Pacific Symposium on Biocomputing 5:401-412 (2000)

IDENTIFYING AMINO ACID RESIDUES IN MEDIUM RESOLUTION CRITICAL POINT GRAPHS USING INSTANCE BASED QUERY GENERATION

K. WHELAN, J. GLASGOW

Instance Based Query Generation is defined and applied to the problem of recognising amino acid residues in medium resolution critical point graphs. The technique is an amalgamation of Relational Instance Based Learning and Frequent Query Discovery in First Order Logic. Instances are automatically constructed from a deductive database and first order association rules are derived from the instances. The initial investigations presented here indicate that the technique is able to discriminate some of the larger amino acid types as well as discriminating the protein from background solvent. Identification of the smaller amino acids remains difficult and requires further work.

1 Introduction

A fundamental goal of research in molecular biology is to understand protein structure. Protein crystallography is currently the most successful method for determining the three-dimensional conformation of a protein, yet it remains labor intensive and relies on an expert's ability to derive and evaluate a protein scene model. The problem of protein structure determination may be formulated as an exercise in *computational scene analysis*[1,2] in which a three-dimensional image of a protein (the electron density map) is segmented into a graph of *critical points* (points where the gradient of the electron density is equal to zero)[3]. At medium resolution, $\sim$3 Å, it has been demonstrated that a *peak critical point* in the graph generally corresponds to the location of an amino acid residue along the backbone of the structure[4,5]. For larger residues, peaks may also denote the location of side chain configurations. *Pass critical points* derived from medium resolution maps correspond to peptide bonds or to side chain connectivities. Thus, peak/pass traces through a critical point graph correspond to potential backbone traces for the protein.

Figure 1 illustrates a depiction of a tryptophan residue and examples of how this residue may be represented as a critical point subgraph resulting from a topological analysis of an electron density map. In Figure 1(b) the residue is denoted as one backbone and two side chain peak critical points. Figure 1(c)

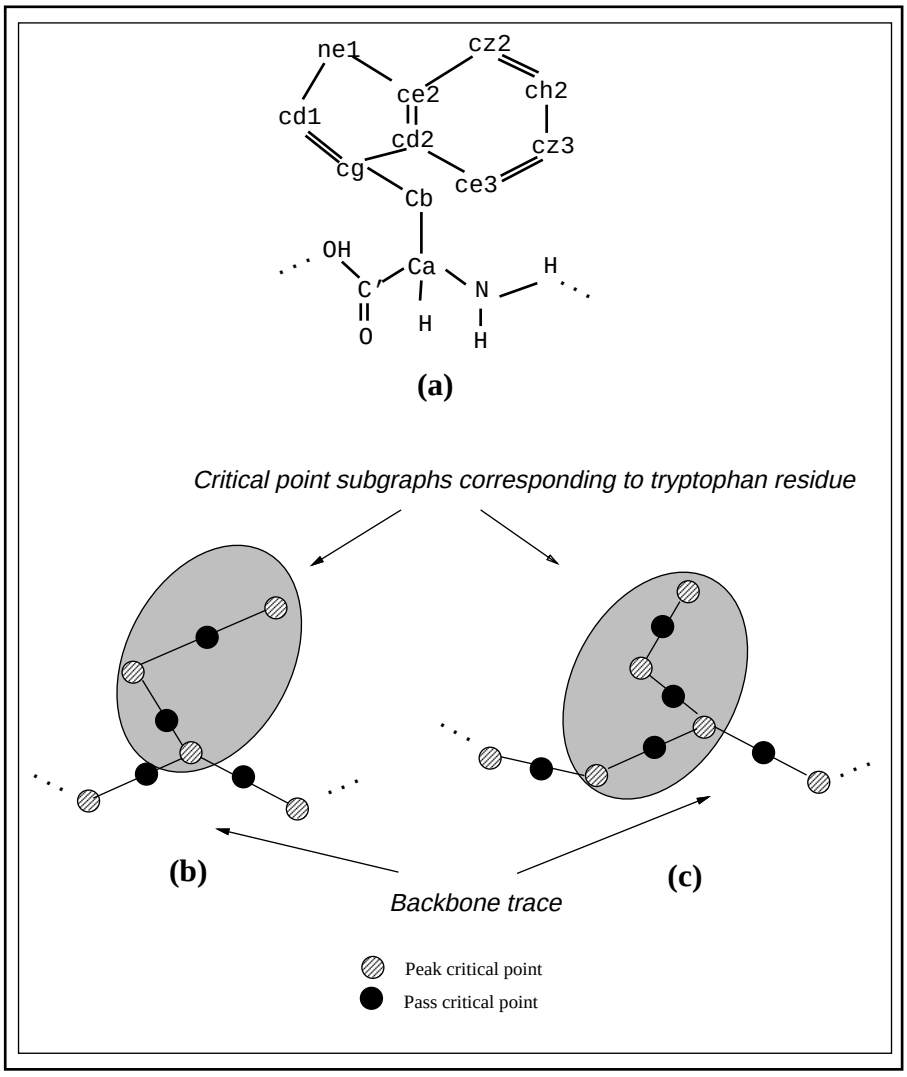


Figure 1: Representations of: (a) tryptophan amino acid residue; (b) critical point subgraph for tryptophan containing side chain with two peaks; (c) critical point subgraph for tryptophan containing side chain with three peaks.

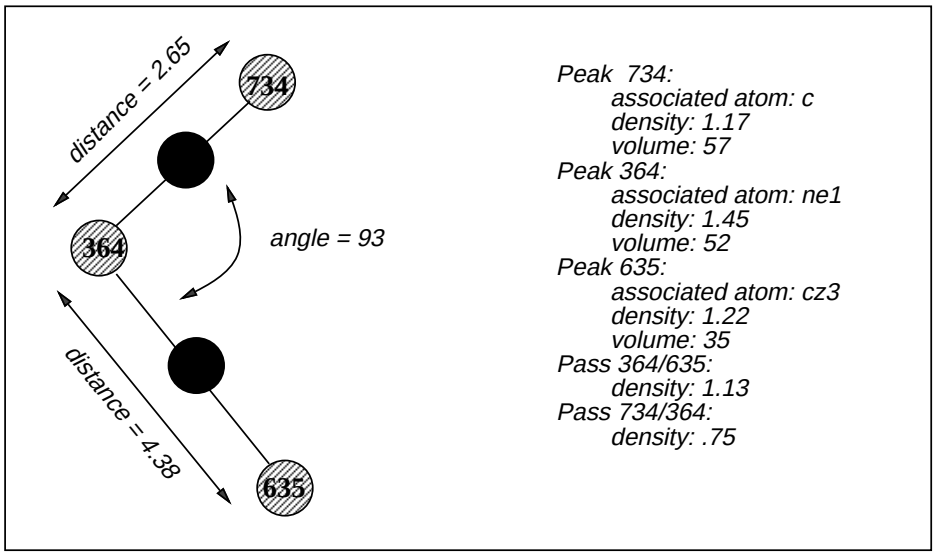
illustrates an example where tryptophan has two backbone and two side chain peaks.

The research described in this paper explores the application of instance-based query generation and learning to the problem of classifying amino acid residues in a critical point graph. In particular, attributes related to the geometric configuration of subgraphs (including calculated angles and distances) along with information obtained from the electron density map (volume and density of critical points), are derived and used to assist in the identification of residues.

2 Instance-Based Query Generation

We define *instance-based query generation* as an amalgamation of methods from *relational instance-based learning* [6] and methods from *knowledge discovery in databases*. Similar to the system WARMR [7], the goal of instance-based query generation is to discover frequently occurring motifs and association rules in deductive databases. Relational instance-based learning can be considered as a first-order logic extension of less expressive propositional logic approaches (e.g., IB3 [8]). Methods in instance-based learning generate and store a set of instances; classification of new instances is performed by the use of a similarity measure to identify the known instances most similar to the unseen instance. This is a k-nearest neighbour approach where an unseen example is allotted to the most similar class. This approach is most common in supervised learning tasks where an instance is an element of a deductive database represented as a logic statement corresponding to the collection of all facts required to describe the instance. For example, Figure 2 illustrates a critical point graph for an instance of *tryptophan* along with a query consisting of logical axioms that describe the instance of the residue occurring in chain a of protein $1dog$.

In general, an instance in a deductive database is a conjunction of axioms associated with a particular entity, where each axiom expresses a relation of interest for the entity. For example, the first axiom in the query declares that the instance corresponds to a tryptophan (*trp*) residue and is the sixth residue in chain a for the protein 1dog. The remaining axioms associate peaks in the critical point graph with particular atoms in the backbone or side chain for the tryptophan instance and associate volume, angle and density values with the individual critical points. For example, the relation *peak(1dog, 635, 1.2* -



(a)

residue(1dog, a, 6, trp),
backbone_association(1dog, a, 6, c, 734),
side_chain_association(1dog, a, 6, ne1, 364),
side_chain_association(1dog, a, 6, cz3, 635),
peak(1dog, 734, 1.1 - 1.2, 55 - 60),
peak(1dog, 364, 1.4 - 1.5, 50 - 55),
peak(1dog, 635, 1.2 - 1.3, 30 - 35),
edge(1dog, 173, 364, 635, 1.1 - 1.2, 2.6 - 2.8),
edge(1dog, 872, 734, 364, 0.7 - 0.8, 4.0 - 4.2),
angle(1dog, 635, 364, 734, 90 - 120).

(b)

Figure 2: (a) Critical point representation of tryptophan instance, and (b) Logic representation of instance.

1.3, 30 - 35) states that peak 635 in the critical point graph for protein 1*dog* has a density value in the range 1.2 - 1.3 and a volume in the range 30 - 35.

Note that the numeric values for an instance are discretized in the construction of the axioms that define the instance. For example, the distance 2.65 Å between peaks 364 and 734 for the instance depicted in Figure 2 was generalized to the range 2.6 - 2.8 Å in the instance declaration. In the construction of these axioms, values for volume, distance density and angle were split into equal sized categories, with unit increments of: 0.1 electrons for density; 5 electrons per cubic Å for volume; 0.2 Å for distance, and 30 degrees for angle.

In our approach to query generation, instances are automatically generated from a deductive database by declaring a set of *modes* and *types* that describe the contents of the axioms comprising each instance [6]. This is a common method by which systems such as PROGOL and WARMR constrain the search space for good hypotheses [9]. A mode defines whether a term in the relation is an input variable (+); an output variable (-); either an input or and output variable (+-); or a constant (#). Types define the range of possible values for the parameters. For example, the type "aa_acid" (amino acid) type ranges over a set of values that correspond to the 20 different amino acids. The mode and type declarations used in this application conform to those used by WARMR, where a special mode (keymode) is used to define the database key for a particular instance. The modes used to express instances of amino acid residues in a critical point graph are given as follows:

keymode(residue(-pid,-chainid,-rid,#aa_acid)).
mode(backbone_association(+pid,+chainid,+rid,#atom_name,-peakid)).
mode(side_chain_association(+pid,+chainid,+rid,#atom_name,-peakid)).
mode(peak(+pid,+peakid,#density,#volume)).
mode(edge(+pid,-edgeid,+peakid,+peakid,#density,#distance)).
mode(angle(pid,+peakid,+peakid,+peakid,#angle)).

The types for these modes are defined in Table 1.

Once the set of instances has been generated for a domain, the next step in instance-based query generation involves a process where mode terms containing constants are generalized to variables. The resulting axioms are referred to as *queries*. The queries generated by this method are analogous to the bottom clauses of the refinement lattice utilised by specific to general ILP programs

TYPE	DEFINITION
pid	name of protein
chainid	letter corresponding to protein chain
rid	unique number identifying amino acid residue
aa_acid	amino acid type
atom name	name of atom in peak/atom association
peakid	unique peak number
edgeid	unique edge number
density	electron density
volume	number of electrons per cubic Å
distance	distance between peaks (in Å's)
angle	angle formed by three peaks (in degrees)

Table 1: Types of data used to define amino acid residues in critical point graphs

such as GOLEM[10]. For example, a generalization of the instance of tryptophan described in Figure 2 is the query[a]:

residue(Pid, Chainid, Rid, trp),
backbone_association(Pid, Chainid, Rid, c, Pk1),
side_chain_association(Pid, Chainid, Rid, ne1, Pk2),
side_chain_association(Pid, Chainid, Rid, cz3, Pk3),
peak(Pid, Pk1, 1.1 - 1.2, 55 - 60),
peak(Pid, Pk2, 1.4 - 1.5, 50 - 55),
peak(Pid, Pk3, 1.2 - 1.3, 30 - 35),
edge(Pid, Edge1, Pk2, Pk3, 1.1 - 1.2, 2.6 - 2.8),
edge(Pid, Edge2, Pk1, Pk2, 0.7 - 0.8, 4.0 - 4.2),
angle(Pid, Pk3, Pk2, Pk1, 90 - 120).

A goal of our research is to determine frequently occurring motifs in the data and associate these with individual amino acid residues. A frequency measure for each query is calculated based on the proportion of instances that are true for (subsumed by) a given query. Truth of a query can be determined by invoking the theorem proving interpreter of Prolog.

Once a set of frequently occurring queries has been determined, the final step in instance-based query generation involves deriving *query extensions*. These are first-order logic representations of association rules that can be used

[a] Note that variable parameters begin with a capital letter in a query.

to explain the given queries. This approach to rule discovery is a technique common to knowledge discovery [11]. In deductive and relational databases association rules define inferences (implications) between the literals or tables comprising the database.

Following is an example of an association rule that can be used to identify a tryptophan residue:

lhs([peak(Pid, Pk1, 1.1 - 1.2, 55 - 60),
peak(Pid, Pk2, 1.4 - 1.5, 50 - 55),
peak(Pid, Pk3, 1.2 - 1.3, 30 - 35),
edge(Pid, Edge1, Pk2, Pk3, 1.1 - 1.2, 2.6 - 2.8),
edge(Pid, Edge2, Pk1, Pk2, 0.7 - 0.8, 4.0 - 4.2),
angle(Pid, Pk3, Pk2, Pk1, 90 - 120)]) $\rightsquigarrow$

rhs([residue(Pid, Chainid, Rid, trp),
backbone_association(Pid, Chainid, Rid, c, Pk1),
side_chain_association(Pid, Chainid, Rid, ne1, Pk2),
side_chain_association(Pid, Chainid, Rid, cz3, Pk3)]).

Note that the rule is of the form $lhs(...) \rightsquigarrow rhs(...)$, which can be interpreted as: "if the query forming the left hand side (lhs) succeeds, then the extended query formed by the conjunction of the left hand side and the right hand side also succeeds". i.e *If LHS then LHS* $\wedge$ *RHS*. This interpretation distinguishes query extensions from definite clauses defined by implication, where all instantiations of the right hand side must be true for the definite clause to be true. The association rule states that a subgraph that exhibits the geometry described in the left hand side of the query extension can be associated with a tryptophan residue where the three peaks correspond to the atoms c, $ne1$ and $cz3$ in the residue.

We calculate the confidence of a query extension in order to provide a probablistic measure of how often the association is true. Confidence is determined by comparing the frequency of the extended query with the frequency of the left hand side of the query [7]. Frequency is a measure of the proportion of examples for which a given query succeeds. Query extensions are often used to classify unseen instances in a supervised learning problem. For the amino acid residue domain, the confidence measure describes the probability of a given subgraph representing the amino acid residue. If a subgraph satisfies the left hand side of multiple query extensions, then the confidence measures define a probability distribution of the possible classifications for the subgraph.

The steps described above for instance-based query generation can be summarized as follows:

1. Generate instances using mode and type declarations.
2. Derive queries for each instance by replacing non-constant terms by variables (according to the mode and type declarations).
3. Determine query extensions from queries derived in step 2.
4. Calculate a confidence value for each query extension.

3 Experimental Methods

Instance based query generation was applied to a training set consisting of the critical point graphs from 12 proteins. These were "ideal" maps generated from the PDB using the program XTAL and interpreted by the CCRIT program. These proteins were selected from different superfamilies and contained a total of 5826 amino acid residues. The graphs had a total of 13625 peaks and 27449 passes. One query was generated for each residue and equivalent queries removed before the query extensions were generated. The resulting antecedent literals were also tested for equivalence to allow the generation of query extensions with disjunctive consequents. Creating disjunctive consequents indicates how many ways a particular subgraph can be interpreted, and how confident each interpretation is. A total of 1703 of these disjunctive query extensions were generated from the 5826 amino acids.

Eighteen test proteins were chosen for evaluation of the discovered query extensions. Each of the proteins was unseen, i.e. was not included in the training set, and each protein was also an ideal critical point graph. The query extensions were used to classify the unseen residues by generating a number of predictions of the subgraphs found. These predictions were generated in a number of ways:

- All predictions were generated.
- Predictions were generated by randomly allocating the set of subgraphs to interpretations sorted according to the distribution of confidences of the various disjunctive consequents

- Only those predictions conforming to the consequent with the highest confidence were generated.

The test proteins also contain a list of associations with atoms from the PDB file from which it was generated. This list was used to evaluate the predictions generated.

4 Results and Discussion

4.1 Results

Initial investigations calculated the percentage of instances in each test protein that were found to be equivalent to the queries generated from the training set. The mean number of equivalent instances is 83.68%, indicating that there is a limited set of subgraphs that conform to the residues types. Investigation of the confidence of the query extensions reveals that many of the subgraphs may be interpreted in many different ways, often with a low confidence. Table 2 illustrates the overall confidence found for each residue type, this gives an indication of the ability of the set of query extensions to correctly identify a given residue type. Many of the smaller residues are very difficult to predict, but some of the larger types, particularly arginine and tryptophan may be predicted with greater than 50% accuracy.

The ability of the query extensions to discriminate protein peaks from non-protein peaks was also undertaken, in an effort to determine how often the query extensions include background solvent. Analysis of the critical point graphs revealed that up to 40% of peaks may not be part of the protein. Currently a density cut-off measure is used to determine protein from non-protein. The list of peaks corresponding to those subgraphs identified by the query extensions was consistently more accurate than the list found by the density cut off. Indeed a result of 98% accuracy with less than 2% false positives was achieved in one case.

Classification accuracy of the query extensions was determined using the test proteins. As predicted by the confidences listed in table two, the query extensions performed poorly. If all predictions were generated, the accuracy could indeed match the percentage of shared cases. However there was an enormous number of predictions. The probability of a prediction being correct

residue type	overall confidence	residue type	overall confidence
ala	0.012	leu	0.0857
arg	0.552	lys	0.166
asn	0.058	met	0.099
asp	0.039	phe	0.138
cys	0.073	pro	0.048
gln	0.061	ser	0.055
glu	0.116	thr	0.031
gly	0.112	trp	0.579
his	0.136	tyr	0.331
ile	0.062	val	0.050

Table 2: Overall confidence of predicting each Amino Acid type by using Query Extensions

was about 1% i.e. there were 99% false positives. The other two methods greatly reduced the number of false positives but classification accuracy fell to between 4 and 11%. This is only a little better than the 3.5% accuracy achieved by randomly allocating an amino acid type to any peak.

4.2 Discussion

The low classification accuracy obtained by the query extensions illustrates the difficulty associated with the identification of individual residues from medium resolution critical point graphs. However, the high confidence results for tryptophan and arginine suggest that the query extensions can successfully identify these larger residue types and may be used to further enhance the performance of other interpretation techniques such as protein threading. The query extensions can also be used to preprocess the critical point graphs, allowing the protein to be distinguished from the background before any interpretation is undertaken.

There are, however, many issues raised by the development of this technique. Solving these issue will increase the performance of the technique. Many instances of the larger residue types were unique or very rare, thus the query extensions generated from these have a high confidence but are never actually used in classification. This results in few predictions of these residue types and the ability to identify them is poor, despite the high confidence. Currently

only smaller subgraphs are classified, and with little confidence of being interpreted correctly. The larger residue types are thus being classified by the wrong query extensions. This problem will be addressed by the incorporation of an inexact matching strategy that computes the similarity between queries. This development was considered during the development of instance based query generation, but incorporation of it was not possible for this experiment. This development will also remove the current need for discretisation of numerical information which also contributes to the poor performance of the query extensions.

The motivation for the development of instance based query generation is to avoid the expensive levelwise generation of queries as implemented by WARMR; by generating instances in a similar way to RIBL; and to avoid the combinatorial k-nearest-neighbour evaluation of RIBL; where all instances are stored; by constructing queries and utilising syntactic matching of variables wherever possible. It is hoped that these improvements will allow the reasonably efficient generation of a subsumption lattice of queries with the incorporation of inexact matching to compensate for the over-specific nature of larger query extensions.

The experiment itself did not take into account all of the possible interpretations of subgraphs: it assumes that all of the subgraphs found can be classified only as amino acids. Some investigation of the mistakes made by the query extensions revealed that a subgraph proposed by a query extension may be interpreted in a different way, e.g. as a backbone edge between two residues or a non-peptide interaction between two different parts of the protein. Incorporation of instances corresponding to these interpretations may improve the ability of instance based query generation to interpret regions of the critical point graph. The next iteration of this experiment will also include additional attributes relevant to the peak critical points - these include a shape measure and a calculation of the moment of inertia.

5 Conclusions

The experiments conducted to test the ability of instance based query generation to identify amino acid residues reveal that the method currently identifies most amino acid residues poorly. However it displays the potential for reliable predictions of the larger types and is very good at discriminating the protein

from the background solvent. There is potential for improving the interpretation of the critical point graph if the inexact matching strategy is implemented and the additional understanding of the nature of the critical point graphs gained from this experiment is incorporated into future experiments. Instance based query generation will also be used in conjunction with other methods such as protein theading and backbone discovery to further improve the machine interpretation of the critical point graphs.

1. S. Fortier, I. Castleden, J. Glasgow, D. Conklin, C. Walmsley, L. Leherte, and F.H. Allen. Molecular scene analysis: The integration of direct methods and artificial intelligence strategies for solving protein crystal structures. *Acta Crystallographica*, D49:168–178, 1993.
2. J.I. Glasgow, S. Fortier, and F.H. Allen. Molecular scene analysis: crystal structure determination through imagery. In L. Hunter, editor, *Artificial Intelligence and Molecular Biology*, pages 433–458. AAAI Press, Menlo Park, California, 1993.
3. C.K. Johnson. Peaks, passes, pales and pits: a tour through the critical points of interest in density maps. In *Proceedings of the American Crystallographic Association Meeting*, Asilomar, California, 1977. Abstract JQ6.
4. L. Leherte, S. Fortier, J. Glasgow, and F.H. Allen. Molecular scene analysis: A topological approach to the automated interpretation of protein electron density maps. *Acta Crystallographica D*, D50:155–166, 1994.
5. L. Leherte, J. Glasgow, K. Baxter, E. Steeg, and S. Fortier. Analysis of three-dimensional protein images. *Journal of Artificial Intelligence Research (JAIR)*, pages 125–159, 1997.
6. W. Emde and D. Wettschereck. Relational instance based learning. In *Proc. 13th Conference on Machine Learning*, 1996.
7. L. Dehapse and H Toivonen. Discovery of frequent datalog patterns. *Data Mining and Knowledge Discovery*, 3(1):7—36, 1999.
8. D.W. Aha, D. Kibler, and M.K Albert. Instance based learning algorithms. *Machine Learning*, 6:37–66, 1991.
9. Muggleton S. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.
10. R.D. King, D.A. Clark, J. Shirazi, and M.J.E Sternberg. On the use of machine learning to identify topological rules in the packing of β—strands. *Protein Engineering*, 7(11):1295–1303, 1994.
11. T. Argawal, T. Imielinski, R. Srikant, Toivonen H., and A.I. Verkamo. Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining*, pages 307 — 328, Menlo Park CA, 1996. AAAI Press.