their target enzymes and this may be best achieved by a constrained main-chain conformation which recognises the main-chain conformation of its target. The inhibitor is highly committed to the enzyme in an evolutionary sense, since a change in the main-chain conformation of the target would compromise binding. However the arrangement is highly entropically favourable since the primary determinate for binding (the main-chain) is generally more highly constrained than the side-chain degrees of freedom.

Here we assess the possibility of using this local structurally similar main-chain motif (where side-chain information is absent) to search a protein 3D database. This is used in conjunction with a residue accessibility filter as a structural fingerprint for inferring protein function. The aim is to assess to what extent it is possible to predict protease binding sites using structure similarity. This increasing realisation that protein recognition often involves structurally similar motifs means that structural similarity will be of increasing use in predicting biomolecular interactions

2 Methods

2.1 Database search for the canonical loop conformation

The co-ordinates of a probe structure (here the canonical loop conformation) are compared to each segment of the same length with the first occurring representative structure from the Structural Classification of Proteins (SCOP) database (Murzin et al, 1995). The database contained one representative from each *species* level in SCOP (version 1.38), yielding a total of 3495 protein domains. For each residue in the structure, a secondary structure assignments from the DSSP database (Kabsch & Sander, 1983), and a relative solvent accessibility calculated using the program NACCESS (Hubbard & Thornton, 1993) were obtained. The accessibilities were calculated by considering each domain in isolation (i.e. outside of any multimeric, or multidomain context). The relative accessibility of residue X is given by dividing its accessibility by the accessibility of that residue type in an extended Ala-X-Ala tripeptide. For each segment in each representative structure, the RMSD with main-chain atoms of the probe was calculated and values above a cutoff were discarded. For remaining hits the relative solvent accessibilities are compared. For all residues in the database segment, it is required that:

$$\frac{(\mathrm{RA(d)} - \mathrm{RA(p)})}{\mathrm{RA(p)}} \geq -0.33$$

where RA(d) and RA(p) denote the relative accessibilities for the database and probe residues respectively. This essentially requires that each database residue have a relative accessibility that is not less than a third smaller than that observed at the equivalent position in the probe.

2.2 Calculation of P values

P values are reported for each RMSD. For the four and five residue loop from bovine pancreatic trypsin inhibitor (1bpi) we extracted all matches in the PDB having an RMSD smaller than 10.0 angstroms and satisfying the accessibility filter. These values were used to derive the empirical distribution function P(x) (this is the proportion of values where RMSD < x). We found that log(P) was linearly related to 1/x for 0.85 < P(x) < 0.98, implying exponential decay in the upper tail of the distribution. A linear model was used to derive the approximate P value for any observed RMSD as described in Russell (1998).

2.3 Docking test

Docking was performed by superimposing the corresponding main-chain atoms of the match onto those of BPTI in the β-trypsin-BPTI complex (PDB code 2ptc). BPTI was removed from the newly generated β-trypsin-protein complex and only Cα atoms were further considered. A maximum of two bad (distance < 3.5 Å) inter-protein Cα->Cα contacts (including those of the putative loop) are allowed, otherwise the complex fails the docking test. Such a small number of bad contacts could conceivably be relieved by slight adjustment of molecular positions or local main-chain conformational change without compromising loop binding.

3 Results

The representative canonical serine protease inhibitors are given in Table 1 for each of the 12 superfamilies, representing nine folds: All-alpha (1), All-beta (2), alpha + beta (2), small protein (4) according to the SCOP classification (Murzin et al. 1995). A representative canonical loop (residues P2-P3' and P2-P2') was searched against the database for each particular fold. For clarity the results of searching a single canonical loop representative (SCOP *Class*: Small proteins, *Fold*: BPTI-like, *Superfamily*: Kunitz-type inhibitors) from uncomplexed BPTI (1bpi) against the representative set of protein domains is given in Figure 1 for the five residue (P2-P3') segment (Residues: Cys 14 - Ile 18) both before (A) and after (B) applying the residue accessibility as a filter. For hits with an RMSD smaller than 1.0 Å, the accessibility filter has a dramatic effect on the sensitivity of the search. Applying the filter reduces the number of false hits ("false": not known to be a protease inhibitor) from 2162 -> 91 while only slightly reducing the hits from known serine protease inhibitors (16 -> 15). Although the five residue (P2-P3') canonical loop fragment of BPTI is highly discriminating (when RMSD and accessibility are combined) the 15 hits represent only four superfamilies and four folds. More folds and superfamilies known to contain the cannonical loop are found if shorter loop fragments are searched against the database, but this also increases the number of false hits. For the four residue (P2-P2') canonical loop fragment of BPTI there are 686 false and 24 true hits that have RMSD <1.0 Å and also pass the accessibility filter. The coverage for this search includes 9 of the 12 superfamilies and 6 of the 9 folds. The best hit for each superfamily is given in Table 1. The three superfamilies/folds with RMSD > 1.0 Å are Ecotin which gives a weak hit (RMSD 1.2 Å) and the bifunctional inhibitor and ascaris which do not give hits <1.5 Å. The three folds have only one representative protein in the database, which are either low resolution Xray structures (bifunctional inhibitor (1jfo) and ecotin (1ecz), Resolution: 3.3 Å and 2.7 Å respectively) or NMR (ascaris (1ata)). Analysis of NMR or low resolution crystal structures is problematic when using RMSD fit. Using more stringent RMSD criteria (hits must have < 0.8 Å RMSD) the number of false hits falls to 145 (and true hits to 22) whilst all 9 superfamilies are still represented. Hence, the search can be made sensitive enough to include most of the known serine protease inhibitor canonical loops whilst inlcuding relatively few false hits.

Figure 1b and Table 1 show that the methodology employed here can be used to identify the recognized canonical serine protease inhibitors. However, the method has also found numerous hits in proteins not known to inhibit proteases, and these may represent new sites of protease interaction. Some confirmation of this idea comes from considering the list of false hits that pass both filters. Two examples that stand out as having a clear inhibitory function but not included in the initial search are (1) a synthetic peptide (1smf chain I) based on a fragment of the bowman-birk inhibitor (Figure 2a) coming under the peptides Class (Thr10-Ile13 to 1bpi P2-P2', RMSD: 0.41 Å) and (2) HL collagenase from common cattle grub (2hlc chain A) which has two hits one of which (Figure 2b) is a loop (Asp 37B-Arg 39 to 1bpi P2-P2', RMSD: 0.94 Å) binding in the active site of the symmetry related dimer, in a self inhibitory mechanism.

The recognized canonical serine protease inhibitors conform to certain other criteria in addition to possessing a loop of defined conformation and solvent accessibility. Accordingly, the following criteria were applied to screen hits that were least likely to perform an inhibitory function. In order to be considered further hits had to: (1) be in extracellular proteins; (2) not near an N- or C-termini (unless constrained by a disulphide bridge) or in a poorly determined region of a structure (e.g. disordered region in NMR structure); (3) not be in a peptide or theoretical model; (4) not be in a β-strand or β-hairpin structure and (5) be appreciably protruding i.e. in a convex region of the protein. This last constraint was imposed after visual inspection and therefore does contain a degree of subjectivity.

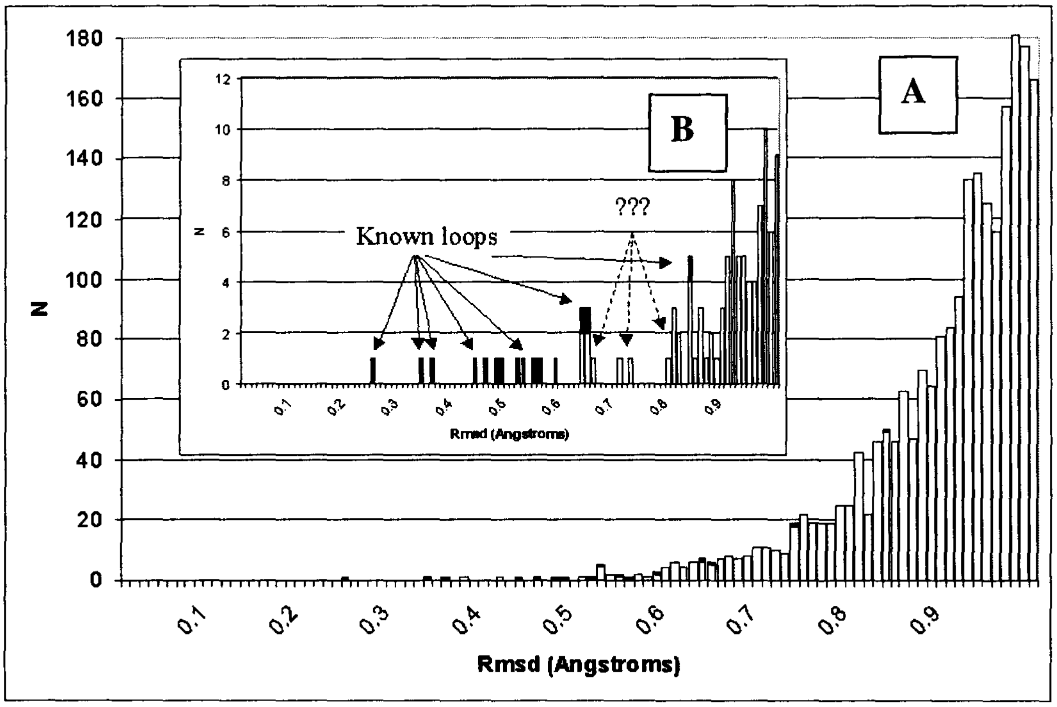


Figure 1

Search using the canonical loop (P2-P3') from uncomplexed BPTI (1bpi) against a representative set of protein domains both before (A) and after (B) applying residue accessibility as a filter. Black bar: serine protease inhibitor, White bar: not known to be a serine protease inhibitor.

Table 1 Known canonical serine protease inhibitors[a] and the first representative "hit" for a given superfamily[b]

Class	Fold	Superfamily	RMSD	P[c]	PDBcode	chain	Range	Sequence	DSSP	Acc.
(probe)					1bpi		14-17	CKAR	S---	3948
All Alpha	Bifunctional inhibitor	Bifunctional inhibitors	N/A	N/A	1jfo					
All Beta	beta-Trefoil	STI-like	0.771	0.013	1ba7	A	62-65	YRIR	SS-S	5937
	Ecotin	Ecotin,trypsin inhibitors	1.203	0.164	1ecz	A	83-86	TMMA	----	4878
Alpha + beta	CI-2 family	CI-2 family of inhibitors	0.275	3e-08	2sni	I	58-61	TMEY	E-B-	6937
	Subtilisin inhibitor	Subtilisin inhibitors	0.372	6e-06	3sgb	I	17-20	TLEY	E--B	7846
Small proteins	BPTI-like	Kunitz-type inhibitors	0.254	7e-09	1tfx	C	14-17	CRGY	B---	4948
	Knottins	Plant inhibitors	0.448	8e-05	1ppe	I	4- 7	PRIL	E---	3957
		Bowman-Birk inhibitors	0.440	6e-05	1tab	I	25-28	TKSM	BSSS	5948
		Elafin-like	0.703	0.006	1fle	I	23-26	CAML	ES-S	2948
		Leech antihemostatic	0.508	3e-04	1hia	I	29-32	CRIR	ES--	3959
	Ovomucoid/PCI-1	Ovomucoid/PCI-1	0.377	8e-06	2sic	I	72-75	PMVY	E---	6867
	Ascaris	Ascaris trypsin inhibitors	N/A	N/A	1ata					

[a] Class, Fold and Superfamily. Classified according to the SCOP database (Murzin et al. 1995)

[b] Search performed with residue 14 – 17 (the P2-P2' segment) of BPTI (1bpi) screened according to residue solvent accessibility. Each "hit" is shown in terms of the protein PDB code, the chain identifier, the main-chain RMSD, the sequence match, the DSSP secondary structure match, and the residue accessibility match.

[c] P values (see methods for details)

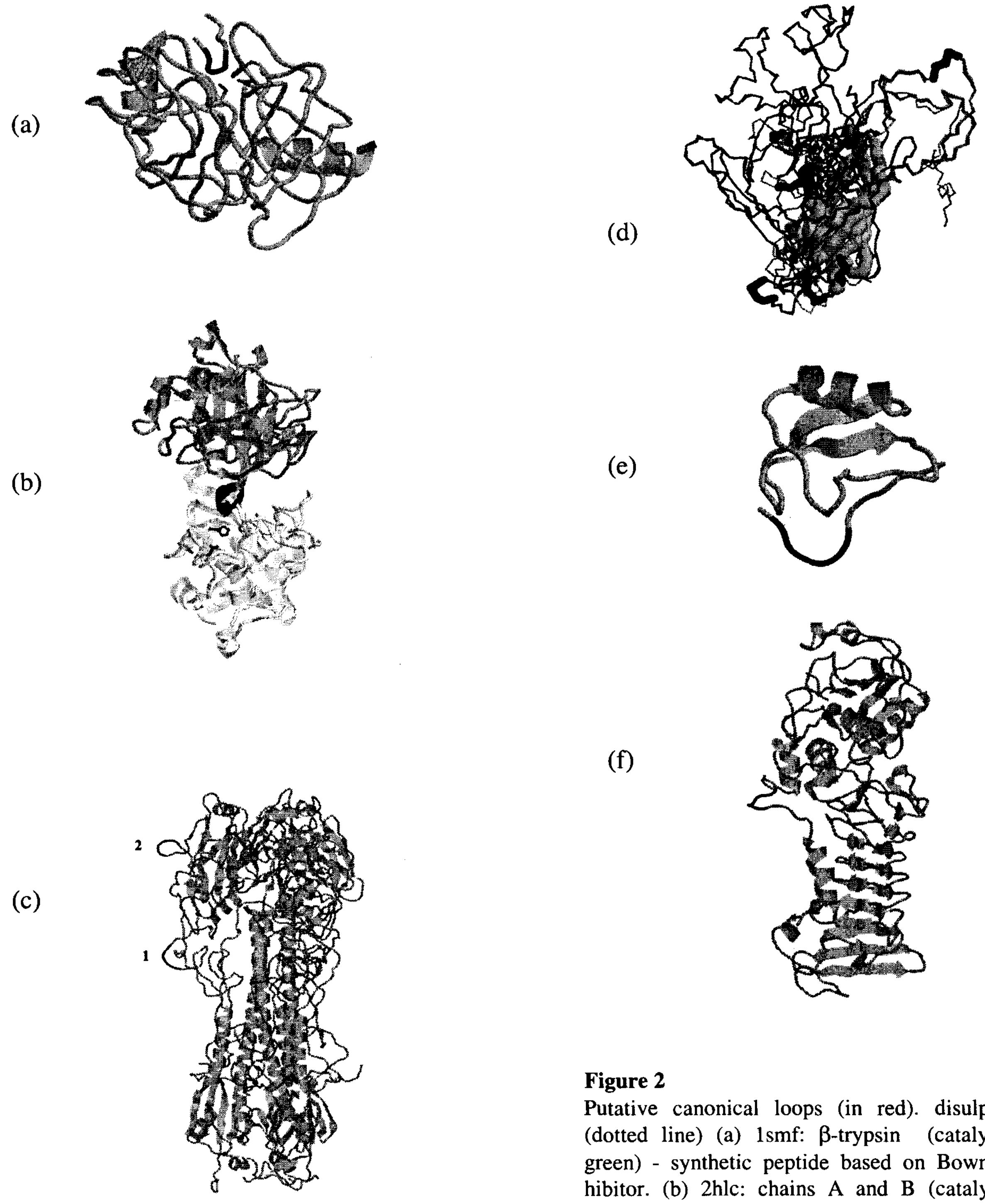


Figure 2
Putative canonical loops (in red). disulphide bonds (dotted line) (a) 1smf: β-trypsin (catalytic triad in green) - synthetic peptide based on Bowman-birk inhibitor. (b) 2hlc: chains A and B (catalytic triad in green). (c) 1hgi: high (1) and low (2) resolution hits. (d) Superimposed viral coat proteins (1rla1 (orange), 2plv1 (green), 1tme1 (cyan), 2cas (light blue), 1fpv (blue), 1cov1 (yellow)). Grey region indicates their common (beta-jelly roll) core structure. (e) 1aho: Toxin II from the Scorpion toxin-like superfamily (f) 1kapp: Zinc Metalloprotease from the Zincin superfamily (catalytic Zn-ion in green).

Table 2 Putative canonical loops[a]

Superfamily	RMSD	P[b]	probe[c]	code chain	class	Range	Sequence	DSSP	Acc.	B[d]	D[e]
Probe				1bpi		14-17/18	CKARI	S---E	39483		
Viral proteins:											
Viral coat proteins	0.551	7e-04	4	1r1a1	beta	276- 279	TAIV	-S--	4888		
Viral coat proteins	0.659	0.004	4	2plv1	beta	146- 149	NNGH	----	5948	X	P
Viral coat proteins	0.714	0.007	4	1tme1	beta	135- 138	GTDT	S-S-	5976		P
Viral coat proteins	0.719	0.008	4	2plv1	beta	38- 41	SKEI	SS--	5986		P
Viral coat proteins	0.753	0.011	4	1r1a1	beta	202- 205	GDNT	-SST	4969		P
Viral coat proteins	0.877	0.030	4	2cas	beta	390- 393	TTGE	--S-	3948		P
Viral coat proteins	0.897	0.035	4	1fpv	beta	474- 477	FDTD	--SS	6959		
Viral coat proteins	0.944	0.047	4	2cas	beta	514- 517	ASAN	-SS-	4978		
Viral coat proteins	0.957	0.051	4	2cas	beta	512- 515	PDAS	TT-S	7949		P
Viral coat proteins	0.935	0.008	5	2plv1	beta	144- 148	ETNNG	SS---	59594	X	P
Viral coat proteins	0.952	0.009	5	1cov1	beta	25- 29	SEAIP	SS--S	69665		P
Segmented RNA-genome virus	0.728	0.008	4	1hgia	beta	275- 278	DTCI	----	3737		P
Scaffolding protein gpD	0.909	0.006	5	1al01	alpha	109- 113	NGVER	HT---	78582		P
RNA bacteriophage capsid	0.923	0.007	5	1unaa	alpha+beta	97- 101	ATDDV	TT---	99673		P
Trypsin-like	0.985	0.011	5	1vcpa	beta	156- 160	AKLAF	HTS--	38282		
Trypsin-like	0.993	0.012	5	1kxf	beta	152- 156	SKLKF	GGS--	38292		
Toxins:											
Yeast killer toxins	0.626	0.002	4	1kvdab	alpha+beta	29- 32	TTIA	----	2946		P
Bacterial enterotoxins	0.657	0.004	4	2chbd	beta	30- 33	SLAG	E-ST	3948		P
Bacterial enterotoxins	0.706	0.007	4	1ltsd	beta	30- 33	SMAG	E-ST	2957		P
Galactose-binding domain	0.882	0.031	4	1dlc	beta	555- 558	VSYS	-SS-	2966	X	P
Snake toxin-like	0.909	0.037	4	1nea	Small	8- 11	SSQP	TTS-	6857	X	
Scorpion toxin-like	0.900	0.006	5	1aho	Small	59- 63	GPGRC	-SS--	39682		P
Hydrolases:											
Trypsin-like	0.817	0.019	4	1tomlh	beta	73- 76	RTRY	SS--	4667		
Trypsin-like	0.807	0.017	4	1autc	beta	145- 148	SSRE	-S--	4878	X	
Trypsin-like	0.913	0.039	4	1danh	beta	170F-170I	GDSP	TT--	9945	X	
Trypsin-like	0.955	0.050	4	1danh	beta	60- 60C	DKIK	TT--	4929	X	
Acid proteases	0.881	0.031	4	1epne	beta	250- 253	CSAT	TT--	3939	X	
Acid proteases	0.904	0.036	4	1mpp	beta	289- 297	DGGN	ESSS	4956	X	
Acid proteases	0.904	0.006	5	1zap	beta	333- 337	TSASS	-S---	49464		
Thermolysin-like, C-terminal	0.831	0.021	4	1npc	alpha	222- 225	YTGS	--SS	4848	X	
Thermolysin-like, C-terminal	0.977	0.010	5	8tlne	alpha	221- 225	YTGTQ	--SSH	28387	X	P
Zincins, catalytic domain	0.888	0.033	4	1ast	alpha+beta	49- 52	TTES	SS-S	7837		
Zincins, catalytic domain	0.838	0.003	5	1kapp	alpha+beta	20- 24	GDELV	SSSEE	89773		P
alpha/beta-Hydrolases	0.843	0.003	5	1akn	alpha/beta	275- 279	GSTEY	SS--S	69698	X	
alpha/beta-Hydrolases	0.939	0.008	5	1ac5	alpha/beta	421- 425	STDDS	TT---	78384	X	
Subtilases	0.937	0.045	4	2prk	alpha/beta	119- 122	NNRN	GGS-	8739		
beta-Lactamase	0.901	0.036	4	3blm	Mul t	270- 273	KSDK	TT--	6828		P
Lysozyme-like	0.641	2e-04	5	1hfx	alpha+beta	65- 69	TTVQS	SS---	49392	X	
Starch-binding domain	0.658	3e-04	5	1kum	beta	601- 605	PQACG	---SS	48366	X	
Cystatin	0.730	0.009	4	1cewi	alpha+beta	36- 39	SNDK	S--S	3936		
Cystatin	0.940	0.046	4	1stfi	alpha+beta	118- 121	KHDE	TTS-	5837		P
Kringle modules	0.950	0.049	4	2hppp	Small	335- 338	KDQD	SSS-	6849	X	
Serpins	0.684	0.005	4	9apiab	Mult	278- 281	NEDR	----	2676	X	
Cytokines or horomones:											
Interleukin 8-like	0.853	0.025	4	3il8	alpha+beta	13- 16	YSKP	--S-	3766		
Neurophysin II	0.851	0.004	5	1npoa	beta	50- 54	LPSPC	-SS-B	39462		P
4-helical cytokines	0.858	0.026	4	3inkc	alpha	40- 43	LTFK	TTS-	3647		
4-helical cytokines	0.938	0.008	5	2ilk	alpha	43- 47	LDNLL	--S-S	69768		P
Cystine-knot cytokines	0.892	0.033	4	1hcna	Small	72- 75	GGFK	SS-E	8659	X	P

[a] See [b] in Table 1 for column definitions. Entries in bold pass the protein docking test (see methods).

[b] The P values were calculated for the appropriate P(x) distributions for the four or five residue data (see Table 1).

[c] Loops with a crystallographic Probe length, with the four and five-residue probes corresponding to 1bpi (14-17) and 1bpi (14-18) respectively.

[d] Loops with a crystallographic B-factor >50Å^2 for at least one of the main-chain atoms or determined by NMR are indicated by "X".

[e] Entries marked P pass the docking test.

The lack of knowledge of the nature of the protease (or proteases) to which any putative inhibitor binds precludes rejection based soley on a docking study involving an arbitrary serine protease (e.g. HL Collagenase inhibits itself by binding in a relatively shallow active site cleft). Moreover, there remains the possibility of conformational changes prior to a putative loop inhibiting a protease. It is nevertheless informative to know which of the loops above pass a simple docking test. Accordingly, we applied a docking test to indicate whether the putative inhibitor loop is sterically compatible with binding to β-trypsin. Thus, to pass this test the protein must posses a suitably exposed loop and must bind in such a way that the rest of the subunit does not appreciably overlap with any part of the trypsin Cα skeleton. The representatives from each known serine protease inhibitor superfamily (Table 1) pass the docking test with the exception of ascaris.

Table 2 shows those hits found during the search with the four and five residue loop from BPTI that are not known to be protease inhibitors and conform to the 5 requirements given. Entries in the table marked P (in colomn D) pass the docking test. Loops with a high crystallographic B-factor (>50Å^2) for at least one of the main-chain atoms or determined by NMR are also indicated. Inspection of the results showed that they could be divided into five categories: 1) viral proteins (16); 2) secreted toxins (6); 3) secreted hydrolases (21); 4) secreted cytokines (5) and 5) others (9). The most interesting hits are discussed in the sections below.

3.1 Viral Proteins

Numerous hits were found in viral proteins. Perhaps the most striking match is that within influenza virus hemagglutinin (PDB code 1hgi chain A; Asp 275- Ile 278, to 1bpi P2-P2', RMSD: 0.73 Å) loop (1) in Figure 2c in a region not known to be cleaved during activation (Chen et al, 1998). A low resolution hit is also found in (2) another prominent surface loop in hemagglutinin (Gly142- Ser 145 to 1bpi P2-P2', RMSD: 1.39 Å).

Several hits were found in viral coat proteins, the core of which all adopt a β jelly roll structure. The hits found are mainly outside of this core structure, in elongated loop regions, and there is essentially no overlap in the location when one considers the superimposed family of protein structures (see Figure 2d), apart from an agreement between the loop in 1cov and 2plv, which share 55 % sequence identity. Of the 25 representative viral coat proteins considered (including those from bacteria (2), plants (11), insects (1) and mammals (12)) all 6 proteins containing are found in viruses that infect mammals. Interestingly the long loops that extend from the core β-jelly roll structure are largely absent from the viruses infecting other organisms.

There is good biological rationale as to why viruses could contain such loops. During host cell infection, viruses can come under attack from a variety of host cell defense mechanisms, many of which involve proteases. The presence of loops that might interfere with such attacks would be an advantage. Moreover, it is possible that viruses could use host cell proteases as shields that might protect them from recognition by other molecules, such as antibodies, involved in the immune system.

3.2 Toxins

Secreted toxins form another group of proteins that contain canonical loop like structures. Their presence in toxins of the Kunitz-type superfamily (e.g. alpha-Dendrotoxin, Dendrotoxin K, Dendrotoxin I) are well documented (Strydom, 1974) since they are closely related to the Kunitz-type protease inhibitors, and analysis shows they are weak protease inhibitors e.g. Dendrotoxin I binds to chymotrypsin with ~5% of BPTI activity (Strydom, 1974). We also found examples of prominently exposed canonical loop structures observed in structurally dissimilar toxins. For example, toxin II (1aho; see Figure 2e), alpha-toxin (1nea), SMK toxin (1kdv Chain A,B) Heat-labile toxin (1lts chain D) and delta-Endotoxin, C-terminal domain (1dlc), all of which are classified as different folds according to SCOP. To our knowledge, none of these toxins are known to inhibit proteases.

Like the viral proteins, toxins, whether from pathogens or used as defensive/predatory mechanisms by insects or reptiles, will be the target of host cell defense mechanisms. They could profit from containing means to protect themselves from attack by host proteases, or from proteins in the immune system. Alternatively, their role may be related to interactions with coagulant enzymes of the host. Venoms contain a variety of factors that affect the haemostatic mechanisms, possessing coagulatant, anticoagulent and haemorrhagic activity, as there is an advantage of the venom producer to spread the venom toxins throughout the body (Marsh, 1994).

3.3 Hydrolases

Canonical loop like structures are present in many secreted hydrolases. For example, the self inhibiting serine protease, HL collagenase from common cattle grub (Figure 2b). There are a number of other serine proteases that have surface exposed canonical loops. In particular those involved in the coagulation cascade, including thrombin (1tom chain L,H), Coagulation factor VIIa (1dan chain H) and Activated protein C (autoprothrombin IIa; 1aut chain C). Like HL collagenase all three proteins have at least one hit in a (different) loop at the edge of the active site, the arrangement of loops partially restrict access to the active site cleft. Four hits are found in Metalloproteases. There are two hits in different regions in the alpha+beta Zincin-like fold in the catalytic (N-terminal) domain of alkaline protease (1kap chain P, residues: 1-239) see Figure 2f and in Astacin (1ast). Although there is no evidence that the canonical loops bind to any of the known serine proteases, inhibitory function could be important in regulation of the serine proteases in the extracellular environment.

3.4 Cytokines

Several hits where found in small, secreted effector molecules, such as cytokines or hormones. Several of these occur on the surfaces of interleukins.

There are nine other hits (not shown in Table 2) that do not fit into the above categories, though still may represent real examples of protease inhibitor loops that serve a possible function that is not obviously apparent. Several are within extracellular receptor attached domains (e.g. fibronectins). It may be that these regions play a roll in avoiding proteolytic cleavage, or are involved in specific protein-protein interactions.

Acknowledgements

This work was supported by the Wellcome Trust in the form of a Career Development Fellowship to RMJ.

References

PJ Artymiuk et al. A graph-theoretic approach to the identification of three-dimensional patterns of amino acid side-chains in protein structures. *J Mol Biol* 243:327-344, 1994.

W Bode and R Huber. Natural protein proteinase inhibitors and their interaction with proteinases. *Eur J Biochem* 204:433-451, 1992.

J Chen et al. Structure of the Hemagglutinin Precursor Cleavage Site, a Determinate of Influenza Pathogenicity and the Origin of the Labile Conformation. *Cell* 95, 409-417, 1998.

AJ Doherty et al. The helix-hairpin-helix DNA-binding motif: a structural basis for non-sequence-specific recognition of DNA. *Nucleic Acids Res* 24:2488-2497, 1996.

SJ Hubbard and JM Thornton. 'NACCESS', Computer Program, Department of Biochemistry and Molecular Biology, University College London, 1993.

RM Jackson. Comparison of protein-protein interactions in serine protease-inhibitor and antibody-antigen complexes: implications for the protein docking problem. *Protein Sci* 8:603-613, 1999.

W Kabsch and C Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features *Biopolymers* 22:2577-2637, 1983.

M Laskowski and I Kato. Protein inhibitors of proteinases. *Annu Rev Biochem* 49:593-626, 1980.

NA Marsh. Snake venoms affecting the haemostatic mechanism - a consideration of their mechanisms, practical applications and biological significance. *Blood Coagul Fibrinolysis* 5:399-410, 1994.

AG Murzin et al. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol* 247:536-540, 1995.

CA Orengo et al. CATH - a hierarchic classification of protein domain structures , *Structure* 5:1093-1108, 1997.

CA Orengo et al. From protein structure to function. *Curr Opin Struct Biol* 9:374-382, 1999.

RB Russell et al. Supersites within superfolds. Binding site similarity in the absence of homology. *J. Mol. Biol.* 282:903-918, 1998.

RB Russell. Identification of protein three-dimensional side-chain patterns: New examples of convergent evolution. *J. Mol. Biol.* 279:1211-1227, 1998.

I Schechter and A Berger. On the size of the active site in proteases. *Biochem. Biophys. Res. Com.* 27:157-162, 1967.

DJ Strydom. Protease inhibitors as snake venom toxins. *Nat New Biol.* 243:88-9, 1973.

MB Swindells. Classification of doubly wound nucleotide binding topologies using automated loop searches. *Protein Sci.* 2:2146-2153, 1993.

Yang F et al., (1998) Crystal structure of Escherichia coli HdeA. *Nat Struct Biol* 1998 5, 763-764.

Consistent Integration of non-reliable heterogenous information resources applied to the annotation of transmembrane proteins

Steffen Möller* and Michael Schroeder†
*European Bioinformatics Institute, Cambridge, UK, moeller@ebi.ac.uk
†City University, London, UK, msch@soi.city.ac.uk

Abstract

Information agents integrate multiple distributed heterogeneous information sources. The challenging, yet unsolved, problem remains to ensure the semantic consistency of the integrated data. In this paper, we set out to develop a general approach to inconsistency management for information agents. It is implemented as part of the EDITtoTrEMBL system and applied on a large real-world problem in the domain of bioinformatics, the annotation of membrane spanning proteins.

Keywords: Information agents, Inconsistency Management, Knowledge Integration, Revise, EDITtoTrEMBL, Automated Protein Sequence Annotation.

1 Introduction

Information agents are computational software systems that have access to multiple, heterogeneous and geographically distributed information sources (Wiederhold and Genesereth (1996)). One of their main tasks is to perform active searches for relevant information in non-local domains on behalf of their users or other agents. Information from multiple autonomous sources is retrieved, analysed, manipulated and integrated to finally provide a high-level access to information that is otherwise not efficiently usable.

A common architecture for information agents consists of information providers, wrappers, facilitators, and mediators (Wiederhold and Genesereth (1996)). A wrapper is associated with each information provider to prepare retrieved data for the mediator. The mediator is the point of contact for a user (human or agent); it uses the facilitator to get in touch with the wrappers and knows what kind of information the wrappers can provide. Given a user query it will then contact the wrappers, integrate the results, and return them to the user.

Recently, much effort has been devoted to information agents resulting in systems such as SIMS (Arens et al. (1993, 1996)), UMDL (Durfee et al. (1997)), Infomaster (Genesereth et al. (1997)), InfoSleuth (Bayardo et al. (1997)) and Softbot (Etzioni and Weld (1994)).

This paper presents how during the process of integration potential inconsistencies can be both revealed and removed. These techniques are implemented in the agent system EDITtoTrEMBL (Möller et al. (1999)), for which the integration of data while preserving consistency is a special challenge due to the inherent uncertainty and incompleteness of provided data.

For this purpose domain knowledge is fomalized in extended logic programs (Alferes and Pereira (1996)), i.e. we model reliability of information and define potential inconsistencies. Revise (Damásio et al. (1997)) is introduced as a tool capable of determining facts responsible for inconsistencies and to make suggestions for a knowledge refinement. We discuss the kinds of inconsistency our method can detect and also its limits.

2 The Agent System EDITtoTrEMBL

Background The pharmaceutical industry and geneticists all over the world expect answers to many questions on the basis of the results of genomic sequencing projects and they already have answered many. This not only includes the human genome but also model organisms, e.g. the fruit fly and yeast, and a range of pathogens. The DNA sequences are semi-automatically searched potential genes and are subsequently submitted to a public database, e.g. the EMBL nucleotide database ((Stoesser et al. (1998))). No biochemical characterization is available at this stage. Since the number of potentially interesting genes is so large, the data must be annotated to allow an easier preselection of sequences for further studies.

This leads to the problem of sequence annotation. SWISS-PROT, a high-quality database for protein sequence data, is annotated manually by a team of professional annotators ((Bairoch and Apweiler (1999))). However, the ever increasing amount of data creates the need for new techniques to complement manual curation. To address this problem, the database TrEMBL (Translation of EMBL) was created as a supplement to SWISS-

PROT to store all coding sequences in EMBL that are not already integrated in SWISS-PROT. The concept of SWISS-PROT+TrEMBL allows the provision of a comprehensive protein sequence database without lowering the editorial standards of SWISS-PROT. Every entry in TrEMBL is enriched by automated annotation. This means that every TrEMBL entry is analysed by a set of programs and from their output new or improved annotation is derived.

At the EBI these arbitrary analysis programs are integrated into a distributed and highly flexible environment EDITtoTrEMBL(Environment for Distributed Information Transfer to TrEMBL), described in detail in (Möller et al. (1999)). Its purpose is to provide a correct, comprehensive and complete annotation of the sequence data available in the databases.

The main contribution of this paper is a method and an algorithm to maintain semantic consistency among the integrated predictions. Before going into the details of our approach we elaborate on the nature and origin of the integrated information.

Architecture The environment comprises two kinds of agents. One, the DISPATCHERs act as a combination of mediator and facilitator. The other, ANALYSERs, function as wrappers around the incorporated heterogenous data sources to provide a homogenous environment.

The ANALYSER's responsibilities are to ensure a consistent use of vocabulary to estimate the quality of the annotation it provides.

For the task presented, the annotation of transmembrane proteins, three databases PROSITE (Bairoch et al. (1997)), PRINTS (Attwood et al. (1998)), and PFAM (Bateman et al. (1999)) are accessed and the applications TMHMM (Sonnhammer et al. (1998)), HMMTOP, PHD (Rost et al. (1996)), TOPPRED (MG and von Heijne (1994)), MEMSAT (Jones et al. (1994)), DAS (?)), SOSUI (Hirokawa et al. (1998)), Eisenberg-analysis (Eisenberg et al. (1982)), and Hydropathy-analysis (Kyte and Doolittle (1982)) are wrapped. Three dispatchers are involved. The first controls the whole process, the second integrates the domain databases and the third integrates the prediction methods.

Figure 1 shows the system's tree structure. A subtree represents a problem domain. The entries are sent to a set of programs and the integration is performed by the respective dispatcher responsible for the problem domain.

Depending on the work load multiple instances of a specific dispatcher and eventually its tools can be created. This ensures the scalability of the approach.

Dispatchers may find the information provided by an analyser *inconsistent*. In this paper, we show how to enhance the dispatchers's capabilites by inconsistency management. We show how it can identify semantic inconsistencies among its annotations and how it can revise them appropriately.

To reduce complexity a dispatcher makes the assumption that entries sent to it are always consistent and hence only cares about inconsistency introduced by analysers under its control. Again this ensures scalability.

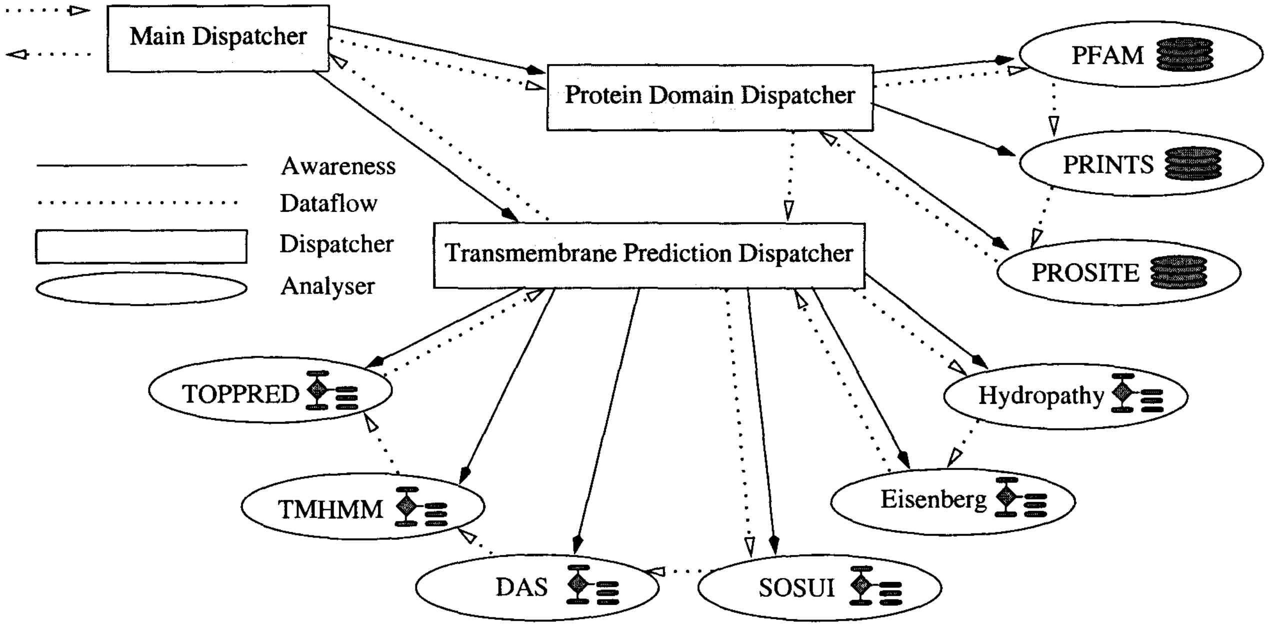


Figure 1: The EDITtoTrEMBL architecture. Dispatchers act as mediators, analysers as wrappers.

Sources of Information The databases incorporated are so called protein domain databases. The protein domain dispatcher retrieves matches to these external databases. The prediction of a domain becomes a rule for annotation by induction from the annotation that is associated in the protein database SWISS-PROT and by manual creation.

We focus on the annotation of transmembrane proteins by interpreting the results of a set of different programs for the prediction of membrane spanning regions. The programs make different premises for the prediction and differ in their quality for different protein families.

A match of a domain's pattern in a protein sequence is associated with a probability by which a random sequence might contain it. Similarly, other tools provide reliability factors which the analyser uses as a basis to determine the probability for the correctness of the information.

The basic assumption underlying the approach of an integration of multiple programs is that if a protein's features are equally determined by different methods then this prediction should more likely be correct.

The transmembrane dispatcher can not revise information that has been provided by protein domain databases. This information respective the proteins topology is usually incomplete but most reliable. It serves as a referee to resolve ambiguities and to avoid wrong annotation.

Sequential annotation of protein sequence data Figure 1 also shows how annotation is added incrementally. The DISPATCHER creates a summary of the results of individual agents. This is the moment when the DISPATCHER may find the provided information *inconsistent* and the techiques presented in this paper are applied.

If available, the information presented by protein domain databases is most valuable for a first characterization of proteins and should therefore be requested first. Dependencies between participating agents can be dynamically derived (Möller et al. (1999)) or otherwise declared.

Membrane Proteins: Biological Background This section presents the biological grounds helpful to understand the facts and rules described in section 4.

Membranes are boundaries of cells or their compartments. Certain proteins are integrated in a membrane and act as transporters or they transduce signals. They are very important for medical research and hence of high interest. For the undertstanding of a transmembrane protein's function it is helpful to determine its structure, especially to determine which moieties are buried within the membrane and which parts of the protein form loops on either side of the membrane.

A protein is represented as a linear chain of amino acids (see figure 2). When integrated into a membrane they can be visualized as boxes spanning the membrane connected by the protein chain.

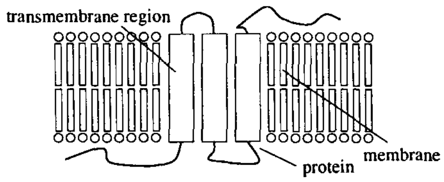


Figure 2: Schematic view of a transmembrane protein

Transmembrane segments for cytoplasmic or mitochondrial inner membranes are assumed helical and have a length between 16 and 25 residues. The majority of positive residues are on the inner side of the membrane (von Heijne (1986)). Several modifications are potentially done to individual residues of the amino-acid chain which depends on the compartment the protein is accessible to. The transmembrane annotation must be compatible with known post-translational modifications made to the protein.

Biochemically, the topology of transmembrane proteins can not be fully determined. Experiments testing the accessibility to proteases or antibodies still leave room for interpretation and so do fusion experiments with indicator proteins. Even the few crystal structures available do not give experimental evidence on how exactly the protein is embedded in the membrane. On the one hand this makes the evaluation of a prediction difficult, but on the other hand this also means that a bullet-proof annotation is not expected by biologists.

3 Representing Knowlege and Uncertain Beliefs

A consensus can be achieved in different ways. One might look for the annotation supported by the majority of tools (Cuff et al. (1998)), though with extra knowledge from domain databases the majority may be proven wrong. An integration can not be achieved without an interpretation of the data on a *semantic level*.

In the this section we introduce extended logic programming to represent the possibly inconsistent biological domain knowledge and to be able to revise the least reliable information.

Extended Logic Programming Well-founded semantics with exlicit negation (WFSX) provides a semantics for extended logic programs, i.e. logic programs, which are extended by a second kind of negation. This powerful language is appropriate for a spate of knowledge representation and reasoning forms (Alferes and Pereira (1996)). Formally, an extended logic program is defined as follows:

Definition 3.1 *An* extended logic program *is a (possibly infinite) set of rules of the form* $L_0 \leftarrow L_1, \ldots, L_m, not\, L_{m+1}, \ldots, not\, L_n$ *where each* L_i *is an* *objective literal* $(0 \leq i \leq n)$. *An objective literal is either an atom A or its explicit negation $\neg A$. Literals of the form not L are called default literals. Literals are either objective or default ones.*

Example 1 *Consider two predicates, domain and ft, for a feature table entry of the databases $SWISS-PROT$ or $TrEMBL$, for example. The ft predicate contains the start and end position of a given region such as $transmembrane$. The derived domain predicate states that all positions between these two boundaries are transmembrane. We can capture this relation by the rule below:*

$$domain(Agent, Pos, transmem) \leftarrow \\ ft(Agent, transmem, Pos1, Pos2), Pos1 \leq Pos, Pos \leq Pos2.$$

Besides facts and rules, we can specifiy integrity constraints.

Definition 3.2 *An* integrity constraint *has the form* $\bot \leftarrow L_1, \ldots, L_m, not\, L_{m+1}, \ldots, not\ L_n$ *with* $0 \leq m \leq n$ *where each* L_i *with* $0 \leq i \leq n$ *is an objective literal, and* $\bot$ *stands for false.*

Syntactically, the only difference between the program rules and the integrity constraints is the head. A rule's head is an objective literal, whereas the constraint's head is $\bot$, the symbol for false. Semantically the difference is that program rules open the solution space, whereas constraints limit it.

Example 2 *The constraint below states that transmembrane regions have to be longer than 16.*

$$\bot \leftarrow \quad ft(Agent, Acc, transmem, Pos1, Pos2), \\ X\ is\ Pos2 - Pos1, X \leq 15.$$

When defining integrity constraints our first objective is to detect violations, the next step is to remove the violations. Since by definition it is not possible to change a fact, we introduce revisables. Revisables are assumptions we are willing to change if inconsistencies arise.

Definition 3.3 *The revisables R of a program P are a subset of the (possibly default negated) literals which do not occur as rule heads in P.*

Example 3 *Predictions of transmembrane regions are formalized in the feature table. For our application we must not take them for granted and hence they are defined as revisables rather than facts. By default we set the entries to true, but should inconsistencies arise, we are willing to withdraw them, i.e. set them to false.*

$$revisable(\quad ft(tmhmm, p12345, transmem, 6, 26), \quad true).$$

Similarly, it is possible to revise assumptions from false to true.

For many cases it is useful to specify how easily a revisable can be changed or, in other words, how reliable an assumption is.

Example 4 *The probabilities below state that tmhmm's assumption about first transmembrane region is not very reliable (0.5), whereas its assumption about the second region is (0.1).*

$$probability(\quad ft(tmhmm, p12345, transmem, 6, 26), \quad 0.5). \\ probability(\quad ft(tmhmm, p12345, transmem, 27, 50), \quad 0.1).$$

Probabilities can also be used to rank competing ft entries generated by a single analyser. This is most useful for a wrapper of a neural network. It would represent the most active neuron with a higher proabillity than the second most active one.

To summarise, we model knowledge by facts, rules, and integrity constraints and beliefs of the agents by revisables. The certainty of the beliefs may be qualified by a probability indicating the degree of reliability.

Revising Inconsistent Domain Knowledge and Agent Beliefs Our objective is to detect violations of the integrity constraints and to revise the assumptions involved as little as possible to repair them. Formally, such as revision is defined as follows:

Definition 3.4 *Let P be a program and R a set of revisables. The set* $R' \subseteq \{L \mid not\ L \in R\} \cup \{\neg L \mid L \in R\}$ *is called a* revision *if it is a minimal set such that* $P \cup R'$ *is free of contradictiction, i.e.* $P \cup R' \not\models_{WFSX} \bot$[1]

Before we show how the revisions are computed, we need some defintions. Conflicts are sets of revisables that lead to a contradiction.

Definition 3.5 *Let P be an extended logic program with revisables R. Then* $R_\bot \subset R$ *is a* conflict *iff* $P \cup R_\bot \models \bot$.

To compute revisions, we have to change revisables so that all conflicts are covered. Such a cover is called a *hitting set*, since all conflicts involved are hit.

Definition 3.6 *A* hitting set *for a collection of sets C is a set* $H \subseteq \bigcup_{S \in C} S$ *such that* $H \cap S \neq \emptyset$ *for each* $S \in C$. *A hitting set is* minimal *iff no proper subset of it is a hitting set for C.*

Theorem 3.7 *Let P be a program. Then R is a revision of P iff R is a minimal hitting set for the collection of conflicts for P.*

Theorem 3.7 states that revisions can be computed from conflicts and hitting sets which can be obtained from hitting set trees (Reiter (1987)):

Definition 3.8 *Let C be a collection of sets. An HS-tree for C, call it T, is a smallest edge-labeled and node-labeled tree with the following properties:*

1. The root is labeled $\surd$ if C is empty. Otherwise the root is labeled by an arbitrary set of C.

2. For each node n of T, let $H(n)$ be the set of edge labels on the path in$T$ from the root node to n. The label for n is any set $\Sigma \in C$ such that $\Sigma \cap H(n) = \emptyset$, if such a set Σ exists. Otherwise, the label for n is $\surd$.

[1] For details on the definition of the inference operator $\models_{WFSX}$ see e.g. (Alferes and Pereira (1996)).

3. If n is labeled by the set Σ, then for each $\sigma \in \Sigma$, n has a successor n_σ joined to n by an edge labeled by σ.

We informally explain the algorithm (proposed in (Reiter (1987)) and corrected in (Greiner et al. (1989))) with its adaption to extended logic programs.

To compute conflicts, the Revise engine uses SLXA, a proof-procedure, which returns the revisables involved in the proof. It is based on the SLX proof procedure for WFSX [1].

The calls to SLXA are driven by the Revise engine. Its main data structure is the hitting-set tree. The construction of the hitting-set tree is started on candidate $\emptyset$, meaning that the revisables initially have their default value.

We say that the node $\emptyset$ has been expanded when the SLXA procedure is called to determine one conflict. If there is none, then the program is non-contradictory and the revision process is finished. Otherwise, the Revise engine computes all the minimal ways of satisfying the conflicted integrity constraint returned by SLXA, i.e. the sets of revisables which have to be added to program in order to remove that particular conflict.

For each of these sets of revisables, a child node of $\emptyset$ is created. If there is no way to satisfy the conflicted integrity then the program is contradictory. Otherwise the Revise engine selects a node to expand according to some preference criterium and cycles: it determines a new conflict, it expands that node with the revisables which remove the conflict. This continues until there is no further conflict remaining and hence a solution is found.

The solution is kept in a table for pruning the revision tree by removing any nodes which contain some solution, and have been selected according to the preference criterium.

The order in which the nodes of the revision tree are expanded is important to obtain minimal solutions first. In the current implementation we cater for minimality by set-inclusion, cardinality and probability (Damásio et al. (1997)).

4 Representation of the Biological Knowledge and the Agent Beliefs

This section presents biological background for a selection of conflicts and their formal representation in Revise. A transmembrane prediction is presented as a set of facts. The numbers denote the respective start and end of a specific region of sequence described as a feature.

A fact's first argument is the source of a information. The second argument is an identifier for the sequence, the accession number.

$$ft(swissprot, p17353, transmem, 31, 50).$$

Besides the localisation of transmembrane regions it is important in what direction the protein is integrated into the membrane. This is denoted by the predicate $topology(Source, Accession, Domain1, Domain2)$. It describes the direction of the first transmembrane helix. Post-translational modifications are subject to individual residues of a peptide sequence only. The two positions will hence be identical. The only exception to this are disulfid bridges which connect two residues.

$$\begin{array}{l} ft(mod\_res, 5, 5, phosphatation).\\ ft(carbohyd, 10, 10).\\ ft(disulfid, 66, 99).\end{array}$$

A disulfide bridge links to redidues within the same compartment only:

$$\begin{array}{ll} \leftarrow & ft(Agent, Acc, disulfid, Pos1, Pos2),\\ & in\_or\_out(Agent, Acc, Pos1, D1),\\ & in\_or\_out(Agent, Acc, Pos2, D2),\\ & D1 \neq D2.\end{array}$$

A conflict also occurs if a disulfide bridge is established in the intracellular domain:

$$\begin{array}{ll} \leftarrow & ft(Agent, Acc, disulfid, Pos1, \_P),\\ & in\_or\_out(Agent, Acc, Pos1, inner).\\ \leftarrow & ft(Agent, Acc, disulfid, \_P, Pos2),\\ & in\_or\_out(Agent, Acc, Pos2, inner).\end{array}$$

Glycosylation is established in the outer domain only:

$$\begin{array}{ll} \leftarrow & ft(Agent, Acc, carbohyd, Pos, Pos),\\ & in\_or\_out(Agent, Acc, Pos, D),\\ & D \neq outer.\end{array}$$

It must be checked that the other modifications are made to residues of the inner compartment

$$\begin{array}{ll} \leftarrow & ft(Agent, Acc, Modification, Pos, Pos, \_),\\ & member(Modification, [lipid, mod\_res]),\\ & in\_or\_out(Agent, Acc, Pos, D),\\ & D \neq inner.\end{array}$$

Knowledge specific for transmembrane proteins The rules presented before only looked at individual revisables and their consistence with knowledge independent from the transmembrane prediction process. The following rules compare transmembrane predictions with each other.

All methods must agree on a protein being transmembrane at all.

$$\begin{array}{ll} \leftarrow & ft(Agent, Acc, transmem,_F rom,_T o),\\ & not \quad is\_transmembrane(Agent2, Acc).\end{array}$$

If two transmembrane regions are predicted to overlap then neither border should differ more than four residues from the other predictions border.

$$\begin{array}{ll} \leftarrow & ft(Agent1, Acc, transmem, From1, To1),\\ & ft(Agent2, Acc, transmem, From2, To2),\\ & (From1 > From2, From1 < To2; To1 > From2, To1 < To2),\\ & (abs(From1 - From2) > 4; abs(To1 - To2) > 4).\end{array}$$

The length of a transmembrane region ist limited:

$$\begin{array}{ll} \leftarrow & ft(\_Origin, \_AccessionNumber, transmem, From, To),\\ & X\ is\ To - From, X \leq 15.\\ \leftarrow & ft(\_Origin, \_AccessionNumber, transmem, From, To),\\ & X\ is\ To - From, X > 25.\end{array}$$

Futher heuristics, like the positive-inside rule (von Heijne (1986)) have been implemented and can be used for the support or refusal of a prediction.

$rev(ft(das, p04633, tm, 19, 29), true).$
$rev(ft(das, p04633, tm, 120, 128), true).$
$rev(ft(das, p04633, tm, 214, 229), true).$
$rev(ft(das, p04633, tm, 216, 227), true).$
$rev(ft(das, p04633, tm, 280, 285), true).$

$rev(top(phd, p04633, inner, outer), true).$
$rev(ft(phd, p04633, tm, 18, 35), true).$
$rev(ft(phd, p04633, tm, 117, 133), true).$
$rev(ft(phd, p04633, tm, 214, 231), true).$
$rev(ft(phd, p04633, tm, 271, 288), true).$
$rev(tm(tmhmm, p04633), false).$

$rev(ft(toppred, p04633, tm, 13, 33), true).$
$rev(ft(toppred, p04633, tm, 113, 133), true).$
$rev(ft(toppred, p04633, tm, 212, 232), true).$
$rev(ft(toppred, p04633, tm, 239, 259), true).$
$rev(ft(toppred, p04633, tm, 269, 289), true).$

Figure 3: Revisables presented to Revise. *transmembrane* is abbreviated as *tm*, *topology* as *top*, *revisable* as *rev*.

Matches with domain databases and derived knowledge Stating that an protein sequence sequences matches a domain specified in a protein domain database is stated with the predicate *matches* (see 5):

$$matches(ProtAccession, DomainAccession, From, To).$$

For the current implementation we make use of a manual translation of information stored in PROSITEDOC. This presents a biochemical interpretation of domains represented in the database PROSITE.

The rules gathered from domain databases are valid independently from the actual sequence since they all require the sequence to match a specific pattern before any further information can be deduced.

A sequence's match can be dynamically determined. For better efficiency, only those rules are presented to revise that have a chance to fire.

5 Application

This section gives an example how Revise works. It is provided with the set of revisables shown in figure 3 and the rules as previously described in 4.

The predicate *solution* returns a list of minimal revisions, consisting of each a set of revisables changed from false to true and a list of those revisables changed from true to false.

Local conflict checks only If neither the domain information is present nor the domain database has any rules available, $solution(X)$ will return a single solution:

X = [[], [*ft*(*das*, p04633, *transmem*, 19, 29), *ft*(*das*, p04633, *transmem*, 120, 128), *ft*(*das*, p04633, *transmem*, 216, 227), *ft*(*das*, p04633, *transmem*, 280, 285)]] ;

This represents the transmembrane regions that are too short.

Balance with other predictions With the additional constraint that all predictions must agree on wether a protein is integrated into the membrane or soluble a revision of TMHMM's prediction from true to false is introduced. The third solution trusts TMHMM and assumes all transmembrane regions to be false positive. This means the revision of all the facts revisables.

X = [[*is_transmembrane*(*tmhmm*, p04633)], [*ft*(*das*, p04633, *transmem*, 19,..., 285), *ft*(*phd*, p04633, *transmem*, 18, 35)]] ;

X = [[*is_transmembrane*(*tmhmm*, p04633)], [*ft*(*das*, p04633, *transmem*, 19,..., 285), *ft*(*toppred*, p04633, *transmem*, 13, 33)]] ;

X = [[], [*ft*(*das*, p04633, *transmem*, 19,..., 285), *ft*(*phd*, p04633, *transmem*, 18, 35), *ft*(*phd*, p04633, *transmem*, 117, 133), *ft*(*phd*, p04633, *transmem*, 214, 231), *ft*(*phd*, p04633, *transmem*, 271, 288), *ft*(*toppred*, p04633, *transmem*, 13, 33), *ft*(*toppred*, p04633, *transmem*, 113, 133), *ft*(*toppred*, p04633, *transmem*, 212, 232), *ft*(*toppred*, p04633, *transmem*, 239, 259), *ft*(*toppred*, p04633, *transmem*, 269, 289)] ;

Use of protein domain database The following matches to the PROSITE database have been derived:

$matches(p04633, ps00215, prosite, 32, 41).$
$matches(p04633, ps00215, prosite, 132, 141).$
$matches(p04633, ps00215, prosite, 231, 240).$

Additional information could be retrieved from matches to the PROSITE database:

is_transmembrane(prositedoc,Acc)←
    $matches(Acc, ps00215, prosite, \_, \_).$
num_tm_regions(prositedoc,Acc,6)←
    $matches(Acc, ps00215, prosite, \_, \_).$
loop(prositedoc,Acc,'',X,T)←
    $matches(Acc, ps00215, prosite, F, T),\ X\ is\ F + 3).$
transmembrane(prositedoc,Acc,X,X,)←
    $matches(Acc, ps00215, prosite, F, T),\ X\ is\ F - 3).$

The knowledge that this protein sequence indeed belongs to a transmembrane protein led to the exclusion of the previously third option. The previously second solution needed to be removed since PHD's transmembrane region from 18 to 35 is in conflict with a loop region between residues 35 and 41.

X = [[], [*ft*(*das*, p04633, *transmem*, 19,...285), *ft*(*phd*, p04633, *transmem*, 18, 35), *ft*(*toppred*, p04633, *transmem*, 239, 259)]] ;

Interpretation of Revise's output The constraints set to the system guarantee that this final solution has the property not to be in conflict with information known for specific protein domains. When multiple tools predict the same transmembrane region then they vary only slighly in their description.

Revise presents all possible interpretations of prediction methods consistent with itself and extra knowledge from protein domain databases. This can be visualized as follows:

In figure 4 the boxes represent membrane spanning regions. Crossed out is a predicted region Revise found to be in conflict with other information.

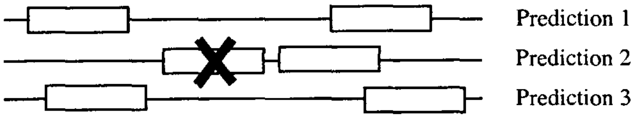


Figure 4: Visualisation of conflict resolution

For the annotation that goes into TrEMBL the median of the transmembrane regions's borders are choosen respectively. If ambiguities remain, ie. multiple solutions, only the common annotation goes into TrEMBL being appropriately marked.

Mutual independence of revisables The revision of a fact may give rise to new conflicts. This inherent nonmonotonicity is the major difficulty in conflict resolution. It is not directly possible to give an advice to the system which revisions should be attempted in case of the need for a specific revision due to a specific conflict.

This feature could also be used to define semantic dependencies among revisables. If these exist, those must be defined in Revise as conflicts. There are two problems with this approach. The obvious one is that these rules may be hard to maintain. But also if there is a semantic dependence the probability of a change may be different if a change was not triggered directly but by a secondary constraint.

For this reason the transmembrane annotation was represented by the transmembrane regions as a collection of predicates plus an additional one for its topology. The alternative was an additional description of the individual loops, leading to an increased efficieny for rules, though thereby loosing the revisables's mutual independece.

6 Conclusion and Future Work

In this paper, we have demonstrated that the integration of heterogenous data sources can have a symbiotic effect on the overall quality of the information provided. For the automated annotation of protein sequences this is absolutely vital and we are very confident that similar approaches will be implemented for other domains in the future.

We have demonstrated how extended logic programming and program revision can be used to represent domain knowledge and agent beliefs in distributed information agent systems. In particular, we have shown how to deal with different degrees of reliability and how to remove inconsistencies using various minimality options.

While a revision is ideal for binary statements, it is not practical to use Revise to allow a fact's refinement. This means the adaption of a fact instead of its removal. The possibility to allow refinements may have been beneficial for our application for which a domain's boundaries could have been changed to fulfill a constraint. This will be addressed in our future work.

It may be at value to note that although the process of a revision leads to a centralization of processing this does not represent a bottleneck. Any agent in the system can be individually cloned and thereby duplicate the bandwith.

7 Acknowledgements

We'd like to thank Gill Frazier and Caleb Webber for reviewing this document and Carlos Damasio, Wolfgang Fleischmann and Rolf Apweiler for many fruitful discussions.

References

J. J. Alferes and L. M. Pereira. *Reasoning with Logic Programming.* (LNAI 1111), Springer-Verlag, 1996.

Y. Arens, , C.-N. Hsu, and C. A. Knoblock. Query processing in the SIMS information mediator. In *Proceedings of the ARPA/Rome Laboratory Knowledge-based Planning and Scheduling Initiative Workshop*, 1996. Reprinted in Readings in Agents. Huhns, Singh (eds.), Morgan Kaufmann.

Y. Arens, C. Y. Chee, C.-N. Hsu, and C. A. Knoblock. Retrieving and integrating data from multiple information sources. *International Journal on Intelligent and Cooperative Information Systems*, pages 127–158, 1993.

Terry K. Attwood, M. E. Beck, D. R. Flower, Philip Scordis, and J. Selley. The PRINTS protein fingerprint database in its fifth year. *Nucl. Acids Res.*, 26(1):304–308, 1998.

A. Bairoch, P. Bucher, and K. Hofmann. The PROSITE database, its status in 1997. *Nucl. Acids Res.*, 25(1):217–221, 1997.

Amos Bairoch and Rolf Apweiler. The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999. *Nucl. Acids Res.*, 27: 49–54, 1999.

Alex Bateman, Ewan Birney, Richard Durbin, Sean R. Eddy, R. D. Finn, and Erik L. L. Sonnhammer. Pfam 3.1: 1313 multiple alignments match the majority of proteins. *Nucl. Acids Res.*, 27:260–262, 1999.

R. J. Bayardo, W. Bohrer, R. Brice, A. Cichoki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. Martin, M. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. Infosleuth: Agent-based semantic integration of information in open and dynamic environments. In *ACM SIGMOD*, pages 195–206, 1997.

James A. Cuff, Michelle E. Clamp, A. S. Siddiqui, M. Finlay, and Geoff J. Barton. Jpred: a consensus secondary structure prediction server. *Bioinformatics*, 14(10):892–3, 1998.

Carlos Viegas Damásio, Luis Moniz Pereira, and Michael Schroeder. REVISE: Logic programming and diagnosis. In *Proceedings of the Conference on Logic Programming and Non-monotonic Reasoning LPNMR97.* LNAI 1265, Springer–Verlag, 1997.

E. Durfee, D. L. Kiskis, and W. P. Birmingham. The agent architecture of the university of michigan digital library. *IEEE Proceedings, Software Engineering*, 144(1):61–71, 1997.

D. Eisenberg, R. M. Weiss, and T. C. Terwilliger. The helical hydrophobic moment: a measure of the amphiphilicity of a helix. *Nature*, 299 (5881):371–374, 1982.

Oren Etzioni and Daniel Weld. A softbot-based interface to the internet. *Communications of the ACM*, pages 72–76, 1994.

M. R. Genesereth, Arthur M. Keller, and Oliver Dischka. Infomaster: An information integration systsm. In *ACM SIGMOD*, 1997.

Russell Greiner, Barbara A. Smith, and Ralph W. Wilkerson. A correction of the algorithm in reiter's theory of diagnosis. *Artificial Intelligence*, 41(1):79–88, 1989.

T. Hirokawa, S. Boon-Chieng, and S. Mitaku. Sosui: classification and secondary structure prediction system for membrane proteins. *Bioinformatics*, 14(4):378–379, 1998.

D. T. Jones, W. R. Taylor, and J. M. Thornton. A model recognition approach to the prediction of all-helical membrane protein structure and topology. *Biochemistry*, 33(10):3038–3049, 1994.

J. Kyte and R. F. Doolittle. A simple method for displaying the hydropathic character of a protein. *J. Mol. Biol.*, 157(1):105–32, 1982.

M. G. Claros MG and G. von Heijne. Toppred ii: an improved software for membrane protein structure predictions. *Comput Appl Biosci.*, 10 (6):685–686, 1994.

Steffen Möller, Ulf Leser, Wolfgang Fleischmann, and Rolf Apweiler. EDITtoTrEMBL: A distributed approach to high-quality automated protein sequence annotation. *Bioinformatics*, 15(3):219–227, 1999.

Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–96, 1987.

Burkhard Rost, Rita Casadio, and Piero Fariselli. Refining neural network predictions for helical transmembraane proteins by dynamic programming. In D States et al., editor, *The fourth international conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 192–200, St. Louis, U.S.A., 1996. AAAI Press.

Erik L. L. Sonnhammer, Gunnar von Heijne, and Anders Krogh. A hidden markov model for predicting transmembrane helices in protein sequences. In *Proc. of Sixth Int. Conf. on Intelligent Systems for Molecular Biology*, pages 122–130, Menlo Park, CA, 1998. AAAI Press. http://genome.cbs.dtu.dk/krogh/TMHMM/index.html.

Guenther Stoesser, Mary Ann Moseley, Joanne Sleep, Michael McGowran, Maria Garcia-Pastor, and Peter Sterk. The EMBL nucleotide sequence database. *Nucl. Acids Res.*, 26(1):8–15, 1998.

Gunnar von Heijne. *EMBO J.*, (5):3021–3027, 1986.

Gio Wiederhold and Michael Genesereth. The basis for mediation. Technical report, Standford University, 1996.

Application of a time–delay neural network to the annotation of the *Drosophila melanogaster* genome

Martin G. Reese
Berkeley Drosophila Genome Project,
Department of Molecular and Cell Biology,
University of California, Berkeley,
California 94720–3200
mgreese@lbl.gov

Abstract

Computational methods for automated genome annotation are critical to understanding and interpreting the bewildering mass of genomic sequence data presently being generated and released. A neural network model of the structural and compositional properties of an eukaryotic core promoter region has been developed and its application for analysis of the *Drosophila melanogaster* genome is presented. The model uses a time–delay architecture a special case of a feed–forward neural network. The structure of this model allows for variable spacing between functional binding sites, which is known to play a key role in the transcription initiation process. Application of this model to a test set of core promoters not only gave better discrimination of potential promoter sites than previous statistical or neural network models, but also revealed indirectly subtle properties of the transcription initiation signal. When tested in the *Adh* region of 2.9 Mbases of the Drosophila genome, the NNPP program that incorporates the time–delay neural network model gives a recognition rate of 75 percent (69/92) with a false positive rate of 1/547 bases. The presented work can be regarded as one of the first intensive studies that applies novel gene regulation technologies for the identification of the complex gene regulation sites in the genome of *Drosophila melanogaster.*

1 Introduction

Recent advances in sequencing technology are making the generation of whole genome sequences commonplace. Capillary sequencers speed the production of raw data. Changing tactics from traditional mapping and sequencing clones in series to an integrated simultaneous mapping and sequencing approach (whole genome shotgun) has significantly reduced the amount of time it takes to completely sequence a genome. These improvements in genomic sequencing are possible because of software advances that fully exploit mapped clone constraint data and directly attack the problems that repetitive sequences cause during sequence assembly.

At present several very large–scale genomic sequencing projects are complete or are expected to complete within a few months. These initial genome sequences are from key model organisms in genetics and include five eukaryotes, *Saccharomyces cerevisiae*, *Schizosaccharomyces pombe*, *Caenorhabditis elegans*, *Drosophila melanogaster* and *Arabidopsis thaliana*, as well as draft human sequence. In a few years sequencing new genomes and individuals will become routine practice. This raw data is not immediately useful and interpreting it places major demands on the field of computational biology.

The development and application of a novel neural network system to recognize eukaryotic polymerase II promoters in the annotation of the *Drosophila melanogaster* genome is presented. A time–delay neural network (TDNN) is developed, an architecture that was originally introduced in speech recognition (Lang & Waibel, 1990; Waibel *et al.*, 1989), to model the complex sequence structure of a transcription start site. The transcription start site (TSS) is the location upstream of a gene where the polymerase II protein binds to the genomic DNA and initiates the transcription process. The entire region around the transcription start site is called a promoter.

A typical polymerase II promoter consists of multiple functional binding sites that are involved in the transcription initiation process. I trained separate neural networks for these individual binding sites (TATA box and initiator (Inr)) and integrate these separate networks into a time–delay neural network. The architecture of a time–delay neural networks is chosen because it is well suited to model this complex sequence structure because it allows for variable spacing between functional sites (equivalent to different time points in speech recognition), a feature common to polymerase II promoters.

These promoters have a very complex structure (for reviews see (Kornberg, 1999; Pugh, 1996; Pugh &

Tjian, 1992; Yokomori *et al.*, 1998)) consisting of these multiple DNA binding sites for transcription factors. Some of these sites enhance transcription and some other repress transcription. The nucleotide pattern of the sites is often related to the strength of binding. In addition to these core promoter elements in the vicinity of the transcription start site there exist long–range interactions through so called enhancer sites. Therefore, current methods to model these promoters are pruned for a high rate of false positives and the task of promoter recognition can be seen as one of the most difficult in the field of DNA sequence analysis.

2 Methods

2.1 Time–delay neural networks

For promoter modeling, a special neural network is chosen, the time–delay neural network (TDNN) architecture developed by Waibel *et al.* (1989). This architecture was originally designed for processing speech sequence patterns in time series with local time shifts. The usual way of transforming sequence patterns into input activity patterns is the extraction of a subsequence using a fixed window. This window is shifted over all positions of the sequence and the subsequences are translated into input activities. The network produces an output activity or score for each input subsequence.

The following two promoter specific features have to be learned:

1. The network has to recognize subsequences that may occur at non-fixed positions in the input window. Therefore the network has to learn that the subsequence is a feature independent of shifts in its position.
2. The network has to recognize features even when those features appear at different relative positions. This situation arises in cases where different subsequences occur in the input window with different relative distances. This happens very frequently in genomic sequences when one or more elements (nucleotides) are inserted or deleted in a given promoter.

The TDNN architecture addresses these problems by imposing certain restrictions on the network topology and by the way in which weights are updated. Hidden units are connected to a limited number of input units that represent a consecutive pattern in the input window. These hidden units have a *receptive* field, that is, they are only sensitive to a part of the input window. The important restriction is that the same *receptive field* has to be present at each position in the input exactly once. If the input window contains, for example, ten positions and a *receptive field* covers a subsequence of three positions, there must be eight hidden units with the same *receptive field.* Since the corresponding weights in all copies of a *receptive field* are forced to have the same values, these hidden units are said to have *linked receptive fields.* In neural network terminology this is also known as *weight sharing.* Each hidden unit is called a *feature unit* because it will recognize a certain feature in the input window irrespective of its relative position. During learning, the partial derivatives of corresponding weights in *linked receptive fields* are calculated separately since these hidden units with their *receptive fields* at different positions in the input window get different activation. To adapt a *receptive field,* the weight update is averaged over all copies of a weight. This average update is then applied to all copies of that weight. In this way, it is ensured that the copies of a *receptive field* remain identical for a given feature. In the basic TDNN architecture the hidden layers (feature units) are connected to the output layer in a standard feed–forward way. Training is performed using a modified backpropagation algorithm.

There are several successful applications of TDNNs in speech recognition (Waibel et al., 1989) and the recognition of handwritten characters (Lang & Waibel, 1990). These references include a detailed description of the time–delay architecture.

2.2 Implementation of the core–promoter time–delay neural network model (NNPP)

Using the time–delay architecture described above, two distinct neural networks, one for the TATA box and one for the Inr, were trained. I selected an input window of 30 bp (−40 to −10) for the TATA box neural network and a window of 25 bp (−14 to +11) for the Inr network. The window sizes were selected so that the consensus sequences for both binding sites are included. The two signals occur at varying distances relative to the TSS.

The two time–delay neural networks were trained independently. It was experimentally determined that a receptive field size of 15 bp performed the best. For the TATA network, this leads to a total of 120 input units (30 bp) and 60 weights (4 x 15) for each unit in the hidden layer. The Inr network has 100 input units (25 bp) and also 60 weights (4 x 15) for each unit in the hidden layer.

The weights of the receptive fields for both of the two networks were initialized using the weight matrices from the literature to "push" them to recognize particular signals. The TATA box weight matrix was taken from Bucher (1990), and the Inr weight matrix from Penotti (1990). These initializations were ideal to train the TDNNs to recognize the appropriate signal in the sequence (i.e. the TATA box time–delay network was forced to train only on the TATA box pattern at

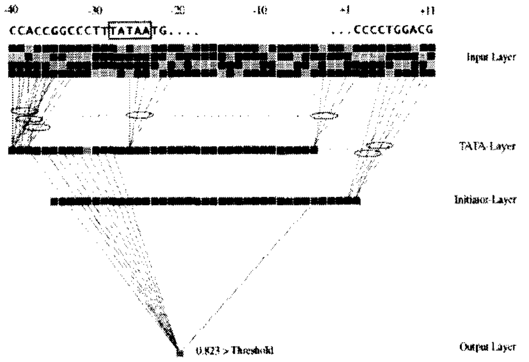


Figure 1: The trained two–layer time–delay neural network. The small squared boxes symbolize the neurons. The input layer is on top with the window reading in the DNA sequence. The receptive fields indicated with a circle grouping connections from the input layer to the two hidden layers (TATA and Inr) show the structure of the time–delay connections. Both hidden layers connect to the single output neuron on the bottom. For clarity, only strong weights are shown. For example, the only significant weights shown from the TATA–layer to the output unit are the ones that localize the position of the TATA box at the beginning of the input window (below CCACCGG). The TATA box is boxed. This test sequence of CCACC....GGACG received a score of 0.823 from NNPP.

approximately –20 bp). The results of both networks can be seen in Table 1 and are discussed below.

2.2 Incorporation of feature detector networks into the final TDNN

To combine the individual feature detector neural networks for TATA and Inr, we use a two–layer time–delay neural network. The input to the final TDNN consists of 51 bp, spanning the transcription start site from position –40 to +11 and including the TATA box and the Inr. The hidden layers from the two previously trained single–feature time–delay neural networks are copied into the combined TDNN and training is carried out. The resulting neural network maps high order correlation between the different features and their relative distance into a complex weight matrix. A snapshot of the two–layer (TATA and initiator) trained TDNN is shown in Figure 1. The weights from the hidden layers can be interpreted as the preferred position for an individual element in the input window.

All neural networks were integrated and tested using the Stuttgart Neural Network Simulator Software toolkit (Zell & al., 1999). The networks were then implemented in the Neural Network for Promoter Prediction (NNPP) program. This program is publicly accessible through a World Wide Web server (http://www.fruitfly.org/seq_tools/promoter.html).

3 Results

3.1 Application of NNPP to a cross–validated set of promoters

Table 1 shows the prediction results for the two single feature time–delay neural networks, the TATA box feature detector (column 2), the Inr feature detector (column 3) and the two–layer TDNN, which incorporates both (column 4 and 5). The results are averaged over four cross–validated test sets produced from the complete dataset of 429 promoters. The correlation coefficient is calculated as defined originally by Matthews (1975) and later adapted to the problem of gene finding evaluation by Burset and Guigó (1996) as:

$$CC = \frac{(TP \times TN) - (FN \times FP)}{\sqrt{(TP + FN) \times (TN + FP) \times (TP + FP) \times (TN + FN)}}$$

As can be seen from Table 1, the performance of the feature detecting networks used in isolation is rather poor. The TATA box network has the better performance of the two, since over 60% of the vertebrate promoters contain a TATA box. The predictive power of the initiator network is weaker because there is no real consensus sequence for vertebrate Inrs. The TATA box network recognizes on average 64 (60%) of the 107 promoter sequences in each test set (4–fold cross–validated) with an average of 38 (3.8%) false positive predictions. If we adjust the threshold so that on average 75 (70%) of the promoters are predicted correctly, there are 72 (7.2%) false positive predictions. The Inr neural network can only detect 11 (10%) of the promoters, with a false positive rate of 0.8%. The combination of both neural networks increases the prediction rate. If on average in the 4 cross–validated sets 54 (50%) promoters are correctly predicted, the false positive rate drops down to 1.0% (ten coding DNA regions predicted as promoters; correlation coefficient of 0.65), but that is similar to the TAT A–only results. Even if 75 (70%) promoters are correctly predicted, the average number of false predictions is only 53 (versus 72 for TATA alone). At a threshold of 0.12, 80% of the promoters predicted, the number of false positive predictions goes up to 125 (12.5%). 21 (19.6%) promoter sites on average in the test sets cannot be predicted at all using this 2–layer neural network.

For comparison, the results for a standard feed–forward backpropagation neural network with one hidden layer trained on the same data sets are shown

% Promoters recognized	TATA box FP-rate (CC)	Initiator FP-rate (CC)	Combined 2-layer TDNN (CC)	Threshold (0-1) for combined TDNN	Multi-layer Perceptron FP-rate (CC)
10	0.2% (0.36)	0.8% (0.28)	0.0% (0.38)	0.99	0.2% (0.35)
20	0.3% (0.45)	2.7% (0.27)	0.1% (0.38)	0.97	0.3% (0.45)
30	0.5% (0.52)	7.0% (0.28)	0.3% (0.50)	0.92	0.8% (0.48)
40	0.9% (0.56)	10.6% (0.26)	0.4% (0.60)	0.85	1.9% (0.50)
50	1.3% (0.62)	18.7% (0.25)	1.0% (0.65)	0.70	3.7% (0.51)
60	3.8% (0.60)	33.0% (0.21)	3.1% (0.61)	0.38	9.9% (0.44)
70	7.2% (0.57)	45.5% (0.18)	5.3% (0.58)	0.20	16.1% (0.40)
80	22.3% (0.39)	60.5% (0.17)	12.5% (0.52)	0.12	45.5% (0.23)

Table 1: NNPP Prediction performance on the 4–fold cross–validated data set. False positive rates and correlation coefficients are averaged over the 4–cross validated sets.

in the last column of Table 1. The number of hidden units and the number of training cycles were optimized exactly the same way as for the time–delay neural network. The results show the superiority of the two–layer TDNN. At a threshold that gives 64 (60%) correct predictions, the number of false positive predictions is more than three times higher for the standard network (99 false predictions) than for the 2–layer TDNN (31 false predictions). This shows that reducing the parameter space from 3,091 adjustable weights in the standard network to 169 in the TDNN, improves the prediction accuracy on a limited training data set (419 promoter sequences).

3.2 Application of NNPP in *Drosophila melanogaster*: The *Adh* region

To apply the 2–layer time–delay neural network to contiguous genomic sequence, a window of 51 base pairs is shifted over the sequence base by base. In this way, a score is computed for every position in the sequence. These individual scores are subsequently smoothed by a simple but efficient function, which selects the position of the highest score in a window of 10 neighboring positions as the final prediction. The smoothing function is implemented as a post–processing procedure and is part of the final NNPP.

To test the accuracy of NNPP in *Drosophila melanogaster*, NNPP was applied to the 2.9 Mbase genomic sequence of the *Adh* region (Ashburner *et al.*, 1999). A careful promoter analysis in this region (Reese *et al.*, 2000a) resulted in high quality full–length cDNA alignments for 92 genes out of the original 222 gene annotations including.
In Table 2 the NNPP results are reported on this test set of genes in the *Adh* region (Ashburner et al., 1999) in comparison to CoreInspector (Scherf *et al.*, 2000) and MCPromoter (Ohler *et al.*, 1999) both evaluated in a recent annotation experiment (Reese et al., 2000a). Although NNPP is far from accurate, this test shows good results similar to those in a 1997 review by Fickett and Hatzigeorgiou (Fickett & Hatzigeorgiou, 1997). In this paper they reported a recognition rate of 54% of the known promoters at a threshold of 0.8. In *Adh*, the same threshold identifies 69 or 75% of the total of 92 annotated promoters with a false positive rate of 1/547, similar to the rate of 1/460 reported in (Fickett & Hatzigeorgiou, 1997). It has to be noted that Fickett and Hatzigeorgiou used both strands to calculate the false positive rate while for *Adh* only the gene strand was used. If one applies a more stringent threshold of 0.97, 35 of the 92 promoters are still recognized with a much lower false positive rate of 1/2,416. The higher classification rate might be due to biased promoter selection in (Fickett & Hatzigeorgiou, 1997).

4 Discussion

The presented tool is an artificial neural network model using a time–delay network architecture. This network has two feature layers: one for the TATA box and one for the *Inr* (initiator). The output of both feature layers is combined in a time–delay neural network. I have shown that such a neural network detects the TATA box and the Inr and is insensitive to their relative spacing and is therefore an excellent model for the compositional sequence properties of a eukaryotic core promoter region. The discriminative ability of such a model for the short core promoter region of –40 to +11 bases spanning the transcription start site is so strong that this model can be used to predict an entire promoter in genomic DNA. These results show that the highest information content in a promoter region exists in the core promoter region.

The NNPP computer program implements the time–delay neural network model. The program is able to predict over 70% of transcription start sites in genomic DNA when used with the default parameters. The false positive rate calculated on the *Adh* region *in Drosophila melanogaster* is 1/ 547 bases. The Matthew's correlation coefficient (Matthews, 1975) is

	System name	Identified TSS	Rate of false predictions in annotated *Adh* region (total 853,180 bases)
From (Reese et al., 2000a)	CoreInspector	1 (1.0%)	1/853,180 (0.00012%)
	MCPromoter v2.0	31 (33.6%)	1/2,437 (0.041%)
NNPP	NNPP (t=0.99)	20 (21.7%)	1/6,227 (0.016%)
	NNPP (t=0.97)	35 (38.0%)	1/2,416 (0.041%)
	NNPP (t=0.80)	69 (75.0%)	1/547 (0.183%)
	NNPP (t=0.70)	80 (86.9%)	1/400 (0.250%)

Table 2: Evaluation of promoter prediction systems on the *Adh* region. The table only shows the results of the "search by signal" program (CoreInspector) and "search by content" programs (MCPromoter) from the experiment of Reese *et al.* (2000a) and the prediction sets from NNPP with different thresholds. The rate of false positives is shown for the sequence where cDNA annotations define the region as non–promoter.

0.58. 30% of all promoter sequences remain undetected and this is probably due to the non–local structure of the promoter region, where initiation control elements can occur at positions many kilobases distant from the transcription start site.
The NNPP program can easily be extended to incorporate novel information as it becomes available. Other known promoter elements such as the CAAT box, GC box, DPE (downstream promoter element; so far known to exist only in Drosophila), and conserved transcription factor binding sites can also be used within the existing framework. The extended parameter space of such an extended model would require more data for training.

The positive results obtained using the time delay architecture will hopefully lead to more widespread application of neural networks to similarly complex problems in molecular biology, such as the detection of splice sites and protein–protein interaction motifs. For the application to complete genome annotations the NNPP code needs to be integrated into a more global annotation system such as Genie (Reese *et al.*, 2000b).

Since I made the NNPP program available on the World Wide Web it has been widely used in the scientific community to hypothesize about potential transcription start sites. Recently the program was used to correct an important *C. elegans* gene, *unc–86*, that encodes a POU IV class transcription factor. In this study the transcription start site prediction by NNPP was experimentally verified through experiments (Roehrig, 2000, personal communication).

This example demonstrates how useful a program like NNPP can be in the right context. It is clear that a program cannot substitute for the final experimental proof but the example shows that it can give direction and guidance for such experiments to verify computational predictions.

References

M. Ashburner, S. Misra, J. Roote, S. E. Lewis, R. Blazej, T. Davis, C. Doyle, R. Galle, R. George, N. Harris, G. Hartzell, D. Harvey, L. Hong, K. Houston, R. Hoskins, G. Johnson, C. Martin, A. Moshrefi, M. Palazzolo, M. G. Reese, A. Spradling, G. Tsang, K. Wan, K. Whitelaw, B. Kimmel and et al. An exploration of the sequence of a 2.9–Mb region of the genome of drosophila melanogaster. The adh region. *Genetics,* 153(1): 179–219, 1999.

P. Bucher. Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences. *J Mol Biol,* 212(4): 563–78, 1990.

M. Burset and R. Guigo. Evaluation of gene structure prediction programs. *Genomics,* 34(3): 353–67, 1996.

J. W. Fickett and A. G. Hatzigeorgiou. Eukaryotic promoter recognition. *Genome Res,* 7(9): 861–78, 1997.

D. Haussler. Computational Genefinding. *Trends in Biochemical Sciences, Supplementary Guide to Bioinformatics* 12–15, 1998.

R. D. Kornberg. Eukaryotic transcriptional control. *Trends Cell Biol,* 9(12): M46–9, 1999.

D. Kulp, D. Haussler, M. G. Reese and F. H. Eeckman. A generalized hidden Markov model for the recognition of human genes in DNA. *Ismb,* 4 134–42, 1996.

K. J. Lang and A. H. Waibel. A Time–Delay Neural Network Architecture for isolated Word Recognition. *Neural Networks,* 3 23–43, 1990.

B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim Biophys Acta,* 405(2): 442–51, 1975.

U. Ohler, S. Harbeck, H. Niemann, E. Noth and M. G. Reese. Interpolated markov chains for eukaryotic promoter recognition. *Bioinformatics,* 15(5): 362–9, 1999.

F. E. Penotti. Human DNA TATA boxes and transcription initiation sites. A statistical study. *J Mol Biol,* 213(1): 37–52, 1990.

B. F. Pugh. Mechanisms of transcription complex assembly. *Curr Opin Cell Biol,* 8(3): 303–11, 1996.

B. F. Pugh and R. Tjian. Diverse transcriptional functions of the multisubunit eukaryotic TFIID complex. *J Biol Chem,* 267(2): 679–82, 1992.

M. G. Reese, F. H. Eeckman, D. Kulp and D. Haussler. Improved splice site detection in Genie. *J Comput Biol,* 4(3): 311–23, 1997.

M. G. Reese, G. Hartzell, N. L. Harris, U. Ohler and S. E. Lewis. Genome Annotation Assessment in *Drosophila melanogaster. Genome Research,* 10(4):, 2000a.

M. G. Reese, D. Kulp, H. Tammana and D. Haussler. Genie– Gene finding in *Drosophila melanogaster. Genome Research,* 10(4):, 2000b.

M. Scherf, A. Klingenhoff and T. Werner. *in preparation.*, 2000.

G. D. Stormo and D. Haussler. Optimally parsing a sequence into different classes based on multiple types of evidence. *Ismb,* 2 369–75, 1994.

A. H. Waibel, T. Hanazawa, G. E. Hinton, K. Shikano and K. J. Lang. Phoneme Recognition Using Time–Delay Neural Networks. *IEEE Transactions on Acoustic, Speech, and Signal Processing,* 37(3): 328–39, 1989.

K. Yokomori, C. P. Verrijzer and R. Tjian. An interplay between TATA box–binding protein and transcription factors IIE and IIA modulates DNA binding and transcription. *Proc Natl Acad Sci U S A,* 95(12): 6722–7, 1998.

A. Zell *et al.* Stuttgart Neural Network Simulator (SNNS) 4.2 edit.,(Issue):. http://www–ra.informatik.uni–tuebingen.de/SNNS/, 1999.

Generating Protein Three-Dimensional Folds Signatures Using Inductive Logic Programming

M. Turcotte*; S. H. Muggleton†; M. J. E. Sternberg*

*Biomolecular Modelling Laboratory
Imperial Cancer Research Fund
{M.Turcotte, M.Sternberg}@icrf.icnet.uk

† Department of Computer Science
University of York
stephen@cs.york.ac.uk

Abstract

This paper presents an application of Inductive Logic Programming (ILP) to automatically discover protein folds signatures. Cross-validation experiments were carried out for the 20 most populated folds. The overall cross-validated accuracy is 75%. The signatures associate local sub-structures and sequence motifs to complex three-dimensional folds. The signatures often highlight regions which are important for function, such as the nucleotide binding site in the case of the Rossmann-fold. ILP can produce rules which are both accurate and easily amenable to human interpretation.

1 Introduction

Understanding protein structure is one of the major challenges of molecular biology. Despite more than three decades of research, predicting the three-dimensional structure from the knowledge of sequence information alone remains an elusive goal. However, the coming of the functional genomics era and the explosion of sequence data are now putting tremendous pressure for progress to be made. Indeed, it is generally believed that a detailed knowledge of the three-dimensional structure is essential to understand, and eventually manipulate, protein function.

Although the pace of protein three-dimensional structure determination has been slower than that of the sequence determination, large amounts of data have been accumulated over the years; approximately 10,000 protein structures are found in the public repository. To facilitate understanding classification schemes have recently been developed. One example is SCOP (Structural Classification of Proteins) (Brenner et al., 1996). Such classifications are hierarchical, proteins which are known to have evolved from a common ancestry are grouped together into families, and super-families. The next level puts together proteins that share the same fold, *i.e.* the same core secondary structure elements and the same interconnections. In this case, the similarity may be the result of convergence towards a stable architecture. At this level, the proteins have quite dissimilar sequences which makes it impossible for sequence comparison methods to detect the kinship. Here, this classification scheme is the starting point for a machine learning experiment which aim to relate local structures to the concept of folds.

2 Protein 3D structure

The three-dimensional structure of proteins is highly complex. In general, three levels of abstraction are distinguished: primary, secondary and tertiary structure. Proteins are long chains of amino acids. There are 20 naturally occurring amino acids, each with different chemical properties. The amino acids are linked by a covalent bond to form chains, typically 100 to 500 amino acids long, and referred to as primary structure or sequence. A particular sequence folds into a specific compact three-dimensional or tertiary structure. The two predominant methods to structure determination are X-ray crystallography and NMR spectroscopy. Those techniques require sophisticated equipments and because of technological limitations, the sequences of amino acids are routinely determined in large quantities whilst the determination of the three-dimensional structure remains difficult. Early on it was predicted that segments of the primary sequence would adopt local regular structures (Pauling et al., 1951), the two main types are the α-helices and the β-strands, while the intervening regions are called loops or coils, collectively those elements are referred to as the secondary structure.

Identifying rules which explain the observed folds remains a challenge and often involves manual intervention of experts (Brenner et al., 1996; Branden and Tooze, 1999; Orengo et al., 1994). For several folds, these signatures are reported in the literature, generally after extensive study. A few experts are familiar with many of these rules and the knowledge is certainly not formalised, with a common language, in a form suitable for automated testing as new structures are determined. Also, automated methods can identify features that are missed by manual examination.

3 Approach

The objective of this work is to automate the discovery of structural rules, also referred to as signatures. Inductive Logic Programming (ILP) is a logic-based approach to machine learning. ILP is particularly well suited to study problems encountered in molecular biology. First, protein structures are the result of complex interactions between sub-structures (secondary structures) and the ability to learn relations might prove to be a key feature. Second, ILP systems can make use of problem-specific background knowledge taking advantage of the vast amount of knowledge that has been accumulated. Third, ILP uses a common representation for the examples, the background knowledge and the hypotheses, and therefore provides a good integration for the development of applications together with the machine learning experiments. Finally, the hypotheses can be made readable, by straightforward translation to natural languages, and integrated to the cycles of scientific debates. In complex domains, such as the structure determination, it is unlikely that a breakthrough will come from a single machine learning experiment, the ability of ILP to make the rules readable is therefore an important advantage to assist the process of scientific discovery.

3.1 Machine learning algorithm

Inductive Logic Programming is concerned with the induction of hypotheses from examples and background knowledge (Muggleton and Raedt, 1994). In this work, we use Progol which is being developed by the second author (Muggleton and Firth, 1999). As mentioned above, a restricted subset of first-order logic is used as a common representation for the examples, the background knowledge and also the generated hypotheses. In the case of the protein folds problem, a (positive) example represents the fact that the domain d1hlb__ belongs to the Globin fold by fold('Globin-like', d1hlb__). The background knowledge contains information such as the relationships between secondary structures and the presence of a proline. The algorithm then constructs a hypothesis which explains this example in terms of the background knowledge, the following rule was generated for the Globin-like fold,

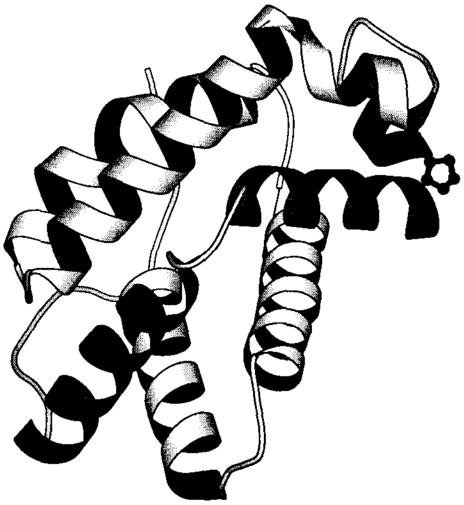


Figure 1: Schematic representation of a member of the globin fold. The first and the second helix that contains the proline are coloured in black. This figure was prepared using Molscript version 2.1 Kraulis (1991).

```
fold('Globin-like', X) :-
    adjacent(X, _, B, 1, h, h),
    has_pro(B).
```

which is interpreted as "domain X belongs to the Globin fold if its first helix is followed by another one that contains a proline". Figure 3.1 illustrate this rule.

More specifically, the background knowledge for those experiments contains information about the secondary structure, calculated with PROMOTIF (Hutchinson and Thornton, 1996) from experimental three-dimensional structures. For each secondary structure we calculate the average hydrophobicity, the hydrophobic moment and the number of amino acids. The presence of a proline is also noted. For each inter-secondary structure region we calculate the number of amino acids. The background knowledge contains global information as well: the total number of strands and helices and the total number of amino acids.

4 Discussion

We have compared two different representations of the background knowledge (Turcotte et al.). For the first one, the background knowledge contained only predicates which encode global characteristics of protein folds, specifically, the total number of residues and the total number of secondary structures of both types, helices and strands. For the second approach, new predicates were added which introduce relationships between secondary structure elements and their properties. The results showed that it is possible to construct good classifiers with a background knowledge which is essentially limited to attribute-values. Statistically better results were obtained with the relational representation, however this might be attributable to the fact that more features were used. Interestingly, in the case of the relational dataset, some of rules can be related to results published in the relevant scientific literature. One such example is that of the Globin fold.

Rule 1 (Globin fold) *Helix A at position 1 is followed by helix B. B contains a proline residue.*

```
fold('Globin-like', X) :-
    adjacent(X, A, B, 1, h, h),
    has_pro(B).
```

The Globin-fold is a good example of divergent evolution. In SCOP, this fold comprises diverse sequences such as myoglobin, hemoglobin and phycocyanins. Yet the three-dimensional structure of these proteins is well preserved. One hallmark of this fold is the presence of a conserved proline residue in helix B, which causes a sharp bend in the main chain. This observation has been reported previously by Bashford *et al.* (Bashford et al., 1987) and is illustrated in Figure 3.1.

One of the main limitations of this application concerns the representation. Secondary structure positions are counted from the N-terminal end of the structure and do not take into account the possibility of insertions. We have developed a new representation that i) sequentially numbers the secondary structures for the C-terminal as well as N-terminal and ii) includes additional information about the topology of the sheets and the packing the helices. Preliminary runs show that Progol can now learn descriptions such as the following:

```
fold(A,'SH3-like barrel') :-
    number_strands(4=<A=<7),
    sheet(A,B,anti),
    has_n_strands(B,5),
    strand(A,C,B,1),
    strand(A,D,B,-1),
    antiparallel(C,D).
```

which allows for insertion into the sheet. Cross-validation data and detailed analysis will be presented at the conference.

Those experiments show that ILP can be used effectively to learn rules in complex domains such as protein structure. The rules produced in the context of the relational learning experiments, were found to be more informative, as judged by our knowledge of protein structure, than those generated in the context of attribute-value experiments. The rules can be explained in terms of structural and/or functional concepts, such active site and binding location. Indeed, when constructing a rule, Progol looks for motifs which are common to all the domains of a given fold but almost never encountered in others, except for a limited number of cases which is set by a user defined threshold (noise). Characteristics which are important for structure and/or function are conserved amongst members of the same fold, at least up to the super-family level. Therefore the rules constructed by Progol can sometimes identify conserved functional motifs. Of the 59 rules generated, at least 5 can be related to previously published results.

Acknowledgements

This work is supported by a BBSRC/EPSRC Bioinformatics grant.

References

D. Bashford, C. Chothia, and A. M. Lesk. Determinants of a protein fold. unique features of the globin amino acid sequences. *Journal of Molecular Biology*, 196(1): 199–216, 1987.

Carl Branden and John Tooze. *Introduction to Protein Structure.* Garland, 1999.

S. E. Brenner, C. Chothia, T. J. Hubbard, and A. G. Murzin. Understanding protein structure: using SCOP for fold interpretation. *Methods in Enzymology*, 266: 635–43, 1996.

E. G. Hutchinson and J. M. Thornton. PROMOTIF – a program to identify and analyze structural motifs in proteins. *Protein Science*, 5(2):212–20, 1996.

Per J. Kraulis. Molscript: A program to produce both detailed and schematic plots of protein structures. *Journal of Applied Crystallography*, 24:946–950, 1991.

S. Muggleton and J. Firth. CProgol4.4: Theory and use. In Sašo Džeroski and Nada Lavrac, editors, *Inductive Logic Programing and Knowledge Discovery in Databases*. 1999. forthcoming book.

S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19/20:629–679, 1994.

C. A. Orengo, D. T. Jones, and J. M. Thornton. Protein superfamilies and domain superfolds. *Nature*, 372(6507): 631–4, 1994.

L. Pauling, R. B. Corey, and H. R. Branson. The structure of proteins: Two hydrogen-bonded helical configurations of the polypeptide chain. *Proc. Natl. Acad. Sci. USA*, 37:205–210, 1951.

M. Turcotte, S.H. Muggleton, and M.J.E. Sternberg. Application of inductive logic programming to derive protein three-dimensional folds signatures. *submitted to Machine Learning*.