Knowledge Discovery in Molecular Structure Databases

by

Darrell Conklin

A thesis submitted to the Department of Computing and Information Science

in conformity with the requirements for

the degree of Doctor of Philosophy

Queen's University

Kingston, Ontario, Canada

April, 1995

Abstract

In recent years there has been an explosion of information generated on molecular structures. This information is usually three-dimensional in nature, and is normally appended to the existing molecular database in the form of raw coordinate data. There is a recognized need for new tools to structure, manage, and analyze these data. This thesis presents a knowledge discovery technique, based on a theory of conceptual clustering, in which recurrent associations between molecular constitution and conformation can be discovered in three-dimensional molecular structure databases. These associations are defined in first-order logic as structured concepts, and are maintained in a taxonomy partially ordered by subsumption. The knowledge discovery method is applied to two large molecular structure databases. In the Cambridge Structural Database the technique is used to perform conformational clustering of several small molecule datasets. The resultant conformational templates are evaluated by comparison with hand analyses and statistical clusterings of the same data. In the Protein Data Bank the technique is used to discover recurrent protein motifs which have sequences predictive of their associated structure. The resultant rules are evaluated with respect to the average structural variation of their instances, and according to a test of statistical correlation between sequence and structure. In both molecular structure databases interesting and significant associations were discovered.

Acknowledgements

My initial inspiration and excitement for the topic of knowledge discovery in molecular databases developed during intensive research meetings with Frank Allen, Suzanne Fortier, and Janice Glasgow at Queen's University in late 1990. Since then, and during my research, I have become increasingly fascinated by the topic. For their role in maintaining my interest, special thanks is due to my supervisors, Janice Glasgow and Suzanne Fortier. They have provided constant encouragement, enthusiasm, interest, and support for this work.

Frank Allen at the Cambridge Crystallographic Data Centre has been a tremendous source of information and inspiration for this work. Much of Chapter 5 would not have been possible without his help in providing me with structural data and listings from statistical clustering experiments. I would also like to thank the Cambridge Crystallographic Data Centre for providing me with the opportunity to use their facilities during several research visits from 1992 through 1995.

I would like to thank Gilles Bisson, Diane Cook, Bob Levinson and Kevin Thompson for their generous time and valuable comments on a draft of Chapter 2. Jude Shavlik, as editor of the Machine Learning journal, helped out a lot with the overall presentation and clarity of Chapter 6. Gregory Piatetsky-Shapiro offered useful advice on the statistical evaluation of predictive rules.

Several groups of people have listened to presentations of this thesis material. For their generous time and useful comments, I would like to thank the Department of Information Studies at Sheffield University, the Ottawa Machine Learning Group, the Biomolecular Informatics group at ZymoGenetics Inc. and the Molecular Scene Analysis Group at Queen's University.

The prototype software developed to support this thesis was done in the Q'Nial array programming language. I would like to thank the creator of the language, Mike Jenkins, for taking an interest in my work and for ensuring that I always had the best programming facilities to work with. Chris Walmsley has given me all kinds of technical advice and support; in particular, I would like to acknowledge his MolView software which was used to create some of the figures in Chapters 5 and 6. Figure 5.10 was prepared using the daVinci graph layout system. The diagrams in Figure II.1 were prepared using the QUEST and PLUTO programs from the Cambridge Structural Database.

This thesis is dedicated to my grandmother, Phyllis Mobray Hughes (1906-1990).

Contents

List of Tables

List of Figures

Glossary of Notation

References in this glossary are to the page where the notation in question is introduced.

$\langle \mathcal{D}, [\![\cdot]\!]_u^I \rangle$	a first-order interpretation with domain $\mathcal{D}$	8
$\models$	first-order entailment	9
$\succeq_{\mathcal{T}}$	subsumption in a theory $\mathcal{T}$	9
$\equiv_{\mathcal{T}}$	extensional equivalence under theory $\mathcal{T}$	9
$(f \mid x \mapsto a)$	the function f', like f, except that $f'(x) = a$	10
$\exists^*$	restricted existential quantification	11
$\overline{v}^m$	a set containing m distinct variables $v_1, \ldots, v_m$	11
π	the fundamental structured concept relation	11
$\Pi(x, \overline{v}^m)$	shorthand for $\pi(x, v_1) \wedge \ldots \wedge \pi(x, v_m)$	11
$\mathrm{ran}(f)$	the range or codomain of the function f	11
θ	a substitution	12
$\overline{X}$	the first-order equivalent of the $\mathcal{SDL}$ term X	45
$[A, R]$	shorthand for the $\mathcal{SDL}$ term (`image` A R)	45
$\mathtt{exp}_{\mathcal{T}}$	term expansion in terminology $\mathcal{T}$	46
$\{\!\vert X \vert\!\}_s$	the operational effect of $\mathcal{SDL}$ statement X in state s	47
$a \overset{f}{\mapsto} b$	$f(a) = b$	48
$\vdash$	first-order derivability	49
$A \overset{R}{\to} B$	the set of all R-monomorphisms from A to B	50
$f : A \overset{R}{\to} B$	f is an R-monomorphism from A to B	51
$A - B$	set difference: $\{x : x \in A \wedge x \notin B\}$	58
$\vert A \vert$	the cardinality of set A	59
$S_{\mathcal{T}}$	a similarity coefficient in $\mathcal{SDL}$	59
$A \cong_R B$	A is R-isomorphic to B	59
$A \overset{R}{\rightharpoondown} B$	the set of all partial R-monomorphisms from A to B	59
$f : A \overset{R}{\rightharpoondown} B$	f is a partial R-monomorphism from A to B	60
$\mathrm{dom}(f)$	the domain of the function f	60
τ	the similarity threshold for concept formation	64
$\mathtt{inst}(X)$	the known instances of concept X	70
$\mathtt{erad}(a)$	the elemental radius of an atom of type a	81
α	the atomic bonding tolerance parameter	81
$\mathtt{d}(x, y)$	the Euclidean distance between points x and y	81
γ	the Klyne-Prelog planarity parameter	83
$\mathtt{t}(w, x, y, z)$	the torsion angle among points w, x, y, z	84
$\mathtt{vrad}(a)$	the van der Waals radius of an atom of type a	95
β	the van der Waals bonding tolerance parameter	95
$\mathrm{graph}(f)$	$\{(x, f(x)) : x \in \mathrm{dom}(f)\}$	124
f^{-1}	the inverse of function f: $f^{-1}(f(a)) = a$	133
$f; g$	the composite of functions f and g: $(f;g)(a) = g(f(a))$	133
$\Re^n$	n-dimensional Euclidean space	135

Chapter 1

Introduction

This thesis explores the application of a knowledge discovery technique to large databases of three-dimensional molecular structures. The central hypothesis of this thesis is that interesting and useful classes of molecular conformations can be represented using specific structured concepts, and that structured concept discovery techniques can be used to extract these concepts from data.

The field of *knowledge discovery in databases* is concerned with the theory and processes involved in the representation and efficient extraction of patterns or *concepts* and rules from very large databases. Among all recurrent patterns, only a few will be known explicitly, and many will be hidden or implicit in database entries. Although all database management systems have search facilities which can recognize instances of patterns, they presuppose that the user has the *a priori* knowledge to formulate meaningful queries. *Pattern discovery*, whether by human or machine, *precedes pattern recognition.* Discovered concepts can be used to group data into meaningful classes, summarizing the data and revealing deviant entries. They can also be used as new primitives to express rules.

The need for automated discovery techniques is particularly acute in molecular structure databases, which are experiencing a growing accumulation of raw three-dimensional atomic coordinate data. Recurrent associations between molecular constitution and conformation can be used as structural templates for building models for molecules of unknown structure. If the associations are predictive, then predictions of structure may be made on the basis of known constitution. Discovered structural patterns can also be used to organize the raw data, providing a conceptual framework to which new data can be related.

One way knowledge discovery can be automated is to instruct a computer to look for similarities among objects, and to express similar portions as generalized concepts. A number of issues arise from adopting this abstract framework. First, a suitably powerful *knowledge representation* for expressing objects, concepts, and prior knowledge must be chosen. A machine can only discover concepts which it can represent internally as configurations of symbols. Concepts must have a *semantics*: it should be possible to specify what it means for an object to be an instance of a concept. This semantics must be put into practice with a sound, complete and efficient algorithm. It should be possible to detect whether one concept *subsumes* another, that is, whether any possible instance of one concept is always an instance of another. Concepts should be *comprehensible*: their semantics should be readily understood by humans. It is also necessary to specify what it means for two patterns to be *similar*. The discovery machine must be *unsupervised*, in that it is not provided with the predetermination that only certain data items can be matched to form a concept. Finally, all discovery systems face an *evaluation problem*: out of the vast number of possible generalizations of the data, which ones represent interesting discoveries? This thesis addresses all of the above issues in ways which will hopefully become clear, and which will be revisited in the Conclusions.

Molecular structures are substances nicely conceptualized as *structured objects*: they have structured parts (e.g., atoms and amino acids) which have relations (e.g., connectivity and bond angles) between them. The words "structure" and "relations," in machine learning research, immediately suggest the need for powerful concept representation languages. Those languages which express objects and patterns only as lists of *features* are inadequate for the task of knowledge discovery in molecular structure databases. The knowledge representation scheme chosen here uses the language of *structured concepts*, which is an expressive subset of *first-order logic*.

Computer Science deals with the representation, creation, processing and management of information. <u>Part I</u> of this thesis is a contribution to computer science, in particular, to machine learning and knowledge representation research. It presents the theory and methods of structured concept discovery, and the particular techniques used to perform knowledge discovery in molecular structure databases. This part of the thesis does not presuppose any background in structural chemistry. <u>Chapter 2</u> surveys the machine learning literature for several instances of structured concept discovery systems. The chapter presents a formal logic, used to unify and express the semantics of several systems. None of the surveyed systems are found to be fully adequate for general-purpose concept discovery. A number of future research directions for the field of structured concept discovery are proposed.

With the theoretical foundations of structured concepts in hand, Chapter 3 formalizes a Spatial Description Logic. The main inferences of a *description logic* are subsumption and *classification*: placing a new description (concept) in the correct location in an existing taxonomy of concepts. In machine learning terms, the spatial description logic circumscribes the hypothesis space for a structured concept discovery system. The logic is unique in that it is *direct*: inferences about relations are computed using functions which inspect spatial configurations of symbols. It provides a concise, comprehensible language for expressing both structured concepts and prior knowledge. The logic is particularly attractive for molecular structures, since these are usually expressed as three-dimensional spatial arrangements of atoms. A method for computing subsumption in the logic, along with a completeness proof, is presented. The contribution of this chapter is a formal logic combining the principles of direct representations and structured concepts.

Chapter 4 presents a theory for measuring similarity between structured concepts. This theory is given by a set of axioms which should be satisfied by any particular similarity function. An important point of this chapter is the *indexing theorem*, which relates concept similarity and concept subsumption. A detailed specification for a structured concept discovery system is presented. This system discovers new concepts by iteratively retrieving similar ones from a taxonomy of extant concepts, and performing a generalization step. The system has been implemented and its application to molecular structure databases is presented in the second part of this thesis. The contribution of this chapter is a similarity-based clustering method which uses a direct knowledge representation for structured concepts.

Structural Chemistry deals with the structure, function and metrical aspects of chemical compounds. Part II of this thesis is a contribution to structural chemistry, in particular to conformational analysis and protein motif identification. This part of the thesis reveals the task of *conformational clustering*, which is the recognition of recurrent three-dimensional arrangements of a given molecular graph. This problem has typically been approached either by paper and pencil, or by machine using statistical clustering schemes rather than ones based on symbolic logic. Statistical clustering schemes have some problems when applied to 3D molecular data. One problem is that user intervention is often required to determine the optimal number of clusters. Another problem is that quantitative dissimilarity between structures, which drives all statistical clustering schemes, can be counterintuitive, not necessarily reflecting topological or qualitative spatial similarity. The hypothesis of this part of the thesis is that structured concepts can be used to represent aspects of molecular conformation, and that the discovery technique developed in Part I offers a useful and complementary approach to existing conformational analysis techniques.

To explore this hypothesis, the technique is applied to several medium-size ($<$ 300 structures) three-dimensional datasets drawn from the Cambridge Structural Database, the largest source of three-dimensional information on small organic molecules. These results are presented in Chapter 5. The chosen datasets illustrate a variety of conformational phenomena: cyclic structures, an acyclic structure unconstrained by ring geometry, and a structure determined by stereochemical effects. Each of these structures has previously been analyzed using statistical clustering schemes, providing external corroboration for the results presented in the chapter.

The task of conformational clustering is carried through to Chapter 6, which is concerned with the automated discovery of structure-sequence associations in polypeptide fragments. These are drawn from the Protein Data Bank, the major source of three-dimensional information on protein structures. The chapter provides an extensive review of previous attempts at protein motif discovery, and finds that none of them fully integrate the representation and discovery of protein structure and sequence motifs. It is hypothesized that the technique presented in Part I of this thesis can be used to discover *predictive* motifs, where the sequence determines or nearly determines the associated three-dimensional structure. To explore this hypothesis, two very large databases ($>$ 15000 fragments) are processed, and all discovered concepts are filtered through a set of stringent evaluation criteria designed to reject non-predictive motifs. Several interesting and predictive motifs are discovered, supporting the main hypothesis of the chapter.

Finally, Chapter 7 summarizes the findings of this thesis, and proposes some directions for future research.

Part I

Representation and Discovery of Structured Concepts

Chapter 2

Structured Concept Discovery: Theory and Methods

...never to lose sight of the distinction between concept and object.

Gottlob Frege's Third Principle (1884).

The field of knowledge discovery is concerned with the theory and processes involved in finding and representing patterns and regularities previously unknown. A new generation of knowledge discovery tools now deals with *structured concepts*: these capture associations between relations among the components of structured objects. This chapter outlines a logic used to express structured concepts, and surveys a number of systems performing structured concept discovery. The chapter concludes with a discussion of important future research directions for the field.

2.1 Introduction and motivations

Knowledge discovery systems search for regularities and detect interesting and useful patterns from incoming data. There is increasing interest in this field, due to the growing number of giga- or tera-byte databases containing largely uninterpreted information. It is becoming clear that automated techniques are needed for structuring, managing, compressing, assimilating and extracting general knowledge from these data.

As an abstract motivation for knowledge discovery, imagine that we have created a robot which lives on twigs and scrap iron and set it free to roam the countryside. To survive, it must have the ability to recognize new things, particularly those hostile to its survival. While we, as system designers, will have provided the robot with a language for representing facts and knowledge, we cannot have anticipated all types of situations the robot will find itself in. One tactic the robot might use is to simply remember factual descriptions of all objects encountered, matching new objects to those in the database and making predictions on the basis of similarity. This process, known as *reasoning by analogy*, will become increasingly tedious as the size of the database grows. If the robot can discover associations and regularities in the data, these can be used both to make predictions about new objects and to index database objects for efficient retrieval.

Current knowledge discovery systems (Matheus et al. 1993) use a highly restricted knowledge representation language for representing patterns and regularities: they can capture associations among different *features* of objects. These attribute-value learning systems have had many practical applications, but are fundamentally limited by their knowledge representation language. A new generation of knowledge discovery tools now deals with *structured concepts*: they capture associations between relations among the components of objects. Such tools are essential, for example, in data analysis and object recognition tasks where the features of an object's components alone are insufficient to determine its classification or function. A description of the interrelationships among parts is crucial.

This chapter has two purposes. One is to present a logic in which all structured concepts discovery systems can be discussed and related. Another purpose of this chapter is to survey a number of operational systems which perform discovery of structured concepts. Although Thompson and Langley (1991) provided a survey of several systems, with recent advancements there is a need for a broader discussion which includes theoretical foundations, and surveys of newer systems, miscellaneous systems, and application domains. This chapter concludes with a discussion of critical future research issues in the field.

2.2 Structured concepts: theoretical foundations

A *concept*, defined extensionally, is simply a set of things in a universe. Machine learning systems construct and manipulate intensional concept descriptions. Knowledge representation is crucial for a machine learning system, as the set of possible learnable concepts is restricted to the set of concepts the system can construct or represent. Much of the early research on machine learning was restricted to a knowledge representation formally equivalent to propositional logic. Objects are described by sets of *features*, or attribute/value pairs. Recent work has seen a shift to more powerful first-order formalisms capable of describing structured objects and relational concepts.

An early paper on structured concept learning (Dietterich and Michalski 1981) lamented the fact that each research group was using its own notation and terminology, making it hard to relate different systems performing similar learning tasks. The situation for current structured concept discovery systems is not much improved. This is especially regrettable in the light of theoretical advancements in the parallel field of knowledge representation. This section will outline a unified notation and theory to which all current structured concept discovery systems can be related.

2.2.1 Logical concepts

A *first-order language* comprises variable, constant, and predicate symbols with their arities.[1] A *term* in a given language is either a variable or a constant. An *atom* is an expression $p(t_1, \ldots, t_n)$, where $t_1, \ldots, t_n$ are terms, and p is an n–ary predicate symbol. An atom is *ground* if it does not contain any variables. A *literal* is an atom (A) or the negation of an atom ($\texttt{not}\, A$). A *well-formed-formula* in the language is formed using literals and operators such as conjunction, negation, and implication, and the quantifiers $\forall$ and $\exists$. A *sentence* is a closed well-formed-formula (all variables quantified). A *concept* can be represented by a unary λ predicate[2]

$$\lambda x.E, \tag{2.1}$$

where x is the only free variable in the well-formed-formula E. Finally, a *clause* is a disjunction of literals, with all variables universally quantified.

The extension of a concept is all things in a universe of discourse such that when their denoting constant is substituted for the variable x in (2.1), E is satisfied. More formally, an *interpretation* I of a first-order language is given by a structure

$$\langle \mathcal{D}, [\![\cdot]\!]_u^I \rangle,$$

where $\mathcal{D}$ is a universe of discourse, and $[\![\cdot]\!]_u^I$ is a (polymorphic) function which maps n-ary predicates to elements of $\mathcal{D}^n \rightarrow \{true, false\}$, sentences to $\{true, false\}$, and variables to elements of $\mathcal{D}$ via the variable assignment function u (see Table 2.1).

A *theory* is a set of sentences in a first-order language. A *model* for a theory is an

[1] See (Lloyd 1987; Genesereth and Nilsson 1988; Crossley et al. 1972) for good discussions of first-order logic. Here function-free first-order logic is described.

[2] This is the standard definition of a concept used by the knowledge representation community (Woods 1991; Woods and Schmolze 1992) and the machine discovery community (Zytkow 1993). See also (Hayes 1985a) for related knowledge representation issues. See (Tennent 1991) for a review of the λ calculus.

interpretation which maps every sentence in the set to *true*. A concept C is *instantiated* by an object a, with respect to a theory $\mathcal{T}$, if

$$\mathcal{T} \models C(a), \tag{2.2}$$

that is, if $C(a)$ is *true* in all models of $\mathcal{T}$. A concept C is said to *subsume* a concept D, with respect to a theory $\mathcal{T}$, written $C \succeq_{\mathcal{T}} D$, if C is instantiated whenever D is:

$$\mathcal{T} \models \forall \mathtt{x}\, D(\mathtt{x}) \Rightarrow C(\mathtt{x}). \tag{2.3}$$

Two concepts C and D are (extensionally) *equivalent* with respect to a theory $\mathcal{T}$, written $C \equiv_{\mathcal{T}} D$, if they co-subsume each other:

$$\mathcal{T} \models \forall \mathtt{x}\, D(\mathtt{x}) \Leftrightarrow C(\mathtt{x}).$$

Two concepts C and D are (extensionally) *disjoint* with respect to a theory $\mathcal{T}$, if instantiation of one implies noninstantiation of the other.

Before we continue, a word about the use of concepts as a hypothesis space for knowledge discovery is in order. Concepts are good tools for representing associations between symbolic attributes of objects. As pointed out by Zytkow (1993), they have limitations and are not the proper formalism for capturing feature associations or numerical regularities. For example, the sentence

$$\forall \mathtt{x}\, \mathtt{height}(\mathtt{x}) = c \times \mathtt{weight}(\mathtt{x})$$

represents a regularity between two attributes `height` and `weight` which holds for all objects. This regularity, however, is poorly represented by concepts, as a new concept must be produced for each value in the range of the `height` function.

A simple example

As a simple illustration of the theory outlined in this section, consider the following example, adapted from (Buntine 1988). Consider the simple theory $\mathcal{T}$ comprising the two sentences

$$\forall \mathtt{x}\, \mathtt{cat}(\mathtt{x}) \Rightarrow \mathtt{pet}(\mathtt{x})$$
$$\forall \mathtt{x}\, \mathtt{dog}(\mathtt{x}) \Rightarrow \mathtt{pet}(\mathtt{x})$$

1. $[\![p(v_1,\ldots,v_n)]\!]^I_u = [\![p]\!]^I_u([\![v_1]\!]^I_u,\ldots,[\![v_n]\!]^I_u)$ *(predicate application)*
2. $[\![\mathtt{true}]\!]^I_u = \mathit{true}$ *(truth constant)*
3. $[\![\mathtt{not}\ P]\!]^I_u = \mathit{true}$ iff $[\![P]\!]^I_u = \mathit{false}$ *(negation)*
4. $[\![\forall v\ P]\!]^I_u = \mathit{true}$ iff $[\![P]\!]^I_{(u\ \mid\ v\mapsto c)} = \mathit{true}$ for all $c \in \mathcal{D}$ *(universal quantification)*
5. $[\![\exists v\ P]\!]^I_u = \mathit{true}$ iff $[\![P]\!]^I_{(u\ \mid\ v\mapsto c)} = \mathit{true}$ for some $c \in \mathcal{D}$ *(existential quantification)*
6. $[\![\exists^* \overline{v}^m\ P]\!]^I_u = \mathit{true}$ iff $[\![P]\!]^I_{u'} = \mathit{true}$, where
   $u' = (u \mid v_1 \mapsto c_1, \ldots, v_m \mapsto c_m)$, $\{c_1,\ldots,c_m\} \subseteq \mathcal{D}$, and
   $\forall ij\ c_i = c_j \Rightarrow i = j$ *(restricted existential quantification)*
7. $[\![P \wedge Q]\!]^I_u = \mathit{true}$ if $[\![P]\!]^I_u = \mathit{true}$ and $[\![Q]\!]^I_u = \mathit{true}$ *(conjunction)*
8. $[\![\lambda x.E]\!]^I_u = f \in \mathcal{D} \rightarrow \{\mathit{true}, \mathit{false}\}$, where
   $f(a) = [\![E]\!]^I_{(u\ \mid\ x\mapsto a)}$ *(concept description)*
9. $[\![P \Rightarrow Q]\!]^I_u = \mathit{true}$ if $[\![P]\!]^I_u = \mathit{false}$ or $[\![Q]\!]^I_u = \mathit{true}$ *(implication)*

Table 2.1: The formal semantics of a first-order logic

In any model of $\mathcal{T}$, cats and dogs are pets. Let the concept C1 be

$$\lambda\mathtt{x}.\mathtt{small}(\mathtt{x}) \wedge \mathtt{fluffy}(\mathtt{x}) \wedge \mathtt{dog}(\mathtt{x})$$

The meaning of this concept (see Table 2.1; rules 8, 7, and 1) is a function that returns *true* for only those objects that are fluffy, small dogs. Let the concept C2 be

$$\lambda\mathtt{x}.\mathtt{fluffy}(\mathtt{x}) \wedge \mathtt{cat}(\mathtt{x})$$

that is, the concept of "fluffy cats". Then the concept of "fluffy pets"

$$\lambda\mathtt{x}.\mathtt{fluffy}(\mathtt{x}) \wedge \mathtt{pet}(\mathtt{x})$$

subsumes both C1 and C2 with respect to $\mathcal{T}$. It subsumes C1 because all small, fluffy dogs are also small fluffy pets, and hence they are also fluffy pets. It subsumes C2 because all fluffy cats are also fluffy pets. These subsumption relations can be verified by substituting for C and D in Expression 2.3 and applying logical deduction.

2.2.2 Structured concepts

Machine learning systems search over sets of possible concept definitions. This underlying first-order language is usually referred to as a *hypothesis space*. The hypothesis space used in the simple example above was weak, allowing only unary predicates. This section will develop and discuss a more powerful hypothesis space for *structured concepts*.

A *structured object* comprises parts along with defined relations among these parts.[3] A *structured concept* has structured objects in its extension. Haussler (1989) has given a precise characterization of a hypothesis space for structured concepts. This will be presented

[3] Later on we will encounter *composite* structured objects, which recursively comprise other structured objects as parts.

here, in a form slightly modified to allow for new theoretical and application developments.

An *existential conjunctive concept* is a sentence of the form

$$(\exists^* \overline{v}^m)\, p_1 \wedge \ldots \wedge p_k \tag{2.4}$$

where $\overline{v}^m$ denotes a set of m variables $v_1, \ldots, v_m$, each p_i ($1 \leq i \leq k$) is a literal with variables from $\overline{v}^m$ as arguments.[4] Expression (2.4) does not quite conform to the definition of a concept given by Expression (2.1). It can be satisfied by an interpretation, but is not a predicate and therefore does not have an extension. This is easily corrected by wrapping the expression in a λ form, introducing a variable $\mathtt{x}$ and a new distinguished predicate π:

$$\lambda \mathtt{x}.(\exists^* \overline{v}^m)\, \Pi(\mathtt{x}, \overline{v}^m) \wedge p_1 \wedge \ldots \wedge p_k \tag{2.5}$$

where $\Pi(x, \overline{v}^m)$ is shorthand for the expression

$$\pi(x, v_1) \wedge \ldots \wedge \pi(x, v_m)$$

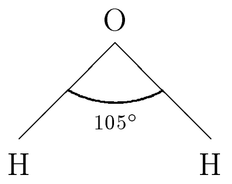
The predicate π, used to decompose a single object into parts, is called a *fundamental relation* (Thompson and Langley 1989) and its intended meaning is usually the "immediate part of" relation. Thus the meaning of (2.5) is "all objects containing at least these parts, in these particular relationships".[5] The predicate symbol π can only appear in this restricted circumstance.[6]

The notation $(\exists^* \overline{v}^m)$ in (2.4) means that the variables $v_1, \ldots, v_m$ refer to *distinct* objects (see Table 2.1). Expression (2.5) captures the semantics of structured concepts as used by many of the systems surveyed in Section 2.3. Stepp (1987) has called this a *contains* versus an *is* semantics. In the latter, the concept is satisfied by objects that have *only* the described parts. An *is* semantics can be encoded in two ways. One is to conjoin to the concept expression a sentence asserting that the object has no other parts. Another idea, not suggested by Stepp, is to form an m-ary predicate from a concept with m parts. The translation between the two forms is mechanical: I assume a *contains* semantics throughout this thesis.

[4]Haussler uses *functional relations*: the p_i are formulae of the form $p(\overline{v}^n) = y$ where p is an n–ary function. The effective hypothesis space is similar, although here quantitative attributes are not expressed. An n-ary functional relation p with $|\mathrm{ran}(p)| = k$ will give rise to k n-ary predicates. A function with an infinite range requires special treatment.

[5]Haussler calls the objects "scenes", in deference to the pioneering work of Winston (1975).

[6]This relation was not called `part-of` or `part` as I did not want to commit to nor influence a particular interpretation. The semantics of the `part` relation is far from clear (Winston et al. 1987; Hayes 1985b).




```
λx.∃* p1 p2 p3
    π(x,p1) ∧
    π(x,p2) ∧
    π(x,p3) ∧
    hydrogen(p1) ∧
    hydrogen(p2) ∧
    oxygen(p3) ∧
    bonded(p1,p3) ∧
    bonded(p2,p3) ∧
    not bonded(p1,p2) ∧
    angular(p1,p2,p3).
```

Figure 2.1: A water molecule. Left: molecular graph depicting two hydrogens bound to an oxygen atom. Right: a structured concept for the water molecule.

It may be useful at this point to visualize a structured concept — with only one binary relation in the first-order language — as a labelled graph: the parts being the vertices and relations between parts captured by edges. Figure 2.1 illustrates a structured concept for a water molecule. It has three parts — two hydrogen and one oxygen — decomposed by the π relation. The quantitative angle measurement (105°) is expressed as a qualitative *angular* spatial relationship (as opposed to a *linear* relationship). At the left is a graph representation of the concept: to the right is its representation in first-order logic.[7] As we shall see, structured concept discovery systems can use various internal representations to capture the standard semantics given in this section: the restricted first-order logic used here provides a unified notation.

2.2.3 Subsumption of concepts

Computing whether one concept subsumes — is more general than — another (recall Expression 2.3) is a central facility of all structured concept discovery systems. A common method for subsumption computation is based on the so–called *θ-subsumption* (Plotkin 1971; Gottlob 1987). I assume familiarity with the idea of substitutions. θ-subsumption usually applies to logical clauses rather than concepts: the modifications made for the following definition are very slight.

[7] Note the semantics assigned to the graph structure in Figure 2.1; the absence of an edge means that the vertices are *not* related by that relation (e.g., the two hydrogens are not bonded).

Definition 1 *A structured concept* $\lambda x.(\exists^* \overline{v}^m)\ C$ θ-subsumes *a structured concept* $\lambda x.(\exists^* \overline{w}^n)\ D$ *if there exists an injective substitution* $\theta : \overline{v}^m \rightarrow \overline{w}^n$ *such that* $C\theta \subseteq D$.

Proposition 1 *For any two structured concepts* C *and* D, $C \succeq_{\emptyset} D$ *if and only if* C θ-*subsumes* D.

(Note that the background theory is empty in $\succeq_{\emptyset}$). The proof of this completeness result is straightforward and a related proof can be found in (Gottlob 1987). The substitution θ can be restricted to be injective due to the $\exists^*$ quantifier. The weakness of θ-subsumption, as pointed out by Buntine (1988), is that it assumes that the prior theory $\mathcal{T}$ is empty. Although there have been many successes in machine learning in theory–free applications, recently researchers have come to realize that theory and knowledge representation is indispensable in many domains (Stepp and Michalski 1986).

From Expression (2.3), and the soundness and completeness of first-order logic, it follows that the subsumption relationship $C \succeq_{\mathcal{T}} D$ can be validated using theorem proving techniques. If the set of sentences

$$\mathcal{T} \cup \{\exists \mathtt{x}\ D(\mathtt{x}) \wedge \mathtt{not}\ C(\mathtt{x})\}$$

is unsatisfiable, then $C \succeq_{\mathcal{T}} D$. Unfortunately, in the presence of an arbitrary theory $\mathcal{T}$, this computation is only semi-decidable. It is, however, decidable in restricted first-order languages such as Datalog programs (sets of Horn clauses without function symbols). In such theories subsumption can be computed by *saturating* the concepts with background knowledge — expanding both concepts to semantically equivalent ones by applying all implications in the theory (Rouveirol 1994) — and testing for standard θ-subsumption.

The instantiation problem — determining whether an object is an instance of a structured concept (recall Expression 2.2) — is very similar to the subsumption problem. In fact, many machine learning systems rely on the *single representation trick* (Cohen and Feigenbaum 1982) by representing objects as specially identified individual concepts. In many terminological knowledge representation systems (Nebel 1990; MacGregor and Brill 1992), instantiation is also computed as subsumption between a concept and an individual concept.

The general θ-subsumption problem between clauses is known to be NP-complete (Garey and Johnson 1979, problem LO18). It is easy to show that it remains so when restricted to structured concepts. Since any structured concept discovery algorithm will need to perform something closely related to subsumption checking, as we shall see below, the need for managing complexity arises. While structured concepts are very powerful, in order to be of practical use the average-case complexity of subsumption checking should be polynomial.

The complexity of subsumption can be managed in various ways, while retaining the spirit of structured concepts. One is to simply acknowledge cases where subsumption may take an inordinate amount of time to compute, relying on an operator to make guiding decisions. Another is to curtail the expressiveness of the hypothesis space, for example, by omitting background theories and requiring that all attributes in a structured concept occur only once. In such a case, there will only be one possible matching between any two concept variables, and subsumption can be computed in polynomial time. Another way to manage complexity is to sacrifice completeness of subsumption checking, for example, by employing an efficient beam search through possible θ-substitutions (Dietterich and Michalski 1981). Yet another is to restrict the number of variables in a structured concept by keeping objects small and aggregating groups of parts in large objects into lower-level objects. The knowledge representation community has two views on this tradeoff between expressiveness and tractability (Levesque and Brachman 1987; Doyle and Patil 1991). Some researchers assert that the expressiveness of a knowledge representation language (or machine learning hypothesis space) should not be compromised. Others argue that a hypothesis space should be restricted so that sound and complete inferences can be made in reasonable time.

2.2.4 Common subsumers and generalization

Machine learning systems rely extensively on computing a concept in a hypothesis space which subsumes two or more other concepts. For two concepts C and D, and theory $\mathcal{T}$, a *common subsumer* is a concept L such that $L \succeq_{\mathcal{T}} C$ and $L \succeq_{\mathcal{T}} D$. A *least common subsumer* has the additional property that for no other common subsumer $L' \not\equiv_{\mathcal{T}} L$ in the hypothesis space does $L \succeq_{\mathcal{T}} L'$.

Constructing a common subsumer between two structured concepts is straightforward and can be done in linear time: essentially any injective matching between parts of the concepts will build a common subsumer (Haussler 1989). Computing a *least* common subsumer is difficult, and is readily shown to be isomorphic to the maximal matching problem which is known to be NP-complete (Garey and Johnson 1979, problem GT50). See (Barrow and Burstall 1976; McGregor 1982) for further discussions of the maximal common subgraph problem.

A task closely related to the common subsumer problem is the *generalization* problem:

given a theory $\mathcal{T}$ and a concept D, produce another concept C in the hypothesis space such that $C \succeq_{\mathcal{T}} D$. One rule of generalization, which accommodates a non-empty theory $\mathcal{T}$ and encompasses many of those generalization rules presented by Dietterich and Michalski (1981), is *truncation*: removing literals from saturated concepts (Rouveirol 1994). For example, the concept C1 from Section 2.2.1 is saturated to the concept

$$\lambda \mathtt{x}.\mathtt{small}(\mathtt{x}) \wedge \mathtt{fluffy}(\mathtt{x}) \wedge \mathtt{dog}(\mathtt{x}) \wedge \mathtt{pet}(\mathtt{x})$$

Subsuming concepts are then produced by removing one or more literals from this saturated concept.

As an aside, in the field of *inductive logic programming* (Muggleton 1992) (also see Section 2.2.6 below), Plotkin's (1971) *least general generalization* algorithm is sometimes employed as a common subsumer method (Muggleton and Feng 1992). While the algorithm runs in polynomial time in the size of two input concepts, when used with a background theory it may produce very long clauses. These will likely contain many redundant variables and literals which aggravate subsequent subsumption computations. Removing these redundant literals requires expensive subsumption testing. Cohen et al. (1992) studied the problem in the hypothesis space of *description logics* (also see Chapter 3 of this thesis), and found that the problem is intractable even for very restricted description logics.

2.2.5 Concept classification and organization

Most structured concept discovery systems organize their discoveries into a network of concepts. A *concept taxonomy* is an acyclic directed graph with vertices denoting concepts and edges denoting subsumption, modulo transitivity and reflexivity. A new concept is *classified* by placing it in its correct location in the taxonomy, just below all *most specific subsumers*, and just above all *most general subsumees* (Woods 1991). A concept C in the taxonomy is a most specific subsumer of a concept Q if $C \succeq_{\mathcal{T}} Q$, and for any other concept D in the taxonomy, $D \succeq_{\mathcal{T}} Q$ implies $D \succeq_{\mathcal{T}} C$. Similarly, a concept C in the taxonomy is a most general subsumee of a concept Q if $Q \succeq_{\mathcal{T}} C$, and for any other concept D in the taxonomy, $Q \succeq_{\mathcal{T}} D$ implies $C \succeq_{\mathcal{T}} D$.

To illustrate the theory outlined in this section, refer to Figure 2.2, which shows a taxonomy of structured concepts. A diagrammatic notation is used: two parts are related if and only if they are connected (on an adjacent row, column, or diagonal) in the diagram.[8]

[8]This is an example of a *direct* representation for structured concepts. Chapter 3 will have much more to say about this class of knowledge representation.

For the two query concepts Q1 and Q2 at the right of Figure 2.2, the most specific subsumers ($\mathtt{mss}$) and most general subsumees ($\mathtt{mgs}$) are as follows:

$$\mathtt{mss}(\mathrm{Q1}) = \{\mathrm{C6}, \mathrm{C1}\}$$
$$\mathtt{mgs}(\mathrm{Q1}) = \emptyset$$
$$\mathtt{mss}(\mathrm{Q2}) = \{\mathrm{C5}, \mathrm{C7}\}$$
$$\mathtt{mgs}(\mathrm{Q2}) = \{\mathrm{C12}\}$$

The children or *immediate subsumees* ($\mathtt{ims}$) of various concepts in the taxonomy are

$$\mathtt{ims}(\mathrm{C1}) = \{\mathrm{C7}, \mathrm{C9}\}$$
$$\mathtt{ims}(\mathrm{C5}) = \{\mathrm{C10}\}$$
$$\mathtt{ims}(\mathrm{C12}) = \emptyset$$

The *fringe* of — all concepts subsumed by — various concepts in the taxonomy are

$$\mathtt{fringe}(\mathrm{C1}) = \{\mathrm{C7}, \mathrm{C9}, \mathrm{C11}, \mathrm{C12}, \mathrm{C13}\}$$
$$\mathtt{fringe}(\mathrm{C8}) = \{\mathrm{C10}, \mathrm{C11}, \mathrm{C12}, \mathrm{C13}\}$$
$$\mathtt{fringe}(\mathrm{C12}) = \emptyset$$

Concept taxonomies can be used as background theories, as will be seen in a couple of structured concept discovery systems below. For example, the taxonomy of Figure 2.2 is a representation of the theory

$$\forall \mathtt{x}\ \mathrm{C12}(\mathtt{x}) \Rightarrow \mathrm{C10}(\mathtt{x})$$
$$\forall \mathtt{x}\ \mathrm{C10}(\mathtt{x}) \Rightarrow \mathrm{C6}(\mathtt{x})$$
$$\forall \mathtt{x}\ \mathrm{C9}(\mathtt{x}) \Rightarrow \mathrm{C4}(\mathtt{x})$$
$$\forall \mathtt{x}\ \mathrm{C3}(\mathtt{x}) \Rightarrow \mathrm{C0}(\mathtt{x})$$
$$\ldots$$

2.2.6 Structured concept discovery

Concept discovery systems can be categorized into three groups, based on their degree of autonomy. *Concept learning* systems are "supervised" by a teacher and restricted by the assumption that all examples are covered by one concept. *Conceptual clustering* systems are "unsupervised" and autonomously partition a set of examples using multiple concepts. *Exploratory* learning approaches are not necessarily guided by examples, and must explore the incoming data or hypothesis space autonomously.

Although the sections below will introduce concept learning, the main focus of this chapter will be on conceptual clustering of structured objects. A number of different systems for this task will be reviewed in Section 2.3.

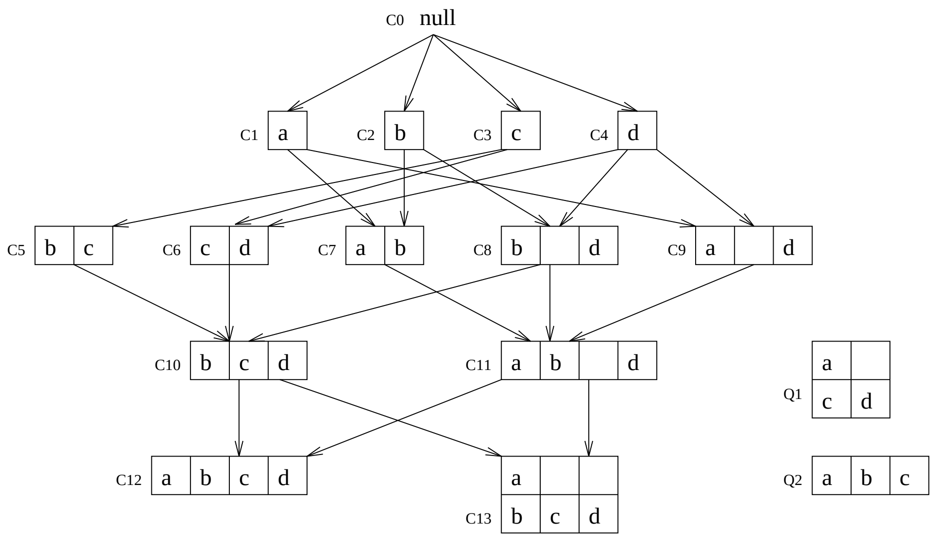


Figure 2.2: A structured concept taxonomy and two query structures

Concept learning

The goal of *concept learning*, one of the most fundamental and well-studied machine learning tasks, can be stated very simply. Given a target concept and examples of that concept, form an intensional description of the concept. The inference from examples to concept description is inductive, rather than deductive.

More formally, consider a universe of discourse $\mathcal{D}$ and let $S \subseteq \mathcal{D}$ be the set of objects that belong to the target concept. Furthermore, let $E \subseteq S$ be a set of n examples of the concept. Giving names $e_1, \ldots, e_n$ to these examples, axiomatize the world using a theory $\mathcal{T}$, and create a concept C such that for $i \in \{1, \ldots, n\}$

$$\mathcal{T} \models C(e_i). \tag{2.6}$$

Note that this expression can be computed by multiple proofs of instantiation relationships.

Concept learning from positive examples is an underconstrained task. Many concepts C will satisfy (2.6), including the two trivial concepts

$$\lambda\texttt{x}.\texttt{true},$$

and

$$\lambda\texttt{x}.(\texttt{x} = e_1) \vee \ldots \vee (\texttt{x} = e_n).$$

(Note that the latter expression is only possible in a hypothesis space supporting disjunction and equality). This invokes the well-known *problem of induction*: there is no logical basis for choosing between two or more admissible concepts. The problem arises regardless of the choice of hypothesis space. In structured concept learning there have been several attempts to address this problem. The most common one is to use *negative examples* of the concept, insisting that no negative example is instantiated by the concept (Muggleton and Feng 1992; Winston 1975; Quinlan 1990). Another way is to define the hypothesis space such that the system is incapable of expressing unreasonable inductive choices. Another is to *minimally* generalize the examples, for instance, by creating their least common subsumer (Dietterich and Michalski 1981). Another is to use an extra-logical preference criterion over inductive hypotheses. For example, both Conklin and Witten (1994) and Muggleton et al. (1992) show how the tradeoff between theory simplicity and the fit of a theory to examples can be balanced using a minimal length encoding heuristic. Finally, some systems rely on an "oracle" or a user to ensure them that overgeneralization has not occurred (Sammut and Banerji 1986).

There has been much work on structured concept learning. Dietterich and Michalski (1981) present a survey of early work, including Winston's (1975) landmark ARCH system. Researchers later realized that the hypothesis space of *logic programs* (Lloyd 1987) is very rich, supports disjunctive concepts, and naturally unifies and supports structured concept learning (Shapiro 1983; Buntine 1988). The new field of inductive logic programming (Muggleton 1992) has made great strides in formalizing the tasks of induction, generalization, and new predicate discovery in the presence of rich background theories expressed as logic programs.[9] The field is advancing rapidly: see (Muggleton 1993) for a recent review.

Conceptual clustering

The task of concept learning, described in the previous section, is to discover a concept which is satisfied by a given set of examples. The task of *conceptual clustering* is to discover a *set* of concepts which, taken together, cover a set of examples. An early definition of conceptual clustering was set out by Fisher and Langley (1986): there are many slight variations on the task. Consider a universe of discourse $\mathcal{D}$ and let $E \subseteq \mathcal{D}$ be a set of n examples. Giving names $e_1, \ldots, e_n$ to these examples, axiomatize the world using a theory $\mathcal{T}$, create a partitioning R of the examples, and for each $r \in R$ create a concept C such

[9] Although systems such as FOIL (Quinlan 1990) and GOLEM (Muggleton and Feng 1992) allow only ground atoms as background knowledge.

that for all $e \in r$

$$\mathcal{T} \models C(e). \tag{2.7}$$

A conceptual clustering system must therefore address two problems (Fisher and Langley 1986). The *aggregation* problem creates useful partitions of the example set. The *characterization* problem — the inductive concept learning problem — creates disjoint concepts which cover these sets. Conceptual clustering systems have tackled the vast search space of clusterings in various ways. One is to define a preference ordering on clusterings. Another is to perform incremental hill-climbing over the space of clusterings, perhaps also using preference criteria (Lebowitz 1987; Fisher 1987). A clustering will depend in such systems on the order of example presentation. Another is to enforce the restriction that the partition should be disjoint, as in the CLUSTER/2 system (Michalski and Stepp 1983b). In this case, instances of other concepts can be used as negative examples to a concept learning algorithm. Another approach is to fix the number of clusters in advance, and search over this much restricted hypothesis space. For example, the CLUSTER/2 system selects a fixed-size set of "seed" objects which are assumed to belong to disjoint clusters.

Conceptual clustering can be viewed as a restricted discovery process, as concepts represent recurrent structure among their instances. As we have seen, this recurrent structure can be put to many uses, such as indexing new structured objects with the same regularities, and making analogical inferences. When the exhaustiveness and disjointness restrictions of clustering are dropped or weakened, conceptual clustering is transformed into a more abstract discovery process: these are instances of *concept discovery* systems. Most work on conceptual clustering has focused on concepts expressed as attribute-value restrictions (disallowing any binary predicates). Recently there has been much interest in structured conceptual clustering.[10] The concept discovery systems reviewed in the next section all perform conceptual clustering of structured objects.

It is appropriate at this point to review a fundamental distinction between conceptual clustering (as developed by the field of machine learning) and numerical taxonomy (as developed by the field of pattern recognition). Numerical taxonomy methods represent objects as vectors in a multidimensional parameter space. A concept is also a vector representing a most representative instance or a centroid of a region of multidimensional space. In numerical taxonomy, instantiation relations are not computed by model-theoretic proof, but rather

[10] It appears that one of the first suggestions that conceptual clustering systems might use structured objects was given by Langley and Carbonell (1984).

What is the hypothesis space?
Can multilevel structured objects be represented?
What form, if any, of background knowledge is employed?
How is background knowledge used?
What is the learning algorithm? Is it incremental?
What is the complexity of subsumption testing and concept discovery?
What is the appropriate or intended semantics of concepts?
What form of memory structure is created by the system?
Are discovered concepts semantically or structurally disjoint?
How are new concepts created?
How are concepts selectively acquired?
Does the system monitor and evaluate its own performance?

Table 2.2: Pertinent questions for structured concept discovery systems

by measuring the *similarity* between an instance to a concept. The distinction between numerical taxonomy and conceptual clustering is blurred in some systems. For example, as we shall see, some systems use a model-theoretic definition of subsumption, but use a similarity semantics for instantiation. Other types of clustering can be characterized by the semantics given to concepts: for example, fuzzy clustering (Bezdek and Pal 1992) and probabilistic clustering (Cheeseman et al. 1988). See (Pitt and Reinke 1988; Gennari et al. 1989; Fisher and Langley 1986; Michalski and Stepp 1983a) for further comparisons of conceptual clustering and numerical taxonomy.

2.3 A review of selected methods

The preceding sections have laid the theoretical foundations for presenting a number of implemented systems which perform structured concept discovery. The order of presentation is roughly according to the first date of publication. Each system surveyed has different strengths, weaknesses, and features. Table 2.2 lists a number of questions that will be addressed throughout this section.

There has been a history in structured concept discovery research of modifying an attribute-value system to deal with structured descriptions. For example. CLUSTER/S is derived from CLUSTER/2, MERGE is derived from UNIMEM, LABYRINTH is derived from COBWEB. In the appropriate sections below a description of the associated attribute-value conceptual clustering system will be given.

2.3.1 MERGE

The UNIMEM system (Lebowitz 1987), like COBWEB (Fisher 1987), addresses the problem of incremental conceptual clustering in an attribute-value hypothesis space. UNIMEM has an interesting evaluation scheme for concepts, based on *predictability* analysis (Lebowitz 1986b). Similar to COBWEB, this is based on how well inferences about missing features can be made using a discovered concept. For each feature (unary literal) in a concept, a *confidence* score says how many times that feature had been confirmed or disconfirmed by examples in the past. After a new example passes through a concept — during the most specific subsumer computation — the confidence values are inspected. Features with low confidence are removed. If this results in a concept with too few features, the concept is removed. When an example reaches a most specific subsumer, high similarity with any of its immediate subsumees will trigger the creation of the least common subsumer[11], and the requisite local rearrangement of the concept taxonomy.

The MERGE system (Wasserman 1985) extends the UNIMEM system to deal with multilevel structured objects and binary relations. The hypothesis space used is presented in (Wasserman and Lebowitz 1983). Like UNIMEM, MERGE finds the most specific subsumers of an input example and, for each, computes the similarity between its immediate subsumees and the example. To test for similarity of two structured concepts, MERGE considers all possible matchings between their parts, and counts common literals for each matching. These common literals, for the maximal matching, can be used to build a common subsumer. The decomposition of an object into parts seems to be only for memory organization purposes, and it appears that the π relation is considered to be transitive, as the part decomposition is sometimes ignored when computing a structural match between two concepts. The LABYRINTH (Thompson and Langley 1991) and IMEM (Conklin and Glasgow 1992) systems, surveyed below, remedy this problem.

MERGE is closely related to UNIMEM, and therefore inherits the strengths and weaknesses of that system. Examples are instantiated according to *similarity* with concepts in the network, and the semantics of the system is largely operational. Whereas links in the concept taxonomy usually denote model-theoretic subsumption, it is not clear that the feature removal process preserves the integrity of the concept taxonomy. Like UNIMEM, MERGE cannot employ background knowledge. A strength of MERGE, like UNIMEM, is

[11] In an attribute-value hypothesis space such as that used by UNIMEM, there is a unique least common subsumer for any two concepts.

that concepts can be nondisjoint. This introduces flexibility into the clustering process in that multiple concepts can be used to describe the same example.

2.3.2 Levinson

Levinson (1985) describes a self-organizing information retrieval system for graphs. A database of graph structures is indexed using a taxonomy of discovered structured concepts. Subsumption is computed using a subgraph isomorphism test. The hypothesis space used — connected graphs with labelled edges and vertices — is equivalent to structured concepts with unary and binary predicates. Only one relation is allowed between any two parts. Negation can be expressed using special "negated" edges.[12] Multilevel objects, multinary relations, or background knowledge cannot be expressed, although recent research by Levinson (1994) corrects these problems.

The central tenet of Levinson's work is that a taxonomy provides an efficient indexing mechanism for structured concepts. This indexing mechanism can be used to answer three types of queries. The *generalization* query asks, for some structured concept Q, what concepts are most specific subsumers of Q. The *specialization* query determines the most general subsumees of Q. A *close match* to Q is any concept subsumed by one of Q's most specific subsumers, that is, some concept in the fringe of some element of $\texttt{mss}(Q)$. Chapter 4 of this thesis outlines the properties of a similarity measure for which close match retrieval is sound with respect to similarity.

Levinson's learning algorithm is similar to that of UNIMEM (and MERGE). It is incremental, incorporating individuated structured objects one by one into an evolving concept taxonomy. For each close match of an input example, the system computes a maximal common subgraph (thereby a least common subsumer) and a minimal different subgraph.[13] A minimal different subgraph of two concepts is a most general generalization of one which does not subsume the other. The newly-formed concept is retained and classified (see Section 2.2.5) only if it may improve retrieval efficiency of objects already encountered. Since concepts added early on in an incremental stream of examples may later turn out to be

[12] The absence of an edge in a graph does not mean that two parts are *not* related; if this were so, the subgraph isomorphism test for subsumption would be unsound, and a slightly modified definition of "subgraph" must be employed. See also the footnote to Figure 2.1.

[13] By definition, a *maximal* common subgraph is not a subgraph of any other subgraph. A *maximum* common subgraph has more vertices than any other maximal common subgraph. Levinson uses a heuristic common subgraph algorithm which does not guarantee the computation of either a maximal or a maximum common subgraph (Levinson, personal correspondence). It is also not clear what the learning algorithm does when more than one maximal common subgraph exists.

detrimental to retrieval efficiency, the system has a concept "garbage collection" utility which removes undesirable concepts. Levinson's learning algorithm differs slightly from UNIMEM's in that the latter inspects only the immediate subsumees, while Levinson inspects the complete fringe of a most specific subsumer. Similarity is not used as a basis for discovered concept acquisition.

Levinson introduced some important ideas to structured concept discovery, including its formulation in terms of information retrieval, the restricted yet powerful hypothesis space of labelled graphs, and the criterion for retaining concepts. The approach relies extensively on maximal common subgraph calculation and subsumption computation. Although an efficient topological search is used to compute the most specific subsumers of an example, mechanisms for improving the average-case complexity of subsumption computation must be explored. This problem will be revisited in Section 2.4.

2.3.3 CLUSTER/S

As discussed earlier, the CLUSTER/2 system (Michalski and Stepp 1983b) transforms the abstract conceptual clustering task into a series of supervised learning problems. Unlike all of the systems surveyed above, CLUSTER/2 uses background knowledge. This is expressed as a network of determination rules between attributes. Examples are not completely saturated with the background knowledge, rather, the network directs the partial saturation process by inspecting the set of attributes present in an example. The intuition behind this restricted saturation process is that not all background knowledge will be relevant to a given clustering task (Fisher and Pazzani 1991). CLUSTER/2 uses a complex evaluation function to drive its search over clusterings: this is roughly based on the simplicity of cluster descriptions and the fit of the clustering to the examples.

The CLUSTER/S (Stepp and Michalski 1986) system clusters structured objects, and invokes CLUSTER/2 as a subroutine. Examples are preprocessed by CLUSTER/S — translated into an attribute-value language — where CLUSTER/2 can apply. Essentially this involves finding a common subsumer L of *all* structured examples (using a heuristic beam search), and freezing a subset of variables in each example by substituting unique constants for them. Some variables, not participating in L but connected by relations to variables in L, are also frozen. The resulting variable-free descriptions are then processed by the CLUSTER/2 system. A postprocessing step melts the discovered concepts, substituting variables for constants and converting them back into a structured concept representation.

Despite the important advances made by CLUSTER/S as one of the first structured concept discovery systems to incorporate background knowledge, it has a shortcoming. Since only *one* least common subsumer of all examples is computed, the system cannot be applied to problems where there is no single, meaningful common subsumer. Although CLUSTER/S uses a hypothesis space capable of representing taxonomies of structured concepts, its discovery algorithm is limited.

2.3.4 KBG

Like MERGE, KBG (Knowledge Base Generalizer) (Bisson 1992a) uses a similarity valuation to provoke concept discovery. Highly similar examples are generalized. An example in KBG is a subset of a database of ground atomic facts. Negated literals cannot be expressed. Background knowledge is applied to each example by a complete saturation process. Unlike CLUSTER/S, which controls saturation, KBG computes a deductive fixpoint of each example. Following this process, each example is individuated.[14]

The uniqueness of the KBG learning algorithm is in its computation of similarity between two examples. Unlike MERGE, KBG does not consider all possible matchings between structured concepts. The similarity measure (Bisson 1992b) considers, for each pair of objects in an example, the number of common predicates in which the pair occur as the same argument. Partial relational similarity is taken into account by inductively considering the similarity of other pairs of entities in each common predicate. The result of a similarity computation is a valuation between the two examples, and a valuation between each pair of objects in the examples. The similarity computation is polynomial in the number of literals in the examples.

The learning algorithm of KBG operates as follows. For a set E of examples, KBG builds a square similarity matrix from all similarity valuation computations. Following this process, the system chooses the two[15] most similar examples A and B of E and, using the injective matching derived by the similarity computation, computes a common subsumer G (not necessarily a least common subsumer). The clustering process iterates with the new set $E - \{A, B\} + \{G\}$ of examples, and terminates when this set is a singleton. The resulting concept taxonomy is incomplete: some subsumption relationships are not present. This is because discovered concepts are not classified back into the concept taxonomy. To correct

[14] Bisson refers to "exemplifying of a generalization"; the inverse of individuation. To remain in line with the theoretical foundations section, I prefer to view KBG as working with individuated examples.

[15] KBG will consider common subsumers of more than two examples using a *thresholding* method, but for simplicity here I shall only consider the two-example case. Thresholding also allows discovered concepts to be nondisjoint.

this, a postprocessing step determines subsumption relationships missed during the learning algorithm (Bisson, personal correspondence). This would appear to be an expensive step, since it involves subsumption computations. Note that the learning algorithm is otherwise quite efficient, since least common subsumer and subsumption computations are avoided. The price paid for this efficiency is that overly-general concepts may be created.

2.3.5 LABYRINTH

The LABYRINTH system (Thompson and Langley 1991) is based on the COBWEB conceptual clustering system (Fisher 1987). COBWEB manipulates *probabilistic* concepts. These concepts encompass a set of features, along with a conditional probability of the feature given membership in the concept. Although COBWEB creates a concept hierarchy, links between concepts do not represent logical subsumption. The goal of COBWEB is to create concepts that maximize the expected number of features that can be correctly predicted for instances of the concept. COBWEB passes examples through its hierarchy — a process analogous to the computation of a least common subsumer (Section 2.2.4) — potentially creating new concepts in the process. Concept discovery is guided by a measure which attempts to maximize intracluster similarity and intercluster dissimilarity. When an example is sorted to a node in the hierarchy, it may recursively pass through that node, or trigger concept formation at that level. Two other learning operators are employed to compensate for incremental learning and example presentation order effects: these attempt to correct a suboptimal hierarchy by merging or splitting concepts at a level of the hierarchy.

Concepts in COBWEB are probabilistic, and there is no use for model-theoretic subsumption. This means that the results of discovery can be difficult to interpret (Langley, personal correspondence). All concepts in COBWEB are "disjoint", however, unlike model-theoretic systems, where disjointness is a *semantic* property of the discovered concepts (recall Section 2.2.1), in COBWEB disjointness is a purely *structural* property of a particular concept taxonomy.

LABYRINTH extends the weak hypothesis space of COBWEB, and is capable of expressing multilevel structured objects with binary relations between parts. Of the systems surveyed earlier which use background knowledge, none were "constructive": able to use named discovered concepts as new unary predicates in the first-order language. The LABYRINTH, SUBDUE, and IMEM systems incorporate this important idea. The LABYRINTH system can express negated literals. A concept with n parts will have n probability distributions for the labels of all possible component concepts. For every binary predicate symbol there will be an additional $O(n^2)$ entries in the concept.

LABYRINTH recursively incorporates an individuated structured object in a depth-first fashion; for a level-1 concept this entails classifying each level-0 part, obtaining a level-0 label for each part. For all possible matchings of labels to concept parts, LABYRINTH evaluates the score of the match. This matching process is performed with each child of a node; this determines the roles of parts so that COBWEB's category utility measure can be applied. As an object progresses through the taxonomy, LABYRINTH considers generalizing the features of a node in the taxonomy. This involves computing a common subsumer of two corresponding features and evaluating the resulting concept using an equation designed to avoid overgeneralization. Unlike CLUSTER/S and KBG which preprocess examples using saturation, in LABYRINTH background knowledge is applied during concept formation. Previously discovered concepts can be used as background knowledge. A problem with LABYRINTH which prohibits its use on real applications is the time complexity introduced by its exhaustive matching routine. This will be unacceptable in domains with large structured objects.

2.3.6 IMEM

The IMEM (Image MEMory) system (Conklin and Glasgow 1992) (also see Chapters 3 and 4 of this thesis) incorporates aspects of both LABYRINTH and Levinson's system. Like LABYRINTH, it can represent multilevel structured concepts, negated literals, background knowledge, and can use the names of discovered concepts as new unary predicates. The learning algorithm of IMEM is similar to Levinson's, and it also uses a model-theoretic semantics for concepts. Unlike both systems, IMEM can express multinary relations between parts.

The knowledge representation scheme used by IMEM is somewhat unique. It uses a syntax called an *image term* to represent structured concept descriptions. An image term is formed by associating an image (a set of concept/coordinate pairs) with a set of relation identifiers. To each relation identifier there corresponds a Boolean-valued function mapping a tuple of image components to values. Relations are computed directly from the image using functions. The space complexity of a structured concept description is therefore linear rather than polynomial in the number of concept variables. Implicitly present in a structured concept description, for a given relation, are *all* tuples for which the associated function is defined. The drawback of this complete representation is that only coarse-grained truncation is available as a generalization operator. IMEM uses two truncation operators: removing a part from an image (i.e., removing all literals that involve a given variable), or removal of a relation from a relation set (i.e., removing all literals with a given predicate symbol). Although a complete representation is weaker than an arbitrary graph representation, as used by Levinson for example, it is comprehensible and has been shown to be remarkably effective in characterizing many different types of spatial concepts in structural chemistry domains. It should be noted that the subsumption problem in this representation remains NP-complete (see Chapter 3).

Of the systems surveyed above, IMEM is most closely related to Levinson's. The learning algorithm differs slightly in that IMEM, like UNIMEM, inspects only the immediate subsumees of each most specific subsumer. Also IMEM, unlike Levinson's system, does not compute "greatest different" subsumers. IMEM was designed as a self-organizing information retrieval system for images. It supports generalization, specialization, and close match queries. Like Levinson's system, the concept taxonomy formed is sound and complete with respect to subsumption. Whereas Levinson uses a heuristic designed to speed graph retrieval, IMEM uses a heuristic designed to maximize intercluster dissimilarity and intracluster similarity. These measures are in line with classical information retrieval goals of balancing recall and precision (Salton and McGill 1983).

2.3.7 SUBDUE

The systems presented so far have performed generalization from two or more examples. By contrast, the SUBDUE system of Cook and Holder (1994) analyzes a single structured object to detect recurrent connected subgraphs. The hypothesis space appears to be equivalent to that of Levinson's system, although an "inexact" subgraph isomorphism test is used to validate subsumption relationships. A minimal length encoding measure is used to quantify the "interestingness" of a discovered structured concept. This measure coaxes the system away from concepts which are too general or too specific. Overly general concepts will require much additional information to reconstruct the original graph, while overly specific concepts will be too complex. In neither case will the original structured object be compressed well by the discovered concept.

The fundamental limitation of this method is that, by definition, substructures which do not recur within a *single* structured object will not offer any compression and will not be proposed by SUBDUE. Although this is an unrealistic restriction for general machine learning, future work could profitably combine inter-example methods with SUBDUE's intra-example discovery method.

In SUBDUE, the components of a concept can refer to a previously discovered concept. Thus, the system is capable of building a concept taxonomy of multilevel structured concepts. Once a substructure is identified in an example, that example is parsed and re-expressed using the substructure. This is in contrast to the LABYRINTH and IMEM systems which assume that examples are parsed in compatible ways. The parsing process of SUBDUE appears to have unspecified semantic implications, since new π relations, and new predicates expressing relations between higher-level objects must be introduced into the underlying first-order language.

2.3.8 Miscellaneous systems

This chapter has surveyed several major structured concept discovery systems in detail. Each system represented some advancement over previous systems in the technology of structured concept formation. Several general-purpose discovery systems were omitted. Although secondary to its main goal of concept learning, Winston's (1975) ARCH system includes a "grouping" method which discovers concepts comprising, for example, sequences of parts related by a common relation. These structured concepts are used to deepen the hierarchical description of an object. The RINCON system (Wogulis and Langley 1989) attempts to improve the efficiency of the relational concept instantiation problem by creating and retaining "intermediate" concepts. The DMP system (Parsons 1989) discovers structured concepts within a database of ground facts. The CAFE system of Handa et al. (1994) has made minor extensions to LABYRINTH's hypothesis space and concept evaluation function. It allows the parts of concepts to be "context-sensitive" by including a description of the inverse of the π relation. Mineau and Godin (1994) describe a conceptual clustering system for restricted structured objects. In contrast to other systems, their system creates a *complete* concept lattice: every subsumer of the examples is formed and made explicit. The IDS system (Nordhausen and Langley 1990) has a structured concept discovery component very similar to MERGE. Yoshida et al. (1993) describe CLIP, an intra-example discovery system similar to the SUBDUE system.

There has been a longstanding interest in the computer vision and pattern recognition communities in the use of structured object representations and relational models. Structured objects and concepts are a powerful way to describe the necessary and sufficient features and relations of a visual object. Most of the machine learning work in this area has been in supervised concept learning (Winston 1975; Connell and Brady 1987), although researchers have also considered the problem of organizing relational models for

Name	Relations	Level	Theory	Semantics	Incr	Nd	SA
MERGE (1985)	binary	any	no	combined	yes	yes	similarity
Levinson (1985)	binary	1	no	model	yes	yes	cost
CLUSTER/S (1986)	multinary	n/a	yes	model	no	no	seed
KBG (1992)	multinary	n/a	yes	model	no	yes	similarity
LABYRINTH (1991)	binary	any	yes	probability	yes	no	probability
IMEM (1992)	multinary	any	yes	model	yes	yes	similarity
SUBDUE (1994)	binary	any	no	similarity	no	yes	compression

Table 2.3: A comparison of several structured concept discovery systems. Incr: incremental, Nd: nondisjoint, SA: selective acquisition method.

efficient search. A common solution is to perform a numerical clustering procedure on a database of structured objects (Sengupta and Boyer 1993; Shapiro and Haralick 1982; Segen 1990). Structured concepts will then have a similarity semantics, and links in the cluster taxonomy will not necessarily reflect subsumption. Dey et al. (1993), by contrast, do use a model-theoretic notion of subsumption. Conklin (1993) shows how the IMEM system applies to the relational model organization problem. In a somewhat related field, research in retrieval of structured pictures from image databases has used a concept abstraction or generalization operator (Chang and Liu 1984).

The notion of a structured concept reappears frequently in many different problem domains. The structural chemistry community, for example, has had a longstanding interest in graph structures. An early system, Meta-DENDRAL (Cohen and Feigenbaum 1982), created a taxonomy of molecular graphs for use in mass spectrum analysis. The system of Okada and Wipke (1989) indexes a database of molecular graphs using a taxonomy of discovered maximal common subgraphs. Okada (1993) shows how this taxonomy can be used for analogical inference. Structurally similar molecules are retrieved by a most specific subsumer computation, and properties of molecules are inferred from similar instances. Wilcox and Levinson (1986) show how Levinson's system can be applied to a database of molecular structures for efficient information retrieval. LABYRINTH has been applied to the problem of DNA promoter sequence recognition (Thompson et al. 1991). IMEM has been applied to the discovery of small molecule conformation classes (Conklin et al. 1993a), and to protein motif discovery (Conklin et al. 1994).

2.3.9 Discussion

The previous section has surveyed a number of structured concept discovery systems. These were reviewed with particular attention to the *knowledge level*, that is, to the expressive power of the hypothesis space and the meaning of concept subsumption and not to the specific symbol structures used. Table 2.3 presents these systems according to several dimensions. The "semantics" column refers to the underlying semantic theory of a concept. In the case of probabilistic concepts (LABYRINTH), concepts are interpreted as conditional probability distributions over literals. This interpretation has the undesirable property of not supporting a clear definition of subsumption. In the case of model-theoretic semantics, the most common form, concepts have an extension (recall Section 2.2.1), and subsumption is defined as a subset relation between extensions in all possible interpretations of the concept. In these systems, unlike probabilistic systems, links between concepts in a taxonomy have a sound denotation with respect to subsumption. Finally, the MERGE system uses a confusing *combined* semantics, where links in the concept taxonomy may denote model-theoretic subsumption, but instantiation is performed using a similarity comparison with a concept. The reasons for this unfortunate development stemmed from the early UNIMEM system, which was more concerned with structural memory organization than with sound semantic principles.

The "Relations", "Level", and "Theory" columns of Table 2.3 attempt to give an idea of the expressive power of a hypothesis space. For example, the LABYRINTH system employs background knowledge, supports multilevel structured concepts, and allows only binary relations between parts to be expressed. Multinary relations can be expressed as compositions of binary relations, but this becomes tedious with large structured objects. The CLUSTER/S and KBG systems do not represent structured objects as multilevel objects: rather, they support arbitrary conjunctive expressions. While their hypothesis space is strictly more expressive than that of other systems, they will suffer from severe computational difficulties when subsumption between large structured concepts must be performed. In systems which employ a true multilevel organization for structured concepts, objects can be organized so that a less expensive subsumption computation is faced (recall Section 2.2.3).

The "Nd" column of Table 2.3 states whether the classes produced by the discovery algorithm are nondisjoint. As stated earlier, the ability to produce nondisjoint categories is one of the differences between a conceptual clustering and a more flexible concept discovery system. In the two systems that produce disjoint classes, a distinction must be made as to whether classes are *structurally* disjoint (LABRYRINTH), or whether they are also *semantically* disjoint (CLUSTER/S).

The "Incr" column of Table 2.3 denotes whether learning algorithm is incremental. Nonincremental systems, such as CLUSTER/S and KBG, will have serious difficulties in coping with large numbers of examples. Extending CLUSTER/S to be incremental would be difficult due to its semantic disjointness requirement on discovered concepts. As pointed out by Bisson (1992b), KBG would be fairly easy to coax into an incremental mode.

Finally, the "Sa" column of Table 2.3 refers to the manner in which concepts are selectively acquired (Markovitch and Scott 1993) or discovered. LABYRINTH is derived from COBWEB and uses a predictiveness measure to judge the expected worth of a potential concept. Concepts formed by the CLUSTER/S system wholly depend on the initial "seed" objects selected from the pool of examples. Levinson (1985) employs a sophisticated heuristic which measures whether the new concept will lead to more efficient retrieval of existing examples. The most common selective acquisition mechanism is to create a concept when a new example is *similar* to a previous example. This similarity function can be purely structural (IMEM), based also on similarity of unary attributes (MERGE), or can be pseudo-structural (KBG).

2.4 Conclusions

This chapter has presented a unified theory of structured concept discovery, and has discussed in detail several operational machine learning systems in this paradigm. Strengths and weaknesses of each system were outlined. This section presents some potentially important future research directions for the field. Haussler (1989) attempted a similar task; however, since that paper was mainly concerned with hypothesis space theory and not with operational systems, several issues were not anticipated. Below I critically review a few of Haussler's proposals, focusing on if and how they have been addressed by current state-of-the-art systems. I then propose my own new set of critical future research directions.

One of Haussler's proposals was to replace the $\exists^*$ quantifier by the $\exists$ quantifier. For instantiation and subsumption computations, this has the effect that a part of a structured object can instantiate more than one variable of a structured concept. In computer vision terminology, this means considering a *relational homomorphism* rather than a *relational monomorphism* between concept variables (Vosselman 1992; Haralick and Shapiro 1993). In terms of θ-subsumption computation, it means searching over the much larger space of both injective and non-injective substitutions. This tradeoff with added search complexity must be justified with convincing examples. In computer vision and chemistry applications, for example, monomorphisms are commonplace. None of the systems surveyed above use general relational homomorphisms.

A second proposal of Haussler is to increase hypothesis space power by including disjunction as a basic operator. Although this sounds simple enough, there are several ways disjunction might be incorporated. One is to allow arbitrary disjunctive normal form expressions as concept descriptions. As mentioned earlier, the field of inductive logic programming uses such a hypothesis space. A simpler idea is to allow disjunction among unary predicates over the same variable. Something similar to disjunction can already be expressed by any conjunctive hypothesis space using background knowledge. A common subsumer of two attributes will include in its extension all objects with either attribute. This mechanism is not quite equivalent to disjunction as the common subsumer may be more general than the disjunction of the two attributes. This points to a problem when incorporating proper attribute disjunction: when generalization is to be performed, the least common subsumer can tend to be overly complex and specific. It is necessary to introduce additional preference criteria for candidate generalizations. It appears that this is a classical complexity versus fit problem, and Conklin and Witten's (1994) complexity-based induction approach may be useful here.

Another topic that came up in Haussler's paper was relation subsumption. For example, the relation "close" subsumes the relation "very close". In chemistry, bonds of lower orders can be viewed as subsuming those of higher orders (Okada and Wipke 1989). The use of relation subsumption is also related to the use of more powerful types of background knowledge, e.g., definite clause programs as used by the inductive logic programming community. The KBG system uses a taxonomy of relations that is applied during the saturation process.

Finally, Haussler refers to the need to consider multilevel structured objects. Clearly this proposal has been satisfied and implemented in the LABYRINTH and the IMEM structured concept discovery systems.

I shall divide my proposals into two types; those concerned with hypothesis space power and those concerned with memory organization, retrieval, and system issues.

A recurrent issue in knowledge representation is finding a good *ontology*; predicates used to describe objects in the world. Structured concept discovery systems face the same problems. For example, personal experience with the IMEM system has indicated that "correct" concepts emerge almost immediately from examples when the appropriate set of relations are used. This indicates that a topic of great importance is the investigation of *constructive* learning, whereby a concept discovery system has the ability to propose, represent, and evaluate new relations to be used in the first-order language. Crandell and Smith (1983), for example, do a relational clustering to discover an ontology for molecules.

The relational discovery task is difficult because of the enormous space of new relations that can be constructed from an initial language. On a brighter note, however, all structured concept discovery systems perform a constructive discovery of concepts — essentially unary relations — and it seems like much of the theory is already in place to cluster relational examples.

An intriguing extension to a hypothesis space is to allow for *cyclic* declarations, where the name of a concept reappears in its description. Consider, for example, the concept of a crystal, which is a basic structural motif repeated ad infinitum. Such a concept could have the form

$$X \stackrel{\mathrm{def}}{=} M \vee \ldots X \ldots$$

where M is the basic motif. The field of inductive logic programming has already investigated recursive relational concepts; many of these principles could apply to structured concept discovery. Cyclic concepts are also studied by the knowledge representation community (Nebel 1991). In pragmatic terms, a cyclic structured concept would be a compact representation of what would otherwise require arbitrarily many separate concepts of various specificity. None of the surveyed systems purportedly allow for cyclic definitions, in fact, few (e.g., IMEM and LABYRINTH) allow for a discovered concept to be expressed in terms of another previously discovered one.

The computational advantages of using a multilevel description was pointed out earlier. However, penalties are incurred by moving to a multilevel representation. First, there is no "correct", and sometimes no intuitive, way to decompose an object into subgroups. This is reminiscent of the general segmentation problem of computer vision and the "perceptual grouping" task (Marr 1982; Pentland 1986). Multilevel concept instantiation will require that a structured object be parsed in a way compatible with the concept. Also, when an object is parsed, "cousin" relationships are awkward to maintain. The latter problem is a necessary tradeoff that must be accepted; the former could be addressed by a rigorous and extended theory for the π relation.

A topic that must be explored in more detail is the evaluation of discovered concepts during selective acquisition and retention. Recall that a selective acquisition method decides, at the time a concept is created, whether it should be incorporated into the background knowledge. A selective retention mechanism decides, in the future, whether to discard a concept that does not appear to be "useful". Thus any scheme for evaluation concepts must be based on a theory of pragmatics or learning performance. For the concept learning task, the goal is to improve the accuracy of classification. A selective acquisition measure should aim to learn concepts that may have higher accuracy, whereas a selective retention mechanism would discard concept descriptions with low accuracy.

For general concept discovery, unfortunately, evaluation is not so clear. Many different, and equally valid, retention measures can be proposed. One can wish that known concepts are "rediscovered", or that discovered concepts are somehow interesting, unanticipated, or intuitively plausible to a human expert. This was the evaluation mechanism used by Conklin et al. (1993a) in their discovery of pyranose molecule configurations. A stronger, less subjective criterion is employed by Levinson's system, where a discovered concept must improve retrieval efficiency. LABYRINTH tries to maximize the inferential predictiveness of features. IMEM aims for a balanced concept taxonomy for information retrieval purposes. The CLUSTER/S system employs a detailed evaluation function for proposed clusterings, but is not based on any theory of pragmatics. To summarize this discussion, there is no "correct" concept evaluation criterion, and this points to a need for a stronger theory of pragmatics for concept discovery.

The final future research topic I discuss is that of knowledge base organization and structured concept retrieval as borne out by the concept classification process (Section 2.2.5). There is no reason to suppose that the number of discovered structured concepts in an application will be small, and if many or all of these are to be retained it is necessary to employ efficient implementation techniques. It seems clear that the size of the concept descriptions must be kept to a minimum. Lebowitz' (1986a) idea is to employ "inheritance" in the frame or object-oriented programming sense. For propositional calculus and frame-based representations, inheritance is deterministic. For structured concepts it is necessary to specify, in the form of a graph embedding, how the parent and child concept are structurally related. Ellis (1991) offers an intriguing approach, which is to specify a sequence of transformation rules which rewrites the parent concept into equivalence with the child. In both of these approaches, in a deep concept taxonomy much of the structure of lower concepts can be "inherited" from above, perhaps resulting in a significant savings in memory.

Levinson (1992) presents a technique for reducing the average-case complexity of structured concept subsumption checking during classification. The technique assigns properties to nodes in a graph, thereby making them more specific and restricting the number of possible concepts they can subsume. A taxonomy of these "node descriptors" is used to index database graphs. Soundness of the technique can only be guaranteed by a subsequent expensive subsumption test, but the algorithm avoids this test. A similar method has been used in chemical structure retrieval systems (Nagy et al. 1988) which must avoid as many subgraph isomorphism tests as possible.

Barrow et al. (1972) make a category-theoretic connection for subsumption and relational structures. The nodes of this category are relational structures and the arrows are relational monomorphisms. The presence of the cached monomorphisms determines subsumption relationships between concepts in a taxonomy and a new query concept. Barrow et al. present a technique called "hierarchical synthesis" for computing subsumption using cached monomorphisms. Their concept taxonomies are hand-coded, and the authors state that "the remaining problem of this approach is that of setting up the modules [concepts] and network [concept taxonomy] automatically in a 'learning phase'." Clearly, this "remaining problem" can now be addressed by the new generation of structured concept discovery systems. Levinson's (1994) "Method V" retrieval algorithm uses a very similar arrow caching technique. All of these methods are fundamentally different from Ellis' (1991) technique, as the category technique requires that *all* injective mappings between parent and child concept must be cached, whereas Ellis' approach only stores one set of transformations.

Finally, although the picture is improving with recent forays into chemistry, vision, and database applications, there are many other interesting domains where data are described by structured objects and the field must be acutely concerned with moving away from toy "blocks-world" problems. Concept discovery systems must be able to deal with very large amounts of data. This requires a real concern for efficient implementation techniques. There is much research to be done, but the future looks bright and it should be possible to retain the high expressive power and generality of structured concept discovery systems while achieving results in important specific applications.

Chapter 3

A Spatial Description Logic

The previous chapter introduced the idea of structured concepts — configurations of generic parts and their interrelationships — and showed with several examples how they are created and managed by existing concept discovery systems. This chapter precisely defines a structured concept hypothesis space for a machine discovery system which will be presented in the next chapter. The hypothesis space, a Spatial Description Logic, draws on research in *direct* knowledge representations. The main contribution of this chapter is to integrate direct knowledge representations with the ideas of taxonomy, subsumption, and classification as studied by researchers in description logics. The result is a logic which provides concise, comprehensible descriptions for structured concepts.

In a direct knowledge representation, functions are used to inspect spatial configurations of symbols for object interrelationships. Direct representations are an attractive representation for knowledge discovery, as much existing structured data (e.g., geographic data, visual sensor data, CAD data, pictorial data) are already in direct form. In particular, objects in molecular databases are given in direct form, and not as a listing of logical interrelationships. The latter are inferred from molecules as needed. Dealing with data in their direct form allows a logical discovery system to bypass the issue of translation into logical formulae (e.g., into Prolog facts). This provides flexibility, as configurations can be re-interpreted in many ways depending on the particular needs of a problem and the inspection functions used.

This chapter has several subgoals. One is to introduce, in a rigorous fashion, the idea of direct knowledge representation. Description logics are introduced, and a *spatial* description logic is crafted. This logic incorporates a direct representation for structured concepts. I will discuss how subsumption in the spatial description logic is computed, and how classification is performed. Finally, I will propose some extensions to the logic.

3.1 Reasoning directly from models

A topic of steady interest in the field of knowledge representation is that of using representations which are in some sense direct models of the world. Reasoning in such representations is performed by "inspecting" or by performing "measurements" on the model. Model-based representations appear under a variety of names, including analogical (Sloman 1971; Meyers and Konolige 1992), diagrammatic (Larkin and Simon 1987), homomorphic (Barwise and Etchemendy 1992), direct (Hayes 1985b), visual (Harel 1988), spatial (Glasgow 1993), iconic (Tanimoto 1976), and so on. Much of the interest in model-based representation stems from the hypothesis that humans do much inference-making from mental images. These internal imagistic or "depictive" representations are often contrasted with their propositional or "descriptive" counterpart (Larkin and Simon 1987).

It is sometimes claimed that inference-making in model-based representations is more efficient than in equivalent, non-model-based representations (Larkin and Simon 1987). Hayes (1985b) has argued that this claim is fallacious. To explore this issue, in this section I shall consider the following restricted yet basic and fundamentally important inference problem: given a set of objects (represented by constant symbols), and a relation (represented by a predicate symbol), determine whether that relation holds among those objects. This sort of inference can certainly be made in a non-model-based representation. For example, we could axiomatize facts and properties of the relation and use a Prolog technology theorem prover to make the inference. However, in most model-based representations the validity of the relation can be computed in *constant time* in the number of objects represented.

As noted by Levesque (1986) and Selman (1988), this constant-time relation validation is a property of all "vivid" knowledge representations: where facts are explicitly represented and all disjunction is removed. It is *not* exclusively a property of model-based depictive representations. Although such representations are, at times, useful *implementations* of models, at the computational level, at least for the simple inference problem referred to above, there is no basis for distinction. For example, consider a finitary representation of a Herbrand model of a logic program. Clearly, such a representation is not depictive, but it does possess the constant time relation computation property for vividness (e.g., consider hashing on every element of the model). As Hayes (1985b) has pointed out, depictive representations may be implemented in a descriptive fashion, and it is fallacious to make general statements about their computational efficiency.

3.1.1 Direct representations

A *direct* representation is a *secondary* (Funt 1980) or *homomorphic* (Hayes 1985b) model for a world. A feature common to all direct representations is that inferences are made using *functions* which operate on some secondary model, rather than by proof procedures which construct a primary model. These functions must be efficiently computable, else a direct representation is needless overhead. If a direct representation is a subformalism in a hybrid system, one may "jump out" of the secondary model, into the primary model to make inferences (Meyers and Konolige 1992). Thus if one takes the view that direct representations, while making sound inferences, need not be complete due to the presence of a primary model, there are few reasons for not using such a representation.

Direct representations are partially motivated by the fact that a complete extensional description of a predicate, including the negative literals, is space–expensive. Systems that make the closed world assumption could partially ameliorate this problem by using, for example, the negation–as–failure rule (Reiter 1985) and retaining only positive ground literals. However, a compact definition of positive information may not exist.

The main components of a direct representation are

- A *medium*: a structure into which symbols are placed. A medium comprises a finite number of non-decomposable regions or *pixels*. An instantiated medium is called a *configuration*.
- An *index* or *coordinate set*: a set that indexes pixels or regions in the medium.
- *Retrieval processes*: Given a symbol and a configuration, find its index in time constant with respect to the number of symbols in the configuration.
- *Inference functions*: Boolean-valued functions which accept a configuration, and a set of symbols as parameters, and detect the validity of a relation in time constant with respect to the number of symbols in the configuration.

Direct representations are homomorphic to a primary model, in the following formal sense. Suppose that in an intended primary model M, relation r holds between the objects denoted by the constants $o_1, \ldots, o_n$. This is denoted as

$$M \models r(o_1, \ldots, o_n) \tag{3.1}$$

Let C be a secondary model for M, r' be the function which computes the relation r, and

$o_1, \ldots, o_n$ be object constants. In a direct representation, the following rule holds:

$$r'(C, o_1, \ldots, o_n) \quad \Rightarrow \quad M \models r(o_1, \ldots, o_n) \tag{3.2}$$

that is, if the relation holds between the objects in the configuration, then it is valid in the associated primary model. If we enforce that (3.2) is valid for all objects in the configuration C, we say that it *preserves* the relation r. In such a configuration, when objects are related, sound inferences can be made without recourse to theorem proving procedures. Note that direct representations are not necessarily complete in that inferences about negated literals cannot be made. If a relation is not displayed in the configuration we cannot say, without other evidence, that it is or is not satisfied by the primary model. In fact, direct representations can be rather trivial, since a configuration in which *nothing* is related satisfies (3.2). A more powerful class of representations, called *strongly direct* (Hayes 1985b), enforce that the implication in (3.2) is bidirectional. Inferences about negated literals can be made in a strongly direct representation.

Expression 3.2 is closely related to the idea of *predicate evaluation* in automated theorem proving (Winston 1984). For example, consider evaluating the term `plus(2,2)` using a theorem prover. While this can be done by applying the Peano axioms of arithmetic, a more efficient way is to attach a procedure to the symbol `plus`, and evaluate the expression at the implementation language level. Similar ideas are *procedural attachment* in frame systems (Winston 1984), and the more recent and general notion of *universal attachment* (Meyers 1994).

Examples of direct (and strongly direct) relation-preserving representations abound: here is a handful of examples. A simple example is the temporal string (Anderson 1983); the medium is a one-dimensional array structure. The index set is the natural numbers. Retrieval is done by array indexing. The function `before` computes the temporal precedence relation by testing the indices of two items in the array. A straightforward generalization of the one-dimensional temporal string is the two-dimensional *symbolic picture* (Chang et al. 1987), which preserves the `left` and `above` relationships. Another example is the concept taxonomy, implemented as described in (Agrawal et al. 1989). The medium is one-dimensional Euclidean space, and the index set is interval ranges as represented by pairs of real numbers. The function `subsumes` computes the subsumption relation between concept names by testing for interval containment. Another example is the molecular database (Allen et al. 1991a). The medium is three–dimensional Euclidean space, and indices are (quantized) real numbers. The function `bonded` determines whether two atoms are bonded by testing whether their Euclidean distance is within a certain range (also see Chapter 5 for many more examples from structural chemistry). A final, related example is the image database (Klinger and Pizano 1989): the medium is three–dimensional Euclidean space, and indices are descriptions of polygons. Spatial relationships are computed by functions which operate on polygons: for example, the function `above` computes whether one polygon is above another in an image by testing for relative position and the absence of spatial overlap.

When appropriate, direct representations can provide a compact representation of a primary model. A configuration with n symbols will require a constant $O(n)$ indices, independently of the number of relations preserved, whereas a non-direct vivid representation (e.g., a graph) may require $O(n^2)$ object constant pairs for every binary relation axiomatized. However, it is entirely possible that some inference functions, which "inspect" a configuration, are more expensive to compute than homomorphic functions in some other sort of vivid representation. While constant-time relation validation is a property of all vivid representations, one of the main motivations for using direct representations should be their conciseness.

3.2 A Spatial Description Logic

The preceding section reviewed Hayes' class of *direct* knowledge representations, and gave several examples of their use in practical knowledge representation tasks. In a direct representation scheme, symbols are placed into a medium, which can be addressed by location. In all examples presented, the symbols were object constants, and a configuration of symbols is interpreted as a set of ground literals. This is quite restrictive, and in a critique of direct representations Goebel (1993) states (italics added):

> The simple notion of "ground" or specific linguistic statements versus generalizations that employ variables (e.g., universals) begs the question of whether there exists a concept such as a *depictive generalization.* [...] Assuming the importance of depictive reasoning, one would certainly want to be able to use "generic" images to create specific image instances.

This section describes such a representation, called $\mathcal{SDL}$, a *spatial description logic* in which "depictive generalizations" can be expressed. The interpretation of a configuration in this scheme is a set of objects, that is, a concept. The semantics of a configuration is given by translating it to a structured concept description in first order logic (see Chapter 2).[1]

[1] The reader may have noticed that Figure 2.2 employed a direct representation for structured concepts.

$\langle Terminology \rangle$ ::= $\langle Sentence \rangle^*$	
$\langle Sentence \rangle$ ::=	`defconcept` $\langle ConceptName \rangle$ $\langle ConceptTerm \rangle$ |
	`defprimconcept` $\langle PrimConceptName \rangle$ $\langle PrimConceptTerm \rangle$

Table 3.1: The two concept-declaring statements of a generic description logic

3.2.1 Description logics

Research in knowledge representation theory has led to the development of a number of formal systems for representing and reasoning about concepts. Broadly grouped under this area of *taxonomic logics* are frame systems (Minsky 1975), semantic networks (Sowa 1991), terminological or description logics (Nebel 1990; Baader and Hollunder 1991), and conceptual graphs (Sowa 1984). These logics have the common property and distinction of being able to explicitly encode or infer the fact that one concept is more general than (henceforth *subsumes*) another, that is, the fact that all instances of the more specialized concept are necessarily instances of the more general one.

Description logics arose out of early attacks on frame theory — Hayes' (1985a) argument that frames were really nothing more than a convenient notation for a restricted subset of first-order logic, and Woods' (1985) and Brachman's (1983) criticism of the lax semantics given to links between frames. It was argued that the semantics of a frame system is largely *operational*, depending on the particular programs developed to manipulate the frames. In general, description logics model and extend the portion of frame theory that can be given a formal denotational semantics. They retain from frame theory the idea of an "object-based representation," in that information about an individual object or concept can be clustered together in one syntactic bundle. The idea of a *description* in logic, however, is standard (see Charniak and McDermott 1985, Section 6.5.2, for an overview) and description logics can also be viewed as restricted yet tractable subsets of first order logic.

There are numerous description logics reported in the literature, including KRIS (Baader and Hollunder 1991), BACK (Nebel 1988), LOOM (MacGregor 1991), CLASSIC (Borgida et al. 1989). They all inherit their basic representational constructs from KL-ONE (Brachman and Schmolze 1985; Woods and Schmolze 1992), one of the first description logics. Description logics like KL-ONE build *defined concepts* and *primitive concepts* from other defined concepts. Defined concepts are associated with necessary and sufficient conditions for concept membership: primitive concepts have only necessary conditions stated (see Table 3.1). A *terminology* $\mathcal{T}$ is a sequence of concept declarations.

3.2.2 A Spatial Description Logic

Concept terms and primitive concept terms are recursively built using various *concept constructors*: these vary between description logics, and indeed, characterize a particular description logic. For example, most description logics have an `and` constructor, where $(\texttt{and}\, C\; D)$ denotes the concept $\lambda\texttt{x}.C(\texttt{x}) \wedge D(\texttt{x})$. Most description logics have a method for expressing relations between objects; these so-called *roles* are binary relations. For example, most description logics have an `all` constructor, where $(\texttt{all}\; r\; C)$ denotes the concept $\lambda\texttt{x}.\forall\texttt{y}\; r(\texttt{x},\texttt{y}) \Rightarrow C(\texttt{y})$. The spatial description logic $\mathcal{SDL}$, true to the spirit of direct representations, uses functions to compute relations of any arity from a configuration of symbols. The basic idea is to associate a set of relation identifiers with a configuration, into which identifiers are placed. These identifiers can be other concept names, and hence $\mathcal{SDL}$ can express multilevel structured concepts.

In order to apply the ideas of direct representations of ground facts (as discussed in Section 3.1) to direct representations of structured concepts, a small theoretical problem needed to be solved. In a direct representation of ground facts, we must make the (reasonable) restriction that every object has a unique name and location. If this assumption is not made, the function r' in (3.2) could not identify which objects, out of a set of duplicates, are intended. The unique name assumption cannot be made with direct representations of structured concepts. There can be many occurrences of the same concept name, and these should denote different instantiations of that concept. This minor technical difficulty is solved by the representation described below, which considers the arguments of the direct inference functions to be *components* or term/coordinate pairs.

Syntax

The syntax of $\mathcal{SDL}$ is straightforward. There are two types of concept terms that can be expressed: *image terms* and *primitive terms*. The latter are conjunctions of primitive concept names. Image terms are used to represent structured concepts and are formed by associating a *symbolic image* with a set of relation identifiers.

A symbolic image is described by a spatial data structure comprising a set of concept terms with their coordinates in multidimensional space. Informally, a symbolic image is a set of *components*, or term/coordinate pairs.[2] It is an extension and generalization of the

[2] Two functions `first` and `second` will be used to access the terms and coordinates of an image component. If $C = (a\; b)$, $\texttt{first}(C) = a$ and $\texttt{second}(C) = b$.

$$
\begin{aligned}
\langle \textit{ConceptTerm} \rangle ::= {} & \texttt{any} \mid \\
& \langle \textit{ConceptName} \rangle \mid \\
& \langle \textit{PrimConceptTerm} \rangle \mid \\
& \langle \textit{ImageTerm} \rangle \\
\langle \textit{PrimConceptTerm} \rangle ::= {} & \texttt{any} \mid \\
& (\texttt{and}\ \langle \textit{PrimConceptName} \rangle^*) \\
\langle \textit{ImageTerm} \rangle ::= {} & (\texttt{image}\ \langle \textit{Image} \rangle\ \langle \textit{Relations} \rangle) \\
\langle \textit{Relations} \rangle ::= {} & \langle \textit{Identifier} \rangle^*
\end{aligned}
$$

Table 3.2: The syntax of $\mathcal{SDL}$, a Spatial Description Logic

2D symbolic pictures of Chang et al. (1987). Formally, the abstract data type *Image* is given by the following signature:

$$
\begin{aligned}
Component = ConceptTerm \times Coordinate \\
\texttt{empty} : \emptyset \rightarrow Image \\
\texttt{put} : Image \times Component \rightarrow Image \\
\texttt{delete} : Image \times Component \rightarrow Image
\end{aligned}
$$

Many other operations on images can be defined using these constructors: for example, a `replace` operation which changes a component, a `move` operation which transfers a single part to a new location, an `overlay` operation which "merges" two images, an `extract` operation which retains only those components of an image lying in a given region of space, or an `invert` operation which changes the sign of all coordinates:

$$
\begin{aligned}
\texttt{replace} : Image \times Component \times Component \rightarrow Image \\
\texttt{move} : Image \times Component \times Coordinate \rightarrow Image \\
\texttt{overlay} : Image \times Image \rightarrow Image \\
\texttt{extract} : Image \times Region \rightarrow Image \\
\texttt{invert} : Image \rightarrow Image
\end{aligned}
$$

The *canonical form* of a symbolic image can be given by a sequence of `put` operations. This canonical form will be abbreviated below simply as a set of components.[3]

A symbolic image is a spatial configuration of symbols, or a sort of *graphical data*

[3]This image data type will allow two concept terms to be placed at the same coordinate. However, if the concept terms are syntactically equivalent, then the canonical form, being a set and not a list, will omit one of them. This scenario is not encountered in molecular databases. It could be corrected by giving each component a unique identifying number, but this would unnecessarily confuse the presentation here.

type (Mallgren 1982) or *image data structure* (Tanimoto 1980), and by itself has a low-level algebraic semantics (Goguen et al. 1978). For $\mathcal{SDL}$, we are interested in a higher level semantics, as addressed by the questions "*what* is in the image?," "what things are *related*?," and "*how* are they related?". This is accomplished by associating relations with symbolic images. They transform a "syntactic" symbolic image into a "semantic" structured concept.

An *image term* is formed by associating a symbolic image with a set of relation identifiers that are preserved by the image. These relations are analogous to description logic roles, except that they are computed by functions which directly manipulate the symbolic image data structure. Functions that operate on symbolic images take a tuple of components as arguments; formally,

$$Component^k \rightarrow \{\texttt{true}, \texttt{false}\}$$

is the function type for an k–ary relation. The $\mathcal{SDL}$ representation language provides a direct rather than a sentential representation of structured concepts. Direct representations have a number of nice properties for structured concept representation, including simple and elegant rules for structural subsumption (Section 3.2.3) and generalization (next chapter) compactness of description, and an exact structural and geometrical correspondence with objects from the domain.

The functions which compute relations from images in $\mathcal{SDL}$ are defined at the *implementation language level.* For example, if $\mathcal{SDL}$ were to be implemented in the C programming language, these would be C functions. Thus the relations are opaque to $\mathcal{SDL}$, that is, they cannot be reasoned about or manipulated. The relations, in this regard, are similar to the *test concepts* provided by the CLASSIC description logic (Borgida et al. 1989). It must be enforced that relations do not change the global state of the underlying $\mathcal{SDL}$ interpreter.

Table 3.2 gives a BNF grammar for $\mathcal{SDL}$. I now turn to the task of developing a formal semantics for the logic.

Denotational semantics

As mentioned above, description logics can be viewed as compact notations for subsets of first order logic. The denotational semantics of a description logic can be given either *directly* — using extension functions which map concept terms directly to subsets of a domain of interpretation (Nebel 1990) — or by *equivalence* — translating a terminology to an equivalent set of sentences in first order logic (Royer and Quantz 1993). I have chosen the latter alternative, since the semantics for first order logic has already been presented in Chapter 2. It is important to keep in mind that $\mathcal{SDL}$ reasons directly with image terms and descriptions and not with translated forms. The following section on subsumption validation will make this point clearer.

To translate an $\mathcal{SDL}$ terminology, I use expressions of the form $\overline{X} = X'$. These should be read: "the translation of $\mathcal{SDL}$ expression X is the well-formed-formula X' in first order logic". The translation function is defined inductively as follows:

$$\overline{\texttt{defconcept}\,\texttt{N}\,\texttt{C}} = \forall \texttt{x}\, \overline{N}(\texttt{x}) \Leftrightarrow \overline{C}(\texttt{x})$$

$$\overline{\texttt{defprimconcept}\,\texttt{N}\,\texttt{C}} = \forall \texttt{x}\, \overline{N}(\texttt{x}) \Rightarrow \overline{C}(\texttt{x})$$

for the two $\mathcal{SDL}$ sentence types,

$$\overline{N} = \lambda \texttt{x}.N(\texttt{x})$$

for all primitive concept names N,

$$\overline{\texttt{any}} = \lambda \texttt{x}.\texttt{true}$$

for the predefined concept `any`, and

$$\overline{(\texttt{and}\, C_1, \ldots, C_n)} = \lambda \texttt{x}.\overline{C_1}(\texttt{x}) \wedge \ldots \wedge \overline{C_n}(\texttt{x})$$

for a primitive term.

The translation of image terms is a bit more complicated, because it involves, for a term (`image` A R),[4] the application of the functions in R to the symbolic image A. The following is a constructive definition of the structured concept $\overline{[A, R]}$:

1. Expand the symbolic image A into canonical form. This produces a set of n components $A' = \{c_1, \ldots, c_n\}$.

2. Create a set $\overline{v}^n$ of n variables. Create an arbitrary bijection $v : A' \rightarrow \overline{v}^n$. The function v fixes the components of A' to unique variables.

3. Let the expression E be `true`. Iterate: for each component $c \in A'$, let E be $E \wedge \overline{\texttt{first}(c)}(v(c))$. Note the recursive invocation of the translation procedure to the components of A'.

[4]Henceforth, abbreviated $[A, R]$.

4. Consider each relation identifier $r \in R$. If r is a k-ary relation, then for all k-tuples $c_1, \ldots, c_k$ of components in A', conjoin to E either the literal

$$r(v(c_1), \ldots, v(c_k))$$

if

$$r(c_1, \ldots, c_k) = \texttt{true}$$

or the literal

$$\texttt{not}\ r(v(c_1), \ldots, v(c_k))$$

if

$$r(c_1, \ldots, c_k) = \texttt{false}.$$

Note that we have employed a strongly direct semantics for image terms in Step 4, since negated literals are computed from the symbolic image.

The desired structured concept description is then

$$\lambda \mathtt{x}.(\exists^* \overline{v}^n)\ \Pi(\mathtt{x}, \overline{v}^n) \wedge E.$$

The translation of an $\mathcal{SDL}$ terminology $\mathcal{T}$ is a set of sentences in first order logic:

$$\overline{\mathcal{T}} = \bigcup_{t \in \mathcal{T}} \overline{t}.$$

This completes the presentation of a denotational semantics for $\mathcal{SDL}$.

Operational semantics

Table 3.3 presents the simple operational semantics for a description logic. Concept names are analogous to *variables* in standard programming languages, and their associations need to be maintained in a symbol table (Tennent 1991). A symbol table (*State* in Table 3.3), is a function from identifiers to concept terms.

An important issue in description logic processing, in particular for knowledge base revision and updating, is the semantics given to concept names. A *lazy* semantics ("reference by name" in Table 3.3) simply places identifiers in the symbol table and expands names occurring in definitions when needed. Revisions to a name in a definition will affect the semantics of the associated identifier. An *eager* semantics ("reference by meaning" in Table 3.3) recursively expands out all names occurring in a definition at the time it is placed into the symbol table, using the `exp` function and the current state. The semantics of an

$State = Identifier \rightarrow ConceptTerm$
$\{\!|\cdot|\!\} : Sentence \times State \rightarrow State$

Sequencing:
$\{\!|s_1 ; s_2|\!\}_s = \{\!|s_2|\!\}_{\{\!|s_1|\!\}_s}$

Reference by name (lazy):
$\{\!|\texttt{defconcept}\, N\, C|\!\}_s = (s \mid N \mapsto C)$
$\{\!|\texttt{defprimconcept}\, N\, C|\!\}_s = (s \mid N \mapsto (\texttt{and}\; C\; N'))$

Reference by meaning (eager):
$\{\!|\texttt{defconcept}\, N\, C|\!\}_s = (s \mid N \mapsto \texttt{exp}_s(C))$
$\{\!|\texttt{defprimconcept}\, N\, C|\!\}_s = (s \mid N \mapsto (\texttt{and}\; \texttt{exp}_s(C)\; N'))$

Table 3.3: Styles of operational semantics for $\mathcal{SDL}$

identifier is fixed at definition time and can only be modified by adding or removing parts of the defining concept term. The current implementation of $\mathcal{SDL}$ uses an eager semantics. *Terminological cycles* (Nebel 1991) — situations where an identifier is reencountered during the expansion of its defining term — are prohibited.

The operational rules for the `defprimconcept` sentence type needs some special discussion. It can be shown that a primitive concept name can only subsume another concept whose expansion also contains that name (Nebel 1990). Thus for every primitive concept name N we introduce an identifier N', called a *primitive component* (Nebel 1990).[5] When primitive terms are expanded, they are expanded down to the primitive component level.[6] The concept term `any` is removed from the expansion of a primitive term, since it is redundant. Primitive components cannot appear in concept declarations. Nebel (1990) shows that an expanded term containing primitive components is extensionally equivalent to the original term, and therefore that expansion preserves the denotation of the original term. I shall assume this result below.

An example

As an illustration of the syntax and semantics of $\mathcal{SDL}$, refer to the simple blocks world program in Figure 3.1. There are three primitive shapes: `odd`, `circle`, and `square`. The coordinate set is the set of integer pairs $\{0, 1\} \times \{0, 1\}$. The bottom part of the figure shows a blocks world stack `i1` represented as an $\mathcal{SDL}$ image term (also see Figure 3.2).

The first statement defines a primitive concept `object` as a specialization of the predefined

[5] Not to be confused with the components of symbolic images.

[6] Throughout this thesis, for convenience, the `and` constructor will be dropped from expanded primitive terms, so that they can be represented as *sets* of primitive components.

```
defprimconcept object any;
defprimconcept odd object;
defprimconcept circle object;
defprimconcept square object;
defconcept i1 (image (
                   (square (1 0))
                   (odd    (0 0))
                   (circle (1 1)))
                (left on));
```

Figure 3.1: A small $\mathcal{SDL}$ terminology from a simple blocks world

concept `any`. The denotational semantics of this statement is

$$\forall \mathtt{x}\, \mathtt{object}(\mathtt{x}) \Rightarrow \mathtt{true}$$

Operationally, the effect to the term table is to add an entry for the expanded definition of `object`, i.e.,

$$\mathtt{object} \mapsto \{\mathtt{object}'\}$$

The next declaration defines a specialization `odd` of the concept `object`. Its denotation is

$$\forall \mathtt{x}\, \mathtt{odd}(\mathtt{x}) \Rightarrow \mathtt{object}(\mathtt{x})$$

and the term table at `odd` is

$$\mathtt{odd} \mapsto \{\mathtt{odd}', \mathtt{object}'\}$$

The next two statements of Figure 3.1 are processed in a similar fashion.

The image term defines a small blocks world stack using three objects and two relations: `left` and `on`. These relations, defined as binary predicates at the implementation language level, are assumed to be equivalent to the following functions:

$$\mathtt{on} \stackrel{\mathrm{def}}{=} \lambda \mathtt{xy}.\mathtt{first}(\mathtt{second}(\mathtt{x})) = \mathtt{first}(\mathtt{second}(\mathtt{y})) + 1$$

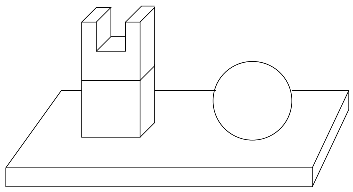
and

$$\mathtt{left} \stackrel{\mathrm{def}}{=} \lambda \mathtt{xy}.\mathtt{second}(\mathtt{second}(\mathtt{x})) < \mathtt{second}(\mathtt{second}(\mathtt{y}))$$

where `first` and `second` index into the concepts and coordinates of a image component. The denotational semantics of the last declaration is

$$\forall \mathtt{x}\, \mathtt{i1}(\mathtt{x}) \Leftrightarrow \overline{[A, R]}(\mathtt{x})$$

To compute $\overline{[A, R]}$ it is necessary to apply the four translation steps outlined earlier using the defined functions `on` and `left`. The result of the translation is illustrated in Figure 3.2.



```
λ x.∃* p1 p2 p3
    π(x,p1) ∧ π(x,p2) ∧ π(x,p3) ∧
    odd(p1) ∧ square(p2) ∧ circle(p3) ∧
    on(p1,p2) ∧
    not on(p1,p3) ∧
    not on(p2,p1) ∧
    not on(p2,p3) ∧
    not on(p3,p1) ∧
    not on(p3,p2) ∧
    not left(p1,p2) ∧
    left(p1,p3) ∧
    not left(p2,p1) ∧
    left(p2,p3) ∧
    not left(p3,p1) ∧
    not left(p3,p2)
    not left(p1,p1) ∧ not left(p2,p2) ∧ not left(p3,p3) ∧
    not on(p1,p1) ∧ not on(p2,p2) ∧ not on(p3,p3)
```

Figure 3.2: The logical equivalence semantics for the image term of `i1` (see Figure 3.1) which preserves the `left` and `on` relations. Left: a stylized depiction of the symbolic image. Right: the translation of the image term into a structured concept.

3.2.3 Validation of subsumption

Subsumption in a description logic is computed by validating the sequent

$$\mathcal{T} \vdash A \succeq B$$

for some terminology $\mathcal{T}$ and concept terms A and B. This computation is called *subsumption in a terminology* (Nebel 1990). In most knowledge representation systems (e.g., CLASSIC and KRIS), the terminology $\mathcal{T}$ is factored out of the sequent, and subsumption in a terminology is computed by a *term subsumption* procedure. Term subsumption is the validation of the sequent

$$\vdash \mathtt{exp}_{\{\!|\mathcal{T}|\!\}}(A) \succeq \mathtt{exp}_{\{\!|\mathcal{T}|\!\}}(B), \tag{3.3}$$

where, according to the sequencing rule and operational semantics of Table 3.3, $\{\!|\mathcal{T}|\!\}$ is the state resulting from interpreting the complete description logic program $\mathcal{T}$.[7] A term

[7] Henceforth, for readability, the curly brackets in $\{\!|\mathcal{T}|\!\}$ in Expression 3.3 will be dropped. Also, the turnstile $\vdash$ in (3.3) will be dropped when it is evident that term subsumption, and not subsumption in a terminology, is being discussed.

$$\frac{C \subseteq D}{C \succeq D}$$

$$\frac{R \subseteq R' \qquad \exists f : I \xrightarrow{R} J \quad \forall i \in I \quad \mathtt{first}(i) \succeq \mathtt{first}(f(i))}{[I, R] \succeq [J, R']}$$

Table 3.4: Structural subsumption rules for $\mathcal{SDL}$. C and D are expanded primitive terms. I and J are symbolic images, and R is a relation set.

subsumption algorithm operates on expanded terms and therefore need not subsequently refer to the underlying terminology $\mathcal{T}$.

There are various approaches to validating term subsumption; they can be roughly classified into *extensional* and *intensional* (or *structural*) approaches (Woods 1991). Extensional approaches attempt to validate the coherence of a concept using refutation proof procedures. Structural approaches inspect the internal syntactic structure of terms with identical constructors (Cohen et al. 1992), and are usually more specialized and efficient that their extensional counterparts. Two examples of extensional approaches are the one used by the $\mathcal{KRIS}$ description logic and the constraint satisfaction method reported by Schmidt-Schauß and Smolka (1991). Both the CLASSIC and the KL-ONE description logics use structural subsumption. The $\mathcal{SDL}$ description logic uses structural subsumption.

Table 3.4 gives the two basic structural subsumption rules for $\mathcal{SDL}$. The rules are understood to be bidirectional. Subsumption between expanded primitive terms is simply a subset relation (first rule). Subsumption between two canonical-form image terms is computed by finding a *relational monomorphism* (Haralick and Shapiro 1993) between them. Formally, let r be a k-ary relation and let I and J be two canonical-form symbolic images. An *r-monomorphism* is an injective function $f : I \to J$ that obeys the following property: for all k-tuples $c_1, \ldots, c_k$ of components in I,

$$r(c_1, \ldots, c_k) = r(f(c_1), \ldots, f(c_k)).$$

If R is a *set* of relations, and a function is a r-monomorphism for all $r \in R$, it is called an *R-monomorphism*. The set of all R-monomorphisms between two images I and J is denoted $I \xrightarrow{R} J$.[8]

By definition, one image term $[I, R]$ subsumes another image term $[J, R']$ if $R \subseteq R'$ and

[8] A *homomorphism* is a monomorphism which is not injective. An *isomorphism* is a bijective monomorphism (Vosselman 1992). The notion of R-monomorphisms and R-isomorphisms will appear again in Chapter 4. As mentioned in Chapter 2, most structured concept discovery systems do not use homomorphisms.

if there exists an R-monomorphism $f : I \xrightarrow{R} J$ which also preserves subsumption between corresponding components. This condition is given by the second rule in Table 3.4. Since the parts of an image can be another concept this inductive definition of subsumption extends immediately and elegantly to multilevel structured concepts.

As expected, there is a soundness and completeness result for $\mathcal{SDL}$:

Theorem 1 *Subsumption in $\mathcal{SDL}$, computed according to the structural subsumption rules of Table 3.4, is sound and complete, that is, $\mathcal{T} \vdash C \succeq D$ if and only if $\mathcal{T} \models C \succeq D$.*

Proof:

1. $\mathcal{T} \vdash C \succeq D$ if and only if $\mathtt{exp}_{\mathcal{T}}(C) \succeq \mathtt{exp}_{\mathcal{T}}(D)$ (Nebel's result),
2. if and only if $\overline{\mathtt{exp}_{\mathcal{T}}(C)}$ θ-subsumes $\overline{\mathtt{exp}_{\mathcal{T}}(D)}$ (a proof is given in Appendix A),
3. if and only if $\mathcal{T} \vdash \forall \mathtt{x}\ \overline{\mathtt{exp}_{\mathcal{T}}(D)}(\mathtt{x}) \Rightarrow \overline{\mathtt{exp}_{\mathcal{T}}(C)}(\mathtt{x})$ (Gottlob 1987, Theorem 6, is a statement and proof of a similar result),
4. if and only if $\mathcal{T} \models \forall \mathtt{x}\ \overline{\mathtt{exp}_{\mathcal{T}}(D)}(\mathtt{x}) \Rightarrow \overline{\mathtt{exp}_{\mathcal{T}}(C)}(\mathtt{x})$ (soundness and completeness of first order logic),
5. if and only if $\mathcal{T} \models C \succeq D$ (Nebel's result; definition of $\succeq$).

As required for soundness and completeness. ■

Since subsumption validation for general structured concepts is an NP-complete problem (see Chapter 2), it should not come as a surprise that this is also true of $\mathcal{SDL}$ concepts. The proof of this proposition is straightforward once it is shown that arbitrary undirected graphs can be expressed in $\mathcal{SDL}$ (see Appendix B).

There are a number of search techniques which can be used for finding monomorphisms, including tree searching, hyper-resolution techniques, and extensions of graph isomorphism techniques (Barrow et al. 1972). Computing monomorphisms in $\mathcal{SDL}$ may be slightly more expensive than for equivalent non-direct representations. This is because the possible extra time needed to compute a literal from an image must be taken into account. It must be stressed again that the main motivation of direct representations such as $\mathcal{SDL}$ is not computational efficiency, but rather comprehensibility and conciseness.[9]

[9]It is possible, however, that the spatial structure of the direct representation could be exploited during monomorphism searching. Given a partial monomorphism, and a component, we could compute in which regions of space to search for corresponding components in the other image. This may reduce the number of candidate partial monomorphisms considered during a search.

An example

To illustrate the computation of subsumption in $\mathcal{SDL}$, refer again to the terminology $\mathcal{T}$ of Figure 3.1 which defines the concept `i1` (also see Figure 3.2):

```
defconcept i1 (image (
                   (square (1 0))
                   (odd    (0 0))
                   (circle (1 1)))
                (left on))
```

Furthermore, consider the concept `uniq-2` given by the definition:

```
defconcept uniq-2 (image (
                   (object (1 0))
                   (object (1 1)))
                (left on))
```

that is, two objects, side by side. Suppose we wish to validate the sequent

$$\mathcal{T} \vdash \texttt{uniq-2} \succeq \texttt{i1}.$$

Here is a trace through the subsumption rules of Table 3.4. First we note that the relation set of `uniq-2` is a subset of the relation set of `i1`:

$$\{\texttt{left}, \texttt{on}\} \subseteq \{\texttt{left}, \texttt{on}\}.$$

Second, a relation-preserving monomorphism between their symbolic images must be found. Consider the monomorphism

$$\texttt{(object (1 0))} \overset{f}{\mapsto} \texttt{(circle (1 0))}$$
$$\texttt{(object (1 1))} \overset{f}{\mapsto} \texttt{(odd (1 1))}$$

That f is a $\{\texttt{left}, \texttt{on}\}$-monomorphism is easily verified, first for the `left` relation:

$$\begin{aligned}\texttt{left}(\ \texttt{(object (1 0))}\ ,\ \texttt{(object (1 1))}\ ) &=\\ (0 < 1) &=\\ \texttt{true}\\ \texttt{left}(\ f(\texttt{(object (1 0))})\ ,\ f(\texttt{(object (1 1))})\ ) &=\\ \texttt{left}(\ \texttt{(circle (1 0))}\ ,\ \texttt{(odd (1 1))}\ ) &=\\ (0 < 1) &=\\ \texttt{true}\end{aligned}$$

then for the `on` relation:

$$\begin{aligned}\texttt{on}(\ \texttt{(object (1 0))}\ ,\ \texttt{(object (1 1))}\ ) &=\\ (1{=}1+1) &=\\ \texttt{false}\\ \texttt{on}(\ f(\texttt{(object (1 0))})\ ,\ f(\texttt{(object (1 1))})\ ) &=\\ \texttt{on}(\ \texttt{(circle (1 0))}\ ,\ \texttt{(odd (1 1))}\ ) &=\\ (1{=}1+1) &=\\ \texttt{false}\end{aligned}$$

Finally, it must be verified that f preserves subsumption between corresponding components. This is done for the two components in the domain of f:

$$\begin{aligned}\mathtt{exp}_{\mathcal{T}}(\mathtt{first}((\mathtt{object}\ \ (1\ \ 0)))) \succeq \mathtt{exp}_{\mathcal{T}}(\mathtt{first}((\mathtt{circle}\ \ (1\ \ 0)))) &= \\ \mathtt{exp}_{\mathcal{T}}(\mathtt{object}) \succeq \mathtt{exp}_{\mathcal{T}}(\mathtt{circle}) &= \\ \{\mathtt{object}'\} \subseteq \{\mathtt{circle}', \mathtt{object}'\} &= \\ &\mathtt{true}\end{aligned}$$

$$\begin{aligned}\mathtt{exp}_{\mathcal{T}}(\mathtt{first}((\mathtt{object}\ \ (1\ \ 1)))) \succeq \mathtt{exp}_{\mathcal{T}}(\mathtt{first}((\mathtt{odd}\ \ (1\ \ 1)))) &= \\ \mathtt{exp}_{\mathcal{T}}(\mathtt{object}) \succeq \mathtt{exp}_{\mathcal{T}}(\mathtt{odd}) &= \\ \{\mathtt{object}'\} \subseteq \{\mathtt{odd}', \mathtt{object}'\} &= \\ &\mathtt{true}\end{aligned}$$

Hence we conclude that $\mathcal{T} \vdash \mathtt{uniq{-}2} \succeq \mathtt{i1}$. This was an exceedingly simple example of subsumption validation. One can imagine scenarios where both concepts contain several parts, relations are not only binary, and so on. Such scenarios will be encountered throughout Part II of this thesis. Also, this example only considered *point* data structures, where parts are placed at single representative points in an address space. It is easy to imagine situations where the address space should allow for the expression of regions (in 2D space) or volumes (in 3D space). This could be done in $\mathcal{SDL}$ by expressing coordinates as implicit equations for regions or volumes. Functions could then manipulate these equations to detect relationships.

3.2.4 Classification

A *classifier* is the most important facility of a description logic implementation. The idea behind classification is to view the knowledge base as a taxonomy of defined concept names. This is a lattice denoting the partial order on concept names induced by the subsumption relationship.

Classification can dramatically reduce the complexity of subsumption–based inferences in a description logic. For example, with appropriate implementation techniques (e.g., Ait-Kaci et al. 1989; Agrawal et al. 1989) a classifier yields a constant time algorithm for concept name subsumption. Classification is also used in the *realization* process (Nebel 1990): finding the set of most specific concept names that an individual instantiates. A classifier is also a crucial component of classification–based information retrieval models (Beck et al. 1989; Patel-Schneider et al. 1984). See (Doyle and Patil 1991) for a more detailed discussion of the role of classification in various reasoning processes.

Classification has been given an elegant formulation by Woods (1991). Imagine a *virtual taxonomy* of concepts, in which all expressible concept terms exist as nodes in a subsumption lattice. There will be an unbounded number of such terms. An *explicit* taxonomy gives names to some of these nodes, and brings them into implementation structures via a symbol table. Classifying a terminology amounts to the construction of an explicit taxonomy. Note that the explicit taxonomy need not only have user-defined names; some nodes can be *anonymous* concepts (Nebel 1990; Woods 1991).

Many different classification algorithms have been proposed (Ellis 1991; Levinson 1992; Nebel 1990; Schmolze and Lipkis 1983): all are based on a common principle (Woods 1991). Given a concept taxonomy, and a defined concept name N to insert into the taxonomy, two sets of concepts must be identified. First the MSS or *most specific subsumers* of N must be found. Once this is done, the MGS or *most general subsumees* of N is found. The new concept is inserted as a node below all MSS and above all MGS in the taxonomy. Care must be taken to manage cycles in the taxonomy which result from the classification of extensionally equivalent concepts. The classification algorithm used in $\mathcal{SDL}$ is a topological breadth-first search method (Ellis 1991), which has been found empirically to be generally more efficient than other algorithms (Baader et al. 1992).

Existing classifications algorithms are efficient in that they reduce the number of subsumption tests required, but where structured concepts are concerned, each individual subsumption test can be very costly. Fortunately, it can be shown that *cached monomorphisms* between terms in the taxonomy can be used to expedite subsumption computations during classification. Briefly, suppose that one has cached the R-monomorphisms between two stored concepts C and D, where $C \succeq_{\mathcal{T}} D$, and suppose that one also knows the R-monomorphisms between C and a query concept Q. Then the number and form of the possible R-monomorphisms between D and Q are constrained by this knowledge. It can be shown that any R-monomorphism between D and Q must be an elaboration of the inverse of some fixed $h : C \overset{R}{\rightarrow} D$ composed with some $f : C \overset{R}{\rightarrow} Q$. This technique was implemented in an early version of the IMEM structured concept discovery system (Conklin and Glasgow 1992). A similar technique has been reported by Barrow and Burstall (1976) and Levinson (1994), and hinted at by Haussler (1989), Lebowitz (1986a), and Wogulis and Langley (1989). The technique applies to any taxonomic logic where subsumption can be expressed as an injective function over expanded descriptions, and Conklin and Jenkins (1993) have proved the general theorem and applied it to a simple description logic without image terms.

3.3 Discussion

This chapter has presented $\mathcal{SDL}$, a Spatial Description Logic for representing structured concepts in a description logic framework. There were two research goals for the development of $\mathcal{SDL}$. One was to explore in an abstract sense the idea of direct knowledge representations, and their use as a representation for structured concepts. The other goal — somewhat more pragmatic — was to formulate a hypothesis space for a machine discovery system.

The $\mathcal{SDL}$ language is specifically tailored to representing structured concepts and lacks some of the more specific facilities of other description logics. For instance, $\mathcal{SDL}$ has no `all` constructor. It does, however, have the ability to express structured concepts in a very concise manner. This is in contrast to other description logics supporting non-direct types of structural descriptions (e.g., Brachman and Schmolze 1985; Napoli 1992; Padgham and Lambrix 1994; Weida and Litman 1992). Also, $\mathcal{SDL}$ has a couple of other nice properties for structured concept representation. First, it is possible to visually depict an $\mathcal{SDL}$ concept on a graphics terminal, since $\mathcal{SDL}$ concepts have a symbolic image component which is spatial in nature. Second, $\mathcal{SDL}$ guides and perhaps simplifies the selection of an ontology, in that the expressible relations are just the ones that can be inferred directly from images.

The machine learning hypothesis space defined by $\mathcal{SDL}$ is less powerful than a general structured concept representation (see Chapter 2), because a translated image term has all k-tuples of variables for a k-ary relation r. That is, for a particular tuple of components $c_1, \ldots, c_k$, either $r(v(c_1), \ldots, v(c_k))$ or its negated form is present in the translated description. It is not possible to leave particular literals out of the description. There are, however, ways around the problem. One is a hybrid scheme, which would associate with an image term a list of tuples for which relations cannot be inferred. Another idea, which retains the "directness" of the scheme, is to use a disjunction of image terms: one disjunct where the tuple is true and another where it is false. An object would satisfy such a concept if it were satisfied by either disjunct. A similar proposal is made by Glasgow (1994) in the context of direct representations of ground atomic facts. A potential problem with this approach is the number of disjunctions that may need to be maintained.

There are many other enhancements that could be made to the expressiveness of $\mathcal{SDL}$. For instance, the components of images could be *negations* or *disjunctions* of primitive concept names. While enhancing expressiveness, these additions would vastly expand the size of the hypothesis space for machine discovery. Because there would be many more possible generalizations of a set of examples, there would be more available nondeterminism and search involved for a discovery system. Another extension to $\mathcal{SDL}$ could be the addition of a "type" primitive component to every image term. This could represent image shape or predetermined classification, and would allow for supervised learning. Rather than develop a fully expressive language, the view taken here is to craft a small yet adequately expressive description logic. Part II of this thesis will show that $\mathcal{SDL}$ concepts, as presented here, are quite effective in expressing known and discovered classes of molecular data.

An issue which has not been addressed here is that of an "assertional reasoning" capability. Some description logics divide their knowledge base into two components; a terminology (or "TBox") representing concept definitions, and a world description (or "ABox") containing ground facts and rules. Instantiation of a concept by an object can be tested by first *individuating* the object — computing a sufficiently specific concept which represents that object — and testing for structural subsumption. KL-ONE, on the other hand, does not have an assertional component and marks all concepts as either "individual" or "generic". Systems that store individual concepts rather than assertions must enforce that an individual cannot subsume any non-individual concept. The K-REP system (Mays et al. 1991) computes individual concepts on the fly from an associated relational database. The idea of individual concepts has been implemented in $\mathcal{SDL}$, which has a mechanism for specifically marking concepts identified as individuals, and has virtual links to direct molecular databases.

Finally, at the 1993 AAAI Symposium on Machine Learning in Computer Vision, I presented the relationship of $\mathcal{SDL}$ to various approaches to object recognition in computer vision (Conklin 1993). I concluded that the success of the $\mathcal{SDL}$ approach to knowledge representation for object recognition depends intimately on the relations used to describe images. For most modelling tasks it is highly desirable that these be invariant under rigid Euclidean transformations (translation, rotation). For example, the relation "nearer" between three parts (van Benthem 1991, Appendix A: "On Space") is invariant under the Similarity group of transformations (translation, rotation, scaling). Invariant relations have the same value regardless of the image coordinate frame. However, merely stating that relations must be transformation-invariant is not enough — the "universal" relation and many others are transformation-invariant, but clearly differ in utility for discrimination during concept formation or for filtering out invalid object models. On the other hand, if very specific relations are used then little similarity between objects can be detected. Overly general and overly specific relations will therefore violate, respectively, the *separability* and *stability* requirements (Wolfson and Lamdan 1992) for object models. The transformation-invariant relations I have used to describe molecular structures will be presented in Part II of this thesis.

Chapter 4

Discovery in the Spatial Description Logic

The preceding chapter outlined the syntax and semantics of $\mathcal{SDL}$, a Spatial Description Logic. In this logic concise and comprehensible descriptions of structured concept — configurations of parts and their interrelationships — can be expressed. In addition, descriptions of complex concepts can be simplified by aggregating their parts into multiple levels, while retaining a well–defined semantics. The logic is capable of the reasoning tasks of classification, subsumption, and instantiation.

The construction of an $\mathcal{SDL}$ knowledge base by hand, however, will likely be tedious for most applications. To exploit the taxonomic reasoning abilities of $\mathcal{SDL}$, a user should express both structured objects and various subsuming concepts which indicate how the structured objects are related. This task requires the recognition of recurrent structural patterns in data and, due to its combinatorial and complex nature, can best be handled by an automated machine discovery procedure.

This chapter describes a method for structured concept discovery called IMEM (Image MEMory) which dynamically forms new concepts, classifies them into an evolving knowledge base, and uses these to guide further concept formation. This chapter uses the underlying $\mathcal{SDL}$ logic to describe the behaviour of the discovery method in largely denotational, as opposed to operational, terms. This means that this chapter describes a *specification* for IMEM rather than a particular implementation.

The IMEM method creates new concepts by iteratively generalizing pairs of concepts deemed to be *similar*. Thus as a prerequisite to presenting IMEM, it is necessary to fully explain the way similarity is measured, and to provide a rationale for the method. This is the role of Section 4.1 of this chapter. A specification of IMEM, and a simple example of structured concept discovery, is given in Section 4.3.

In addition to enhancing the logic $\mathcal{SDL}$ by supporting the inference of generalization, IMEM also supports the important task of efficient similarity–based retrieval of structured objects. Section 4.2 of this chapter presents a theorem which, loosely stated, guarantees that IMEM will retrieve similar concepts in response to a query. Thus IMEM can support *analogical reasoning*: predicting further similarities between two objects on the basis of partial observed similarity. The topic of analogical reasoning is the subject of Section 4.4 of this chapter. Part II of this thesis will demonstrate the application of IMEM to knowledge discovery in molecular structure databases.

4.1 An axiomatic theory of similarity

In this section I will present an axiomatic theory of similarity for $\mathcal{SDL}$. An axiomatic theory of similarity comprises axioms which must be obeyed by any function used for measuring similarity (Tversky 1977). There has been a tendency in concept discovery research of using similarity functions that may appear to be arbitrarily chosen (e.g., MERGE and KBG; see Chapter 2). I feel that it is important to establish, prior to upholding a particular similarity function, the intuitive desired properties expected of a similarity function. These are given by natural deduction rules.

In numerical clustering, objects are represented by quantitative feature vectors: points in multidimensional space.[1] A distance metric on points is defined, and clusters with low overall intra-cluster and high inter-cluster distances are sought after. For conceptual clustering, objects are usually represented by a set of qualitative features and relations, and it is more appropriate to measure similarity by performing qualitative feature matching. In his seminal work on feature similarity, Tversky (1977) claims that a similarity coefficient S between feature sets A and B should be a function F of three different sets:

$$S(A,B) \stackrel{\text{def}}{=} F(A \cap B, A-B, B-A) \tag{4.1}$$

that is, a function of the features common to A and B, the features in A but not in B, and the features in B but not in A. The features sets A and B need not have the same cardinality. Tversky outlines an axiomatic theory of similarity, and claims that the axioms

[1] Recall that a *feature* is an attribute-value pair. I make the simple distinction that a quantitative feature has an infinite domain, while a qualitative feature has a finite domain. Similarly, Smith and Medin (1981) distinguish qualitative *features* from quantitative *dimensions*.

are satisfied by the *ratio model* [2]

$$S(A,B) \stackrel{\text{def}}{=} \frac{|A \cap B|}{|A \cap B| + \alpha|A-B| + \beta|B-A|} \tag{4.2}$$

for non-negative constants α and β. This function is standardized to lie in the real interval $[0,1]$. Standardization is essential when assessing the relative similarity of objects with varying numbers of features. The Dice and Tanimoto similarity coefficients for feature set similarity, popular in textual and chemical information retrieval systems (Salton and McGill 1983; Willett 1990), arise by choosing $\alpha = \beta = 0.5$ (Dice) or $\alpha = \beta = 1$ (Tanimoto).

A similarity coefficient in $\mathcal{SDL}$ is a function between two concept terms:

$$S_{\mathcal{T}} : ConceptTerm \times ConceptTerm \rightarrow [0,1].$$

Note that the function S is subscripted by a terminology $\mathcal{T}$: the similarity of two concepts can and should vary between different terminologies. This is achieved by expanding primitive terms during the assessment of similarity. There are two types of concept terms in $\mathcal{SDL}$: image terms and primitive concept terms. Since expanded primitive concept terms are just feature sets, Expression 4.2 can be applied to them directly. A technique for measuring the similarity of image terms, on the other hand, requires a different approach.

Like primitive concept terms, the similarity of two images can be measured in terms of the number of syntactic transformations needed to bring them into extensional equivalence (Eshera and Fu 1984). Many types of image transformations are possible, such as replacing, deleting, or moving a part, or moving all parts simultaneously (e.g., rotation). Here, as in (Shapiro 1980), we only consider part deletion. It turns out that many other complex transformations are implicit, by requiring that appropriate invariant relations are preserved.

Two symbolic images A and B are considered to be *structurally equivalent*, denoted $A \cong_R B$, if there exists an R-isomorphism between them. Note that the extensional equivalence $A \equiv_{\mathcal{T}} B$ implies the structural equivalence $A \cong_R B$ but not vice versa, because the latter is based only on image structure, and not on part identity. Let $A \stackrel{R}{\rightharpoonup} B$ denote the set of all *partial* relational monomorphisms between the components of images A and B. A particular partial function in this set reflects the number of parts deleted from each image, since the partial function need not involve all parts of the images. Functions which involve more parts are preferable since they indicate fewer transformations, and similarity should be

[2]The ratio model is more general than this: here I use the set cardinality function $|\cdot|$ as an instance of Tversky's interval scale.

dependent on the chosen function. Let $f : A \xrightarrow{R} B$ be a particular partial R-monomorphism between two images. The similarity between two image terms $[A, R]$ and $[B, R]$ with respect to the monomorphism f should be, analogous to (4.1), a function F of three different sets:

$$S([A, R], [B, R], f) \stackrel{\text{def}}{=} F(\text{dom}(f), A - \text{dom}(f), B - \text{ran}(f)) \tag{4.3}$$

where $\text{dom}(f)$ are the parts participating in a structurally equivalent portion of the two images, $A - \text{dom}(f)$ are the parts deleted from A, and $B - \text{ran}(f)$ are the parts deleted from B. Note that these sets are analogous to $A \cap B$, $A - B$, and $B - A$ in (4.1). Just as the feature sets in (4.1) can be of different sizes, the image terms in (4.3) can contain different numbers of components.

The value of (4.3) varies with the choice of a particular monomorphism. Often we are interested in the *maximum* value of (4.3) over all partial R-monomorphisms:

$$S_{\mathcal{T}}([A, R], [B, R]) \stackrel{\text{def}}{=} \max_{f \in A \xrightarrow{R} B} S_{\mathcal{T}}([A, R], [B, R], f) \tag{4.4}$$

Henceforth, when we speak of *the* similarity between two image terms, the maximum value in (4.4) is assumed.

Table 4.1 presents the axioms of a similarity theory for $\mathcal{SDL}$. All of the rules, despite the formal presentation, are intuitive and can be explained in words. The first rule states that similarity is symmetric. The second states that equivalent concepts can be freely substituted with no change in similarity valuation. The third rule applies to image terms and states that similarity is monotonic: by overlaying an image (I) with a nonempty subimage of another containing it, a more similar image term is produced. The fourth rule is an analogous monotonicity rule for primitive concept terms.[3] The last rule states that replacing a part of an image with a more similar part yields a more similar image.

The axiomatic approach to similarity is interesting as it relates *pairs* of similarity quantities in the consequents of axioms. This is true to the philosophy of Goodman (1951), who states that "normally one asks not whether two qualities are similar but rather whether they are more or less similar than some other pair". There are a few noteworthy points about the particular axioms presented here. First, the similarity of an image term and a primitive term is undefined. Second, the similarity of two image terms is undefined if the two respective relation sets are not equivalent. This is a somewhat arbitrary restriction, and it would be a simple matter to include a monotonicity rule outlining the effects of

[3]For simplicity, here I have presented a slight specialization of Tversky's monotonicity axiom.

1. Symmetry

$$\frac{}{S_{\mathcal{T}}(A,B) = S_{\mathcal{T}}(B,A)}$$

2. Substitution

$$\frac{B \equiv_{\mathcal{T}} C}{S_{\mathcal{T}}(A,B) = S_{\mathcal{T}}(A,C)}$$

3. Image term monotonicity

$$\frac{I \subset I' \subseteq I''}{S_{\mathcal{T}}([I,R],[I'',R]) < S_{\mathcal{T}}([I',R],[I'',R])}$$

4. Primitive term monotonicity

$$\frac{P \subset P' \subseteq P''}{S_{\mathcal{T}}(P,P'') < S_{\mathcal{T}}(P',P'')}$$

5. Part replacement

$$\frac{S_{\mathcal{T}}(A,B) \geq S_{\mathcal{T}}(A,C)}{S_{\mathcal{T}}([\mathtt{put}(A,c,I),R],[\mathtt{put}(B,c,I),R]) \geq S_{\mathcal{T}}([\mathtt{put}(A,c,I),R],[\mathtt{put}(C,c,I),R])}$$

Table 4.1: Axioms for an $\mathcal{SDL}$ similarity coefficient $S_{\mathcal{T}}$. A, B, C are concept terms; I, I', I'' are symbolic images; P, P', P'' are primitive terms; R is a relation set; c is a coordinate.

dropping a relation identifier from a relation set. However, as Part II of this dissertation will demonstrate, the restriction to relation set equivalence has proved to be sufficient in practice. A third point has to do with the presence of the symmetry axiom. Tversky (1977) argues that humans sometimes display asymmetric similarity judgments, and his theory can accommodate asymmetric similarity coefficients. However, in real applications it is not clear when or why two image terms should be asymmetric in their similarity.

It is now possible to present the Dice similarity coefficient for $\mathcal{SDL}$. The similarity of two primitive terms C and D in $\mathcal{SDL}$ is defined to be

$$S_{\mathcal{T}}(C,D) \stackrel{\text{def}}{=} \frac{2 \times |\mathtt{exp}_{\mathcal{T}}(C) \cap \mathtt{exp}_{\mathcal{T}}(D)|}{|\mathtt{exp}_{\mathcal{T}}(C)| + |\mathtt{exp}_{\mathcal{T}}(D)|} \tag{4.5}$$

The similarity of two image terms $[A,R]$ and $[B,R]$ under a particular R-monomorphism f is defined to be

$$S_{\mathcal{T}}([A,R],[B,R],f) \stackrel{\text{def}}{=} \frac{2 \times \displaystyle\sum_{C \in \mathrm{dom}(f)} S_{\mathcal{T}}(\mathtt{first}(C), \mathtt{first}(f(C)))}{|A| + |B|} \tag{4.6}$$

Note that primitive concept terms are expanded during the assessment of similarity in (4.5).

This is how the background terminology $\mathcal{T}$ is taken into account.[4] Note also that similarity is recursively invoked for the components of image monomorphisms in (4.6). It can be shown, by structural induction, that (4.5) and (4.6) together preserve all of the axioms of Table 4.1. There are countless variations on the admissible similarity coefficients; as in Tversky's framework it is possible to outline general conditions for the satisfaction of the axioms of Table 4.1. Such a rigorous treatment of similarity, although possible, is excessive for this thesis and the coefficient of (4.5) and (4.6) will be used henceforth.

4.2 Similarity and subsumption

The previous section described, in some detail, how similarity of concept terms in $\mathcal{SDL}$ is measured. It remains to be shown how similar images are retrieved from a concept taxonomy. This is the role of the following theorem, which establishes the close relationship between similarity and subsumption.

Theorem 2 (Indexing) *Let $\mathcal{T}$ be an $\mathcal{SDL}$ terminology, C and D be any two concept terms such that $C \succeq_{\mathcal{T}} D$, and $S_{\mathcal{T}}$ be a similarity coefficient obeying the axioms of Table 4.1. For any concept term Q, if $D \succeq_{\mathcal{T}} Q$, then $S_{\mathcal{T}}(D,Q) \geq S_{\mathcal{T}}(C,Q)$. Furthermore, the inequality is strict whenever the relation $C \succeq_{\mathcal{T}} D$ is proper (i.e., $C \not\equiv_{\mathcal{T}} D$) (see Figure 4.1).*

The proof of the indexing theorem, by structural induction, is straightforward but lengthy and is deferred to Appendix C. Its impact is that the concept taxonomy can be used to index images for close-match retrieval. The theorem guarantees that as a query concept descends a concept taxonomy, fewer and more closely-matching source concepts will be retrieved. For a given query concept, all other concepts in the taxonomy above its most specific subsumers are less similar. Thus retrieval can be directed by the concept taxonomy, and need not involve enumeration of concepts. It must be emphasized that the theorem applies to "local" paths through a taxonomy, and is not "global" in that the most similar images in the whole knowledge base may not be retrieved.

Research in information retrieval (Salton and Wong 1978) has also led to a hierarchical cluster model for efficient retrieval of data objects. A hierarchy of clusters, each described by their centroid in multidimensional space, is used to restrict the number of similarity comparisons that must be made for a given query. Cluster centroids act as "locks," which can only be opened if a query concept is similar. The query then descends, recursively, to the

[4] Winston (1980) calls this process "classification–exploiting hypothesizing".

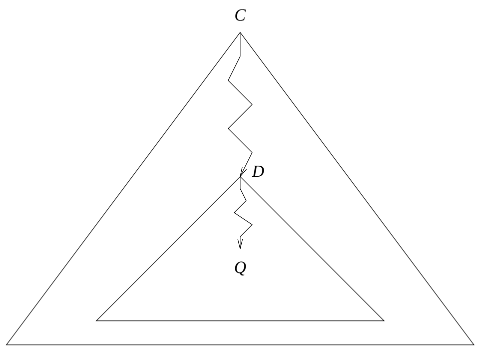


Figure 4.1: The indexing theorem: the triangles enclose the fringe of concepts C, D. Arrows denote subsumption. The concept Q is subsumed by D, and hence it is more similar to D than it is to C.

lower levels of the hierarchy. A fundamental result from hierarchical information retrieval is that there is a tradeoff between retrieval recall and precision, based on the number and tightness of the clusters.

As we have seen in Chapter 2, UNIMEM and several other concept formation systems employ a hierarchy of concepts to facilitate retrieval. Although these approaches to hierarchical retrieval do not use subsumption (clustered file searches use a distance from a centroid, while UNIMEM uses a similarity threshold from a concept), the principles of hierarchical retrieval also apply directly when concepts have a logical semantics. Centroids are analogous to concepts, while the cluster hierarchy is analogous to a concept taxonomy.

4.3 Concept discovery

This section will present a method for structured concept discovery in $\mathcal{SDL}$. The method relies heavily on the theory of $\mathcal{SDL}$ concept similarity outlined above. A prototype system called IMEM (Image MEMory) clusters a stream of image terms by iteratively recalling similar images, and by performing a generalization and classification step. IMEM is similar to a centroid hierarchical agglomerative method (Aldenderfer and Blashfield 1984), except that it is incremental, and it performs conceptual rather than numerical clustering. IMEM aims to create concepts that are both specific and recurrent in observed data.

Table 4.2 provides a high-level description of the IMEM algorithm. Given a current concept taxonomy, conceptual clustering is performed by iteratively invoking the `incorporate` routine. The parameter *image* is assumed to be a name in the $\mathcal{SDL}$ term table. A generalization process is considered by the `merge` routine for every immediate subsumee of each most specific subsumer. For a given immediate subsumee and monomorphism, if the

```
incorporate(image)
    let M be the most specific subsumers of image
    for each concept C in M do
        (a) place a subsumption link between C and image
        (b) merge(image,C)

merge(image,C)
    for all immediate subsumees D of C do
        for all maximal f : image -R-> D where S_T(image,D,f) >= τ do
            (a) use f to build a new concept which is a common subsumer of
                image and D, and give it a unique name U
            (b) classify(U)
```

Table 4.2: The IMEM algorithm. Top–level call: `incorporate`(*image*).

similarity is above a researcher-specified threshold τ (where $\tau \in [0,1]$), concept formation takes place. This invokes the (a) computation and (b) classification of a common subsumer.

The IMEM learning strategy is similar to Levinson's (1985) method (which, in turn, was based on UNIMEM), with a couple of notable exceptions. First, IMEM uses a threshold τ to evaluate a proposed concept, whereas Levinson evaluates a "cost" heuristic function (see Chapter 2 for a more detailed discussion of Levinson's system). Both of these techniques aim to create a taxonomy that is fairly well-balanced. In IMEM this is done by varying the concept formation threshold τ: if it is close to 0, many general concepts will be created, whereas if it is close to 1, few concepts will be created and the taxonomy will be maximally broad. Secondly, Levinson inspects the complete fringe of each most specific subsumer for similarity; IMEM, like UNIMEM, inspects only the *immediate subsumees* of each most specific subsumer.

4.3.1 Least common subsumers

Step (a) in the `merge` routine of IMEM computes a common subsumer of two image terms. The *least* common subsumer of two expanded concept terms is defined inductively as follows. The least common subsumer of expanded primitive concept terms is simply their set intersection. Note that background knowledge is taken into account, since the terms are expanded prior to the common subsumer computation. This is analogous to the concept saturation process as carried out in, for example, the KBG concept discovery system. Cohen et al. (1992) give a similar definition of least common subsumers for conjunctive concepts in description logics.

The computation of a structural similarity valuation between two image terms C and D necessarily produces a set of partial relational monomorphisms between them. Each such function f will build a common subsumer which is structurally equivalent to a subimage of C and a subimage of D. The parts of the common subsumer are produced by computing a common subsumer of parts put into correspondence by f. The coordinates of the common subsumer are those of the components in $\mathrm{dom}(f)$. To create *least* common subsumers, a *maximal* partial monomorphism must be used:

Definition 2 *A partial R-monomorphism* $f : A \xrightarrow{R} B$ *is* maximal *if, for any other partial R-monomorphism* $f' : A \xrightarrow{R} B$ *where* $|\mathrm{dom}(f')| > |\mathrm{dom}(f)|$, *there is no R-monomorphism in* $\mathrm{dom}(f) \xrightarrow{R} \mathrm{dom}(f')$.

It can be shown that if a partial monomorphism f is maximal, then it will build a least common subsumer. It can also be shown that the largest of all maximal monomorphisms is also the one which maximizes (4.4). The proofs are straightforward and are omitted here.

A least common subsumer is a concept in the $\mathcal{SDL}$ language. As such, it is also a direct structured concept: having relations defined between all parts present in the image. IMEM computes relatively coarse common subsumers. It is not possible, for example, to drop a single literal from a concept (as can be done in inductive logic programming systems). Note, however, that there may be more than one relational monomorphism leading to a similarity greater than τ, and more than one common subsumer will then be built by IMEM in step (a) of the `merge` routine.

4.3.2 Incrementality, complexity, and ordering effects

The IMEM method is incremental, in the sense that it does not extensively reprocess all previously–seen examples when a new one is encountered. There has been much recent interest in incremental conceptual clustering systems, which have also been called "concept formation" systems (Gennari et al. 1989). Incremental algorithms typically process a set of examples more efficiently than their nonincremental counterparts. A nonincremental version of IMEM would look very much like the KBG algorithm (Bisson 1992a), and would require $O(n^2)$ similarity comparisons for a dataset of size n. By contrast, with an incremental algorithm, after processing n examples we subsequently expect to perform $O(\log n)$ subsumption comparisons to reach a most specific subsumer, hence $O(n \log n)$ subsumption comparisons to process a dataset of size n.

A drawback of incremental algorithms is, of course, their sensitivity to the order of data presentation. In IMEM, a reordering of the stream of examples presented to the `incorporate` routine may produce a different taxonomy.[5] It should be noted that obtaining an optimal clustering — one that maximizes intra–cluster similarity and inter–cluster dissimilarity — is an NP-complete problem (Garey and Johnson 1979, problem MS9). Given the massive sizes of biomolecular databases, considered in Part II of this dissertation, it seems apparent that examples must be processed in time sublinear in the current number of examples processed. Incremental algorithms such as IMEM exhibit this performance.

The above informal complexity argument only considered the number of similarity and subsumption comparisons made, and ignored the complexity of an individual comparison. For $\mathcal{SDL}$ structured concepts, as mentioned in Chapter 3, subsumption is isomorphic to a subgraph isomorphism problem, which is known to be NP-complete. The philosophy I have adopted with IMEM is to carefully study, for different problems, how maximal relational monomorphisms can be best computed. Part II of this dissertation will elaborate on this point.

4.3.3 An example

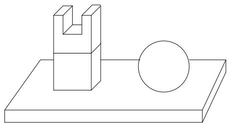
To illustrate the concepts of similarity and structured concept discovery, in this section I show how IMEM operates on the "blocks world" problem used by Thompson and Langley (1991) in their illustration of the LABYRINTH system. Although the problem is a simple one, the representation of spatial relations can be transferred almost directly to problems of great importance in chemistry such as conformational analysis, motifs in proteins, the topology of hydrogen bonding, and chemical connectivity. In this simple blocks world example, the spatial relations are with respect to a particular orientation, and rigid rotations to the images may shatter part relationships. In the molecular domain, it is crucial that spatial relations be invariant under 3D rotations and translations. The application of IMEM to large molecular datasets will be described in Part II of this dissertation.

Suppose that the learner has available the background terminology given in Figure 4.2. There are three primitive shapes: `odd`, `circle`, and `square`. The bottom part of the figure shows four blocks world stacks. Each of these images preserves the two relations `left` and `on` (see Section 3.2.2). Suppose that the images `i1` through `i4` are processed by IMEM in sequence, with a similarity threshold $\tau = 0.3$. The first image `i1` is added to the knowledge base. When `i2` is encountered, its most specific subsumers are found. In this case, the only most specific subsumer is the concept `any`. The current immediate subsumees of `anything` are identified (`object`, `i1`). The similarity of `i1` (an image term) and `object` (a primitive term) is undefined. Computing the similarity of `i1` and `i2` involves finding maximal relational monomorphisms between them. There are several maximal monomorphisms between `i1` and `i2`, each leading to a distinct similarity valuation. According to (4.6), their similarity $S_{\mathcal{T}}(\texttt{i1}, \texttt{i2}, f)$ with respect to the monomorphism

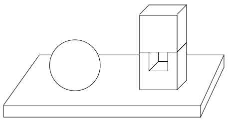
$$(\texttt{square}\ (1\ 0)) \overset{f}{\mapsto} (\texttt{circle}\ (1\ 0))$$
$$(\texttt{circle}\ (1\ 1)) \overset{f}{\mapsto} (\texttt{odd}\ (1\ 1))$$

[5]However, if $\tau = 1$, as in many of the experiments reported in Chapter 5, ordering effects are not present.

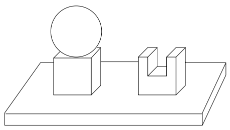
```
defprimconcept object any;
defprimconcept odd object;
defprimconcept circle object;
defprimconcept square object;

defconcept i1 (image (
                      (square (1 0))
                      (odd    (0 0))
                      (circle (1 1)))
                   (left on));
```



```
defconcept i2 (image (
                      (circle (1 0))
                      (odd    (1 1))
                      (square (0 1)))
                   (left on));
```



```
defconcept i3 (image (
                      (square (1 0))
                      (circle (0 0))
                      (odd    (1 1)))
                   (left on));
```



```
defconcept i4 (image (
                      (odd    (1 0))
                      (circle (1 1))
                      (square (0 1)))
                   (left on));
```

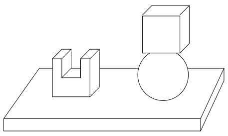


Figure 4.2: The background terminology and four structured objects (`i1`, `i2`, `i3`, `i4`) from a simple blocks world.

is

$$\frac{2 \times \sum_{C \in \mathrm{dom}(f)} S_{\mathcal{T}}(\texttt{first}(C), \texttt{first}(f(C)))}{6}$$

or

$$\frac{2 \times (S_{\mathcal{T}}(\texttt{square}, \texttt{circle}) + S_{\mathcal{T}}(\texttt{circle}, \texttt{odd}))}{6} \tag{4.7}$$

Evaluating this expression requires a recursive computation of the similarity of the parts put into correspondence by f. For example, according to (4.5),

$$\begin{aligned} S_{\mathcal{T}}(\texttt{square}, \texttt{circle}) &= \frac{2 \times |\texttt{exp}_{\mathcal{T}}(\texttt{square}) \cap \texttt{exp}_{\mathcal{T}}(\texttt{circle})|}{|\texttt{exp}_{\mathcal{T}}(\texttt{square})| + |\texttt{exp}_{\mathcal{T}}(\texttt{circle})|} \\ &= \frac{2 \times |\{\texttt{square}', \texttt{object}'\} \cap \{\texttt{circle}', \texttt{object}'\}|}{4} \\ &= 0.5 \end{aligned}$$

Substituting this and $S_{\mathcal{T}}(\texttt{circle}, \texttt{odd})$ (also 0.5) into (4.7) yields

$$S_{\mathcal{T}}(\texttt{i1}, \texttt{i2}, f) = \frac{2 \times (0.5 + 0.5)}{6}$$

or 0.33. Since this similarity is acceptably high (in this case, ≥ 0.3), a common subsumer `uniq-2` of the image terms `i1` and `i2` is formed and classified (see Section 3.2.3 for the $\mathcal{SDL}$ definition of `uniq-2`). The parts of `uniq-2` are the least common subsumers of parts put into correspondence by f. Finally, since f is maximal, `uniq-2` will be a least common subsumer of `i1` and `i2`.

There are four additional maximal monomorphisms between `i1` and `i2` that lead to a similarity above 0.3. The five monomorphisms are used to build the concepts `uniq-1` through `uniq-5`. The final concept taxonomy, after incorporation of all four examples, is displayed in Figure 4.3. There are several recurrent structured concepts that were discovered from the four examples. IMEM discovered that all examples have two objects, one left of the other (`uniq-2`); they all have a stack of two objects (`uniq-1`); `i1` and `i3` have a square supporting an object, with the square to the left of another object, and so on.

4.4 Analogical reasoning and knowledge discovery

The spatial description logic $\mathcal{SDL}$ is capable of the fundamental reasoning tasks of classification, instantiation, and subsumption. In this chapter, I have shown how a machine discovery procedure can be used to create new $\mathcal{SDL}$ concepts from examples. As a basis for this procedure, a theory of similarity and indexing in $\mathcal{SDL}$ was developed in Section 4.1. Using this similarity theory it is possible to discuss another reasoning method for $\mathcal{SDL}$: reasoning by analogy.

Analogical reasoning is based on the assumption that if two things share some properties, they might also share other properties (Winston 1980). An *analogical inference* is based on the following schema:

$$\frac{\mathcal{T} \vdash P(s) \wedge Q(s) \qquad \mathcal{T} \vdash P(t) \qquad \mathcal{T} \not\vdash \texttt{not}\ Q(t)}{\mathcal{T} \vdash Q(t)} \tag{4.8}$$

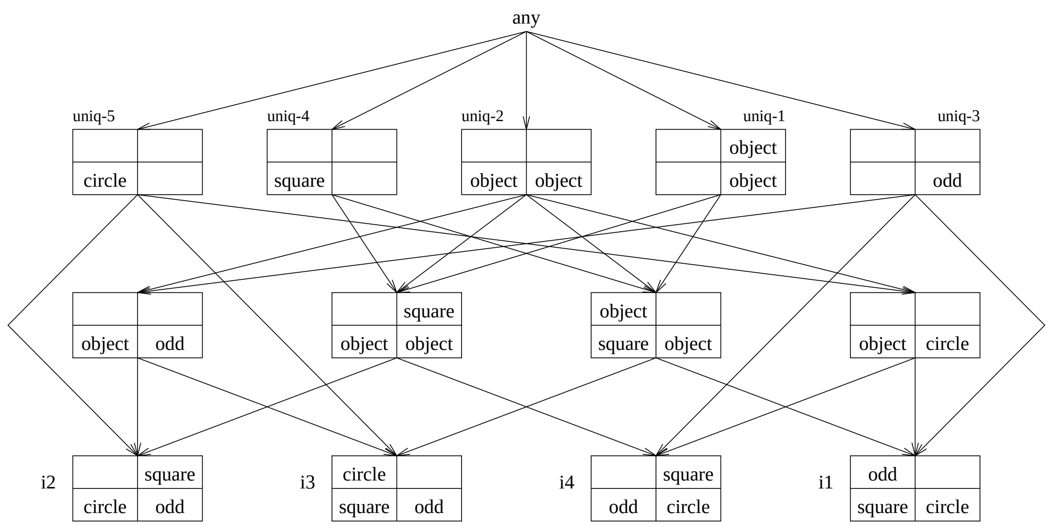


Figure 4.3: The concept taxonomy after IMEM processes the four structured objects of Figure 4.2. Primitive concepts have been omitted from the taxonomy.

where $\mathcal{T}$ is background knowledge, P and Q are concepts, and s and t are object constants (called the *source* and *target* analogues, respectively). The schema says that if s and t share some properties P, if s has some additional properties Q, and if $Q(t)$ is not inconsistent with what is already known, then infer $Q(t)$.

Actually, analogical reasoning can be thought of as the application of two separate steps, the first (marked with a $*$) nondeductive, and the second deductive:

$$\frac{\dfrac{\mathcal{T} \vdash P(s) \wedge Q(s) \qquad \mathcal{T} \not\vdash \exists x P(x) \wedge \texttt{not}\, Q(x)}{\mathcal{T} \vdash \forall x P(x) \Rightarrow Q(x)}\,(*) \qquad \mathcal{T} \vdash P(t)}{\mathcal{T} \vdash Q(t)} \tag{4.9}$$

Given a source analogue s, and the assumption that no object can be found which is an instance of P but not of Q, the first step $(*)$ hypothesizes a predictive rule, and the second step applies this rule. Note that the consequent of the chain of inferences in Expressions 4.8 and 4.9 does not follow from the assumptions under the rules of deductive inference. There has been much debate about when and whether analogical inferences can be considered plausible, and how best to recognize these plausible inferences. Indurkhya (1989) criticizes the so-called "predictive" analogy of (4.8) and (4.9), because it is logically unjustified. Although the consequent $Q(t)$ is plausible if its negation $\texttt{not}\, Q(t)$ is not provable, the rules of analogical inference can lead to many spurious conclusions. Any computational model of analogical reasoning must place controls on these inferences. The most common way is to require that s and t be sufficiently *similar* (Hall 1989; Winston 1980), that is, that their common subsumer P be sufficiently specific. The more properties that are shared by the two objects, the more plausible it should be to expect them to share further properties. Gentner (1989) and Falkenhainer et al. (1989) require that P be "systematic," comprising many predicate symbols of high arity. Davies and Russell (1987) add expressions of the form

$$\forall x\, P(x) \Rightarrow Q(x) \quad \vee \quad \forall x\, P(x) \Rightarrow \texttt{not}\, Q(x) \tag{4.10}$$

to the background knowledge $\mathcal{T}$ in (4.8), and say that P *determines* Q. The presence of the source analogue s will then confirm the first disjunct in (4.10), and thereby the inference $(*)$ in (4.9).

Though predictive analogy is logically unjustified (like inductive reasoning), it is a powerful mode of inference which can be given some statistical justification. If a concept P has been instantiated by many objects, and if most of these also instantiate a concept Q, this lends statistical support to the conclusion that they often co-occur and that P may imply Q. These two statistical notions — *support* for a rule, and *confidence* in a rule — have been used by researchers in knowledge discovery (Agrawal et al. 1993). Piatetsky-Shapiro (1991; 1992) proposed a "rule interest" measure which orders discovered rules in the attempt to retain only highly predictive ones. For rules of the form $\forall x P(x) \Rightarrow Q(x)$, P and Q must fail a test for statistical independence: rules for which P and Q are independent are not interesting. In addition, rule interest decreases as $|\texttt{inst}(P)|$ increases without a concomitant increase in $|\texttt{inst}(P\,\texttt{and}\,Q)|$ ($\texttt{inst}(X)$ is the set of known instances of — or individual concepts subsumed by — the concept X). This will happen when P is an overly-general concept, and this effect lends statistical justification for choosing a source analogue of high similarity to the target, since these are precisely the ones which lead to specific antecedents for rules.

It is satisfying that there exists this connection between analogical reasoning and knowledge discovery, because both techniques use a database to make inferences. A computational model of analogical reasoning is concerned with drawing on a database of target analogues to make predictions, while knowledge discovery is concerned with discovering probabilistic rules from a database. The difference between the two is that knowledge discovery is concerned with creating and maintaining a knowledge base of rules, while analogical inference is concerned with indexing a database and *dynamically* creating predictive rules when presented with a single target case.

The goal of IMEM is to discover recurrent, specific structured patterns in a database and index these using a subsumption taxonomy. It supports close-match retrieval of source analogues, and subsequent *spatio-analogical* inferences (Conklin and Glasgow 1992). The inferences of spatial analogy can be divided into three types:

1. Classify an image.
2. Predict plausible locations for parts known to be present in an image.
3. Predict plausible labellings for uninterpreted parts of an image.

An example Type 1 inference — the classification of chess endgame patterns represented as image terms — was discussed in (Conklin and Glasgow 1992). It was shown that the efficiency of the indexing scheme was high relative to a sequential search over a flat database of patterns. The prediction of chemical structure from constitution is an example of a Type 2 inference, and is discussed in (Conklin et al. 1994) and in more detail in Part II of this dissertation. I presented the ideas behind Type 3 analogical inference at the AAAI Symposium on Machine Learning in Computer Vision (Conklin 1993). Experimental results on the labelling of small molecule electron density maps was reported there and in (Conklin et al. 1993a).

Part II

Knowledge Discovery in Molecular Structure Databases

In recent years there has been an explosion of information generated on molecular structures. This information is usually three-dimensional in nature, and is normally appended to the existing molecular database in the form of raw coordinate data. Buried in these databases are many rules and constraints of structural chemistry. There is a growing need for techniques to structure, manage, compress, assimilate, and extract explicit general knowledge from molecular structure data. Part I of this dissertation presented the theory and methods of structured concept discovery. The goal of Part II is to show how structured concept discovery can be applied to two large three-dimensional molecular structure databases, the Cambridge Structural Database (Allen et al. 1991a) (CSD) and the Protein Data Bank (Bernstein et al. 1977) (PDB).

It is necessary to introduce some definitions from structural chemistry and conformational analysis (Solomons 1988; Riddell 1980) for this part of the thesis. The *composition* of a molecule is given by the numbers and types of atoms present in the molecule. It is represented by a *molecular formula* (see Figure II.1). The *constitution* of a molecule is the bonding topology of its atoms. Constitution is represented by a *molecular graph*, with bonds as edges and atom types as vertices. A molecular formula may denote different *isomers*. Each isomer will have a different arrangement of its atoms in 3D (three-dimensional) space. *Constitutional isomers* have different molecular graphs, while *stereoisomers* have the same graphs, but different spatial positioning of certain atoms. More shall be said of stereoisomers in Section 5.2.4.

Molecules do not typically exist in a single rigid 3D arrangement, but rather are mobile and exist in a number of different possible forms: these are produced by rotations about single bonds and by small distortions in bond lengths and angles. Each 3D arrangement has a *potential energy*, which dictates its relative stability. There are many different inter- and intra- molecular forces which contribute to potential energy, including torsional forces, bond stretching, and non-bonded interactions. The *conformation space* (Crippen and Havel 1988) of a molecule is the set of its 3D arrangements, modulo rotation and translation, which obey reasonable constraints on bond lengths, bond angles, and torsion angles (these terms are defined below). The *conformations* or *conformation classes* of a molecule are the local energy minima in conformational space. A molecule may *interconvert* between different conformations. A conformation can be represented by a listing of atom types and Cartesian coordinates, or by a complete set of interatomic distances (Crippen and Havel 1988). One final term: the *enantiomorph* of a 3D arrangment is its 3D mirror image.

C13 H12 C12 N6 O2

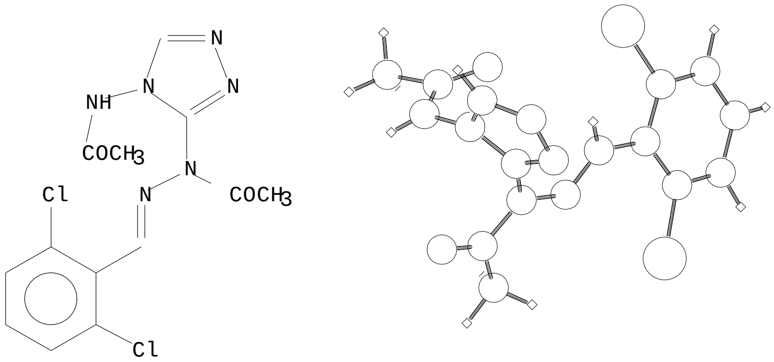


Figure II.1: Three representations for a molecular structure. Left: molecular formula; middle: molecular graph; right: a 3D arrangement. The structure is CSD refcode `AABHTZ`.

Understanding the biological activity and function of molecules requires detailed knowledge of molecular conformation. Information on the conformations of a structure with known constitution can be gained in various ways (see Willett 1991, for a good overview of this critically important problem). To date, most of the information on the 3D structure of molecules has been obtained through X-ray crystallography and crystal structure determination methods. While structure determination for small molecules can be done in a routine way, there are still substantial experimental and theoretical difficulties to be overcome before large macromolecules will be routinely solved with X-ray crystallography. In principle, a conformation can be predicted by *ab initio* quantum calculations, or by molecular mechanics methods (Burkert and Allinger 1982) which search for minimum-energy conformations (e.g., Shah and Dolata 1993). The computing time taken by such algorithms, however, normally increases exponentially with the number of atoms in the target molecule, prohibiting their use on many interesting structures. In both the crystal structure determination and the molecular mechanics approaches, predictions about the conformation of significant substructures can be invaluable. Predicted structural motifs can be incorporated into the direct methods of X-ray crystallography (Fortier et al. 1993). The *structure generation* problem (Pearlman 1993) relies on predicting conformations that are reasonably close to energy minima before simplified numerical molecular mechanics routines are invoked.

There are two approaches to structure generation that are of particular relevance to this thesis. They are based on symbolic logic techniques, and they both strive to avoid expensive minimum-energy searches over conformation space by reasoning using previously-solved conformations of substructures. The first system, WIZARD (Dolata 1986; Dolata et al. 1987; Leach et al. 1988), uses predefined 3D *conformational units* which are matched against substructures of the molecular graph under consideration using a subsumption taxonomy of graphs. Typical conformational units include 3D arrangements of cyclohexane, heterocyclic and fused rings, and small acyclic structures. The input graph is parsed for large conformational units, and these are joined together in a depth-first search process to build predictions. Conformational units are computed using either energy minimization or statistical clustering of exemplar 3D arrangements drawn from the CSD. As Allen et al. (1991b) note, the definition of conformational units is an excellent problem for clustering techniques, since these may reduce the number of exemplar 3D arrangements to a few preferred conformations.

The second approach to structure generation, AIMB (Analogy and Intelligence in Model Building) (Wipke and Hahn 1986), is similar to WIZARD in that previously-determined structures in the CSD are used to make predictions about new conformations. The hypothesis of AIMB is *if two molecules have similar molecular graphs, they might also have similar conformations*. AIMB uses the input molecular graph to invoke a substructure search of the CSD. If a match is obtained, a conformation is predicted for the matching portion. If not, the procedure is recursively invoked using smaller substructures of the molecular graph. The predicted 3D substructures are assembled into a complete prediction of conformation. Energy minimization techniques can then be used to refine the predicted models.

Both of the techniques discussed above use a form of analogical reasoning (see Section 4.4) to attack the structure generation problem for small molecules. For large molecules, the related area of protein model building by homology (Thornton and Swindells 1993; Blundell et al. 1987) relies on searching a 3D database for molecules with similar amino acid sequences (constitution) and inferring that the unknown structure has a similar conformation. The exciting results of Jones and Thirup (1986) have indicated that the 3D structure of proteins can often be built from substructures of known proteins. Thus an initial and necessary step to molecular-analogical reasoning is the classification and organization of structural data so as to permit efficient and rapid retrieval of useful molecular analogues.

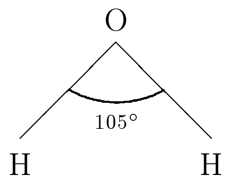
Various *quantitative* and *qualitative* relational attributes of molecular structures can be used to describe conformation classes. As noted by Allen et al. (1992), these relations can be computed from the underlying 3D conformational data, and should be "internal" or invariant under translation and rotation operations. A quantitative relation can be given by a real–valued function. A qualitative relation, on the other hand, is Boolean-valued. Table II.1 lists some quantitative and qualitative relational attributes used for the

Relational attribute	Arity
interatomic distance	2
bond angle	3
torsion angle	4
bonded	2
proximal	2
linear	3
angular	3
syn, anti-periplanar	4
+, -clinal	4
orientation	4
R/S	5
cis/trans	> 5
axial/equatorial	> 5

Table II.1: Relational attributes for molecular conformation. Top: quantitative, Bottom: qualitative attributes. The *orientation* relation is presented by Wirth (1986). The periplanar and clinal relations are presented by Klyne and Prelog (1960). The other relations can be found in textbooks on organic chemistry (e.g., Solomons 1988).

expression of molecular conformation. The *distance* between two atoms is the standard Euclidean distance, measured in Angstroms (Å). The *bond* or *valence* angle among three connected atoms A, B, C is the angle between the vectors AB and BC. Finally, the *torsion* or *dihedral* angle among a connected chain of four atoms A, B, C, D is the angle between the normals to the planes ABC and BCD. The torsion angle is called *improper* if the four atoms do not form a connected chain. These quantitative attributes can also be used to define many different qualitative relations. For example, the *bonded* relation, although normally stored in a connection table (Allen et al. 1991a), can be defined as an interatomic distance range. The *linear* and *anti-periplanar* relations can be based on bond angle and torsion angle ranges, respectively.

Aspects of molecular conformation are amenable to a structured concept representation. Any of the qualitative attributes of Table II.1 can form an underlying first-order language (Chapter 2) for structured concept discovery. Molecular conformation can be represented using an $\mathcal{SDL}$ image term, and qualitative relations can be expressed as functions over image components. For example, Figure II.2 illustrates the water molecule — encountered in Chapter 2 — as an $\mathcal{SDL}$ concept. This part of the thesis will illustrate the application of the $\mathcal{SDL}$ representation language and the IMEM discovery method to the classification of small molecules (Chapter 5) and macromolecules (Chapter 6). The problem common to both chapters is *conformational clustering*: given a graph representing some molecular




```
defprimconcept hydrogen any;
defprimconcept oxygen any;
defconcept water (image (
                     (hydrogen (0.77 16.76 2.54))
                     (hydrogen (0.49 15.86 3.68))
                     (oxygen   (0.14 16.20 2.90)))
                 (bonded angular));
```

Figure II.2: A water molecule in $\mathcal{SDL}$. Left: molecular graph depicting two hydrogens bound to an oxygen atom. Right: an $\mathcal{SDL}$ representation of the water molecule. The 3D coordinates are from CSD entry AAGAGG10.

constitution, and a set of examples of 3D arrangements of that graph, group the examples into classes characterized by structured concepts. It is reasonable to assume that well-characterized classes correspond to potential energy minima in conformational space.

Despite the common task of conformational clustering, there are several significant differences between the CSD and PDB experiments. First, the PDB datasets are two orders of magnitude larger than the CSD ones. Thus it is simply not possible to comprehend them by hand and eye. Second, the PDB experiments make more extensive use of the $\mathcal{SDL}$ knowledge representation formalism (Chapter 3) and similarity theory (Chapter 4), encoding primitive concepts as background knowledge. This knowledge is essential in producing adequate characterizations of the clusters in data. Third, there is no single accepted classification of the protein fragments used in Chapter 6. Whereas the CSD results can be validated against accepted chemical classifications, the PDB discoveries will be evaluated according to the strength of derived analogical inference rules. These rules predict the structure (conformation) of polypeptide fragments from their amino acid sequence (constitution).

Chapter 5

Knowledge Discovery in the Cambridge Structural Database

The Cambridge Structural Database (Allen et al. 1991a) is a large database of experimental crystal structure determinations of organo-carbon compounds. Each CSD entry comprises bibliographic data, chemical constitution and 3D atomic coordinates for a molecular structure. The CSD is continually expanding, and at the end of 1994 contained roughly 130,000 entries. Integrated with the CSD is a suite of search and data analysis tools. The QUEST program performs 2D substructure searching; finding all entries which contain, as a subgraph, a query molecular graph. To avoid many costly subgraph isomorphism tests, the CSD uses sophisticated *screening* mechanisms to filter out structures from the database which could not possibly contain the query graph. A facility for *similarity* searching returns molecular graphs "similar" to the query graph. The topics of similarity and screening will be revisited later in this chapter. The GSTAT (geometry-statistics) program handles 3D substructure searching; queries are molecular graphs annotated with distance, bond angle, and torsion angle range constraints. The 3D searching modules from GSTAT have recently been upgraded and blended with the QUEST program to extend the 2D substructure searching capabilities to the 3D arena; the new program is designated QUEST3D.

The CSD contains a large amount of raw molecular data. In it are hidden many rules and constraints of structural chemistry, and recurrent 3D patterns. There is a recognized need for the integration of knowledge discovery tools with the CSD (Allen et al. 1992). Research by Allen et al. in this regard (references throughout this chapter), has led to a methodology for conformational clustering of 3D molecular structures. First, the program QUEST3D is used to retrieve instances of a molecular graph. The retrieved structures are then annotated by GSTAT with quantitative relational attributes. The resultant feature lists are processed by various numerical clustering routines. This yields a number of prototypical fragments which, it is hoped, define the spectrum of discrete, preferred conformation classes for the particular query graph. As noted by Allen et al. (1993b), these 3D prototypes can be used for structure generation and model building and as 3D screens.

The numerical clustering approach of Allen et al. has been applied with success to many small molecule datasets. There are, however, three main problems with their approach which must be resolved if a more generally applicable and fully automated technique is desired. First, the numerical clustering approach requires that the number of conformation classes be decided upon by a user, since finding suitable and robust criteria for the autoselection of stopping points is a problem. Second, there is a vast number of ways in which a query graph can be annotated with a set of quantitative attributes, and it is not always apparent which features should be used to produce a desired clustering. Third, the results of a clustering are not retained and linked with previously discovered concepts, and these concepts cannot be incrementally refined. The IMEM methodology is a step towards removing these limitations. Aside from the issue of data presentation order, the lowest-level discovered concepts are fully determined by the data in IMEM. As outlined in Chapter 3, an image term represents a complete relational graph, and it is not necessary to annotate individual groups of atoms with parameters. Although it is still necessary to choose which relations to use, this chapter will illustrate that this is not likely to be a problem in practice. Finally, an $\mathcal{SDL}$ knowledge base comprises structured concepts linked to individual data items. New items can be classified into this knowledge base, leading to new concept formation.

This chapter will argue that the IMEM methodology represents a complementary approach to the numerical clustering work referred to above, and could be an important future knowledge representation and discovery tool for the CSD. As a basis for this argument, Section 5.2 will illustrate the application of the IMEM structured concept discovery approach to a handful of CSD datasets. Four datasets have been chosen: they are listed in Table 5.1. Each has been previously analyzed by Allen et al. (second column of Table 5.1) using Jarvis-Patrick (1973) and single-linkage clustering schemes. In addition, Conklin et al. (1992) have previously applied conceptual clustering to the hexopyranose dataset. Section 5.1 will outline the common methodology that was used for these datasets, including a detailed discussion of two slight specializations made to the basic IMEM algorithm. Section 5.2 will present and evaluate the discoveries of IMEM from the CSD datasets. Section 5.3 will present some conclusions and defend the claim that IMEM could be a valuable

#	Name	Reference	Nf	Topological symmetry	Intermediate concepts
1	six-member carbocycles	ADT2	222*	12	no
2	cycloheptane	AHP (7C1)	101	14	no
3	steroid C17 sidechain	ABW	151	2	no
4	hexopyranose	AF	249	1	yes

Table 5.1: CSD datasets used in this chapter. *: denotes a subset of all CSD instances. References; ADT2: (Allen et al. 1991c), AHP: (Allen et al. 1993a), ABW: (Allen et al. 1995), AF: (Allen and Fortier 1993). Nf: number of examples.

addition to the CSD data analysis tools.

5.1 Method

The IMEM method (Chapter 4) was applied to the CSD datasets listed in Table 1. Two slight specializations were made to the basic algorithm. The first is only for efficiency purposes; the second changes the semantics of $\mathcal{SDL}$ slightly. First, the 2D *atomic permutation symmetry group* is taken into account, for each dataset. The atomic permutation symmetry group represents the bonding and atom type-preserving isomorphisms from a 2D structure onto itself (Gray 1986; Randic et al. 1981).[1] This leads to an efficient similarity computation between concepts when a similarity threshold of $\tau = 1$ is used. It follows from the definitions of similarity (Chapter 4) that for any two image terms $[A, R]$ and $[B, R]$, where connectivity is one of the relations in R, $S_{\mathcal{T}}([A, R], [B, R], f) = 1$ only if f is an element of the atomic permutation symmetry group. Possessing the symmetry group is invaluable when trying to find isomorphisms between two 3D structures with the same 2D graphs because any isomorphism which preserves connectivity must be one of the elements of the symmetry group. Thus to find an R-isomorphism it is only necessary to check, for each element of the atomic permutation symmetry group, whether it preserves all relations in R other than connectivity. This can be done simply by comparing an appropriately-permuted listing of all other relation tuples in the two structures. The atomic permutation symmetry group and the relational tuples are computed only once at the outset of a clustering

[1] An isomorphism between a structure and itself is sometimes called an *automorphism* or a *permutation*. A *group* is a set, together with an associative binary operation which has an identity element, where every element in the set has an inverse. The set of permutations together with function composition is a group: function composition is associative, there is an identity permutation, and every permutation has an inverse which undoes its effect. A group is represented internally simply as a list of permutations (see Figure 5.1).

experiment.[2]

A second specialization of the basic IMEM algorithm was made to properly accommodate enantiomorphs (3D mirror images of structures). For a given structure, it is often indeterminate which enantiomorph is described in the CSD, and the two enantiomorphs should be considered formally equivalent (Allen et al. 1991a). In $\mathcal{SDL}$, this has the effect that two image terms $[A, R]$ and $[B, R]$ are structurally equivalent, $[A, R] \cong_R [B, R]$, if there is an R-isomorphism between A and B, *or* if $[A, R] \cong_R [\mathtt{invert}(B), R]$.[3] An R-isomorphism between two image terms, whether or not an inversion is considered, must be one of the elements of the atomic permutation symmetry group. This is because the `bonded` relation is invariant under inversion.

Only a few of the qualitative relations of Table II.1 will be used by IMEM for the CSD datasets. A relation used in all datasets is that of atomic connectivity. Letting $\mathtt{erad}(a)$ be the elemental radius[4] of an atom of type a, the `bonded` relation is defined as:

$$\begin{aligned}\mathtt{bonded(x,y)} \quad \stackrel{\text{def}}{=} \quad & \mathtt{erad(first(x))} + \mathtt{erad(first(y))} - \alpha \leq \\ & \mathtt{d(second(x), second(y))} \leq \\ & \mathtt{erad(first(x))} + \mathtt{erad(first(y))} + \alpha\end{aligned}$$

where `d` is the Euclidean distance function.[5] Two atoms are bonded if their interatomic distance is less than the sum of their atomic radii, plus a constant factor α. Taking the elemental radius of carbon to be 0.68, and setting $\alpha = 0.4$, determines that two carbons are bonded if their distance is between $0.68 + 0.68 - 0.4$ and $0.68 + 0.68 + 0.4$ (0.96 to 1.76) Å. Since oxygen also has an atomic radius of 0.68, the same range is used for carbon-oxygen and oxygen-oxygen bonding. This same definition for crystallographic connectivity is used by the CSD software. Note that the `bonded` relation depends on the types of the atoms given as parameters.

All of the molecular graphs exhibited by the CSD datasets have been previously classified manually, and automatically using numerical clustering methods. Thus, for each CSD dataset it is possible to ask two questions: "are the archetypal classes representable by

[2]More precisely, considering only the relation of connectivity, let f be the *initial isomorphism* between two structures C and D. This isomorphism f is given implicitly by the initial ordering of components in each structure. Any other isomorphism between C and D can be expressed as a composite function $g; f$ for some g in the atomic permutation symmetry group.

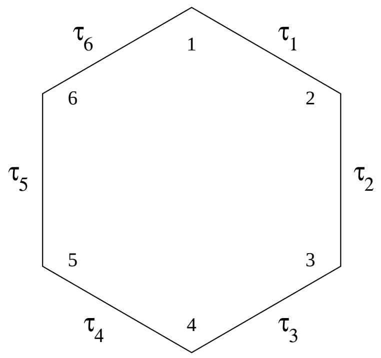
[3]In crystallographic terms, the `invert` operation inverts a structure about an inversion center.

[4]Detailed tables of atomic radii are given in Appendix E of (CCDC 1994).

[5]Formally, $\mathtt{d}(x, y) \stackrel{\text{def}}{=} \|x - y\|$, where x and y are vectors of the same dimension, and $\|\cdot\|$ is the vector norm function.

Atoms	Attribute
1 2 3 4	τ_2
2 3 4 5	τ_3
3 4 5 6	τ_4
4 5 6 1	τ_5
5 6 1 2	τ_6
6 1 2 3	τ_1

Table 5.2: The definition of six–member carbocycle torsional attributes




```
1 2 3 4 5 6
2 3 4 5 6 1
3 4 5 6 1 2
4 5 6 1 2 3
5 6 1 2 3 4
6 1 2 3 4 5
6 5 4 3 2 1
5 4 3 2 1 6
4 3 2 1 6 5
3 2 1 6 5 4
2 1 6 5 4 3
1 6 5 4 3 2
```

Figure 5.1: The six–member carbocycle, with its symmetry group

a small set of disjoint $\mathcal{SDL}$ concepts?" (the question of *representational* adequacy), and "are there sufficient numbers of instances of these concepts in the data?" (the question of *empirical* adequacy).

5.2 Results

5.2.1 Six-member carbocycles

The six-member carbocycle adopts a small number of typical conformations, and conformation classes discovered by an unsupervised learning procedure can be validated against known classes. A dataset of 222 six-member rings, chosen to exhibit a broad spectrum of chemical environments and conformations, has been retrieved from the CSD and previously analyzed using various numerical clustering procedures (Allen et al. 1991b; Allen et al. 1991c). This dataset is re-analyzed here using IMEM, and compared with the Jarvis-Patrick numerical clustering scheme of Allen et al. (1991c) (henceforth ADT2).

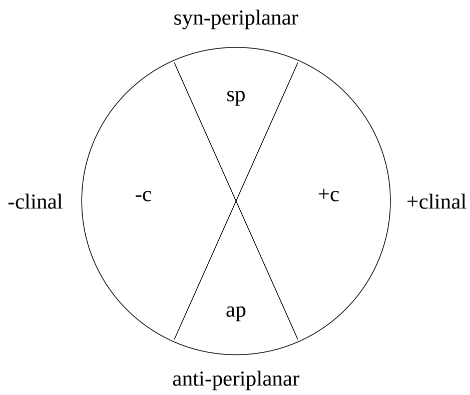


Figure 5.2: The Klyne-Prelog relations, which partition torsion angle space to form four qualitative relational attributes.

Figure 5.1 displays the 2D topology of the six-member ring. For the purposes of numerical clustering, ADT2 use six intra-annular torsion angles $\tau_1, \ldots, \tau_6$ (see Figure 5.1 and Table 5.2). The six attributes τ_1 through τ_6 take on different values with each application of a 2D symmetry permutation, and this introduces complications for numerical clustering routines which require a metric of dissimilarity between examples. Since the six-member ring has extensive permutational symmetry (Figure 5.1), the similarity of two structures under one permutation may be different from their similarity with respect to another permutation. The solution presented by ADT2 is to choose as a measure of distance the minimum over all permutations. This is called "symmetry-modified" clustering. The semantics of $\mathcal{SDL}$ removes the need for such explicit modifications, since a search is automatically performed over the space of partial R-monomorphisms to compute similarity. However, as discussed above, the symmetry group can sometimes be employed for efficiency purposes.

To represent six-member rings, cycloheptanes, and steroid sidechains in $\mathcal{SDL}$, we use the Klyne-Prelog relations, which partition the circular torsion angle space into four regions (Klyne and Prelog 1960) (see Figure 5.2). The partitioning gives rise to four qualitative spatial relations, or equivalently, one functional relation (Chapter 2). A connected chain of four atoms is planar if their torsion angle is in the sp or ap ranges. Klyne and Prelog used a planarity value of 30° in their paper, so that sp is -30 to 30, and ap is 150 to 210. Tests with IMEM have shown, however, that the value of 30° is not equally appropriate for all problems because it often overgeneralizes planarity. Thus the Klyne-Prelog framework has been used to define a *family* of functional relations, each given by a particular choice of a planarity parameter γ (see Figure 5.3).

Table 5.3 lists the seven standard canonical conformations of a six-member ring. These

```
sp(p,q,r,s)  ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
           t(second(p),second(q),second(r),second(s)) ≥ 360 − γ or
           t(second(p),second(q),second(r),second(s)) < γ

+c(p,q,r,s)  ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
           γ ≤ t(second(p),second(q),second(r),second(s)) < 180 − γ

ap(p,q,r,s)  ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
           t(second(p),second(q),second(r),second(s)) ≥ 180 − γ and
           t(second(p),second(q),second(r),second(s)) < 180 + γ

-c(p,q,r,s)  ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
           180 + γ ≤ t(second(p),second(q),second(r),second(s)) < 360 − γ
```

Figure 5.3: Definitions for the Klyne-Prelog relations. `t` is the torsion angle function.

Name	Standard concept						$\mathcal{SDL}$ concept					
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6
phenyl	0	0	0	0	0	0	sp	sp	sp	sp	sp	sp
chair	60	-60	60	-60	60	-60	+c	-c	+c	-c	+c	-c
boat	0	-60	60	0	-60	60	sp	-c	+c	sp	-c	+c
twist-boat (tb)	33	33	-70	33	33	-70	+c	+c	-c	+c	+c	-c
sofa	30	0	0	-30	60	-60	+c	sp	sp	-c	+c	-c
half-chair (hc)	45	-15	0	-15	45	-62	+c	sp or -c	sp	sp or -c	+c	-c
screw-boat (sb)	40	0	-22	0	40	-60	+c	sp	sp or -c	sp	+c	-c

Table 5.3: Archetypal six-member carbocycles, from (Allen and Taylor 1991, Table 1), with corresponding $\mathcal{SDL}$ concepts.

Id	Discovered concept						Standard name(s)	Frequency
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6		
1	sp	sp	sp	sp	sp	sp	phenyl	35
2	sp	-c	+c	sp	-c	+c	boat	52
3	+c	-c	+c	-c	+c	-c	chair	64
4	+c	+c	-c	+c	+c	-c	tb	8
5	-c	sp	sp	sp	-c	+c	hc	5
6	-c	+c	sp	+c	-c	+c		5
7	sp	-c	+c	+c	-c	+c	boat-like	12
8	-c	sp	+c	sp	-c	+c	sb	6
9	-c	sp	sp	+c	-c	+c	hc, sofa	29
10	+c	-c	+c	-c	-c	+c	boat-like	3
11*	sp	sp	-c	+c	sp	sp	phenyl-like	1

Table 5.4: The six-member carbocycle conformations discovered by IMEM. *: an individual concept.

were derived by energy minimization calculations and appear in (Allen and Taylor 1991, Table 1). The rightmost column lists the $\mathcal{SDL}$ *structural sequence* for the canonical series of torsion angles: a listing of the corresponding Klyne-Prelog relations. When there is some ambiguity about the structural sequence, that is, when a canonical angle is close to a boundary, this is noted with a disjunction of possibilities. It can be seen that $\mathcal{SDL}$ unequivocally represents the four major conformation classes of phenyls, chairs, boats, and twist-boats. That is, no $\mathcal{SDL}$ concept for any of these archetypes could possibly denote another archetype. However, the half-chair concept subsumes the sofa concept in $\mathcal{SDL}$. These two conformations are actually quite close in conformational space (Allen and Taylor 1991). Also, some screw-boats could be interpreted as half-chairs in the $\mathcal{SDL}$ representation, if τ_3 is sp. Again, these two conformations are close in conformational space, so this occasional equivocation is permissible.

Having considered the question of representational adequacy for the six-member rings, we now turn to the question of empirical adequacy. The IMEM method was used to cluster 222 six-member rings, using a similarity threshold of $\tau = 1$, and a planarity value of $\gamma = 15°$. Ten concepts, covering 219 instances, were created. Table 5.4 presents the discovered concepts. The second column shows the structural sequence under a permutation from the symmetry group chosen to match the standard conformation classes listed in Table 5.3. The third column gives a standard name to the discovered concept. These were derived from a comparison with the archetypal concepts given in Table 5.3. The three singletons in the IMEM clustering are instances 58 (`BABPIP`), 116 (`BEWNOG`), and 21

Class	ADT2 Id	Np	IMEM Id	Np
phenyl	1	35	1	35
boat	2 3 4 5 6	63	2 7 10	67
chair	7 8 9	59	3	64
twist-boat	12	9	4	8
half-chair/sofa/screw-boat	10 11	38	5 6 8 9	45

Table 5.5: Concept discovery results for the six-member carbocycles: ADT2 vs. IMEM. Id: concept identifier; Np: combined membership of concepts.

(`ACCITR10`). Instance 58, while being a reasonable-looking boat, has a transannular carbon 1-3 distance of 1.51 Å, and instance 116 has a carbon 1-4 distance of 1.56 Å. Both of these values are below the upper limit of carbon-carbon bonding distance (1.76 Å), which indicates that their crystallographic coordinates are in error. Since isomorphisms in IMEM had to preserve the `bonded` relation, these two individuals are not instances of any other concept. The third singleton, instance 21, has torsion angles $(5, 7, -18, 17, -6, -6)$ and is very close to a phenyl conformation.

Using the same dataset of 222 six-member carbocycles, the Jarvis-Patrick numerical clustering method used by ADT2 created 14 clusters with membership greater than one, including 2 doublet clusters, and 14 singletons. To compare the results of ADT2 and IMEM, the 12 clusters of ADT2 with membership greater than 2 were compared with the 10 IMEM concepts. The 12 clusters of ADT2 are given in Table 5 of their paper, and are reproduced here in Table 5.5. A comparative analysis of the ADT2 and IMEM clusterings reveals similarities. These can be considered for each archetypal concept:

phenyl The clustering results of ADT2 and IMEM are identical, with the exact same 35 instances assigned to the phenyl class.

boat ADT2 use 5 concepts to describe 63 boats. IMEM uses 3 concepts to describe 67 boats. All of the entries of the 5 ADT2 clusters appear in the 3 IMEM concepts, except for the suspicious instance 58 (a singleton in IMEM). The 5 extra instances in

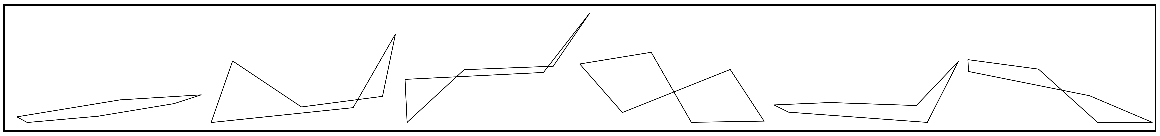


Figure 5.4: Some six-member carbocycle concepts discovered by IMEM. From left to right: phenyl (concept 1 of Table 5.4), boat (2), chair (3), twist-boat (4), half-chair/sofa (9), and screw-boat (8).

the 3 IMEM boat concepts are (IMEM Id : ADT2 Id in brackets):

instance 83 (`BEHFAV`): $(-41, 78, -15, -60, 103, -37)$ $(7 : 12)$

instance 85 (`BEHFAV`): $(-46, 80, -15, -60, 99, -31)$ $(7 : 12)$

instance 120 (`BEWNOG`): $(22, 30, -32, -17, 60, -62)$ (10 : singleton)

instance 48 (`ACONTN10`): $(-22, 27, 20, -67, 73, -30)$ (10 : singleton)

instance 175 (`AMHPEN10`): $(-16, 16, 3, -21, 21, -3)$ (2 : doublet)

chair ADT2 find 59 chairs, IMEM 64. The three ADT2 chair clusters are completely contained within the single IMEM concept. The five extra IMEM chairs — all singletons in ADT2 — are:

instance 20 (`ACAMYA`): $(56, -57, 55, -50, 49, -53)$

instance 34 (`ACLYCA10`): $(55, -63, 63, -54, 43, -44)$

instance 76 (`BCYLON10`): $(65, -70, 70, -53, 45, -56)$

instance 117 (`BEWNOG`): $(66, -92, 89, -92, 62, -44)$

instance 181 (`BEWNOG`): $(28, -22, 37, -54, 59, -47)$

twist-boat As mentioned above, ADT2 place two IMEM boats (83 and 85) in this class. IMEM places instance 78 (`BCYLON10`) $(-31, -30, 56, -18 - 44, 71)$ — a singleton in ADT2 — in the twist-boat concept. The ADT2 and IMEM clusters are otherwise identical.

half-chair/sofa/screw-boat The two methods differ the most on these classes. ADT2 discover 29 half-chairs, and lump 9 instances in a sofa/screw-boat class. IMEM lumps together 39 instances in a half-chair/sofa class, but also discovers a distinct screw-boat concept concepts covering 6 instances. There are four possible $\mathcal{SDL}$ concepts for the half-chair (see Figure 5.3), and only two of these occur in the dataset.

Figure 5.4 depicts the $\mathcal{SDL}$ image for each archetypal concept discussed above.

The proportions of the archetypal conformations in the 222 six-member carbocycle dataset are not representative of the whole CSD. To clarify this point, QUEST3D was used to extract all rings of six carbons connected by single bonds from the CSD, regardless of their chemical environment. Only structures with an R-factor less than or equal to 0.01 were retrieved. This search produced 16466 examples. Processing this dataset using IMEM gave the following results:

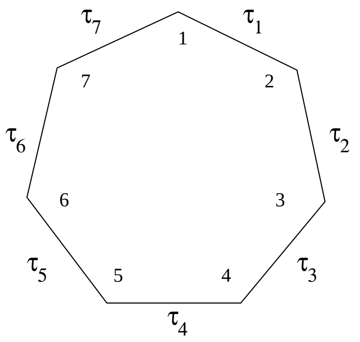
chair	11892	(72.2%)
boat	2030	(12.3%)
tb	503	(3.1%)
planar	103	(0.6%)
errors	289	(1.9%)

The chair is the energetically preferred conformation of a six-member carbocycle, and this is reflected in its relative population.

5.2.2 Cycloheptane

The cycloheptane dataset (7C1 in Allen et al. 1993a, henceforth AHP) is processed by IMEM in much the same way as the six-member carbocycles: the only differences are the ring size and hence the order of the permutation symmetry group (see Figure 5.5). A planarity value of $\gamma = 15°$ was also used to represent cycloheptane conformations. The four archetypal cycloheptanes are listed in Table 5.6. These were obtained from AHP, Table 1, and were derived by energy minimization calculations. As Table 5.6 illustrates, the twist-boat concept subsumes the boat concept in $\mathcal{SDL}$, that is, $\mathcal{SDL}$ cannot uniquely represent the twist-boat class using the Klyne-Prelog relations. Aside from this minor equivocation, $\mathcal{SDL}$ can adequately represent the major cycloheptane conformations.

AHP extracted a dataset of 101 cycloheptane structures from the CSD. They note that there are three main conformational subgroups in this particular dataset: chairs, twist-chairs, and boats/twist-boats. Their Jarvis-Patrick numerical clustering procedure produced 4 clusters with membership greater than two, 2 clusters with two instances, and 9 singletons. IMEM created 5 concepts with four singleton or individual concepts (see Table 5.7). One singleton, instance 87 (CSD refcode `SILYIV`) was subsequently found to have crystallographic coordinates in error, with a transannular carbon 1-3 distance of 1.52 Å. The other three singletons are instances 100 (CSD refcode `HDAPTX`) $(-31, 50, -5, 0, -54, 102, -48)$, 17



1	2	3	4	5	6	7
2	3	4	5	6	7	1
3	4	5	6	7	1	2
4	5	6	7	1	2	3
5	6	7	1	2	3	4
6	7	1	2	3	4	5
7	1	2	3	4	5	6
7	6	5	4	3	2	1
6	5	4	3	2	1	7
5	4	3	2	1	7	6
4	3	2	1	7	6	5
3	2	1	7	6	5	4
2	1	7	6	5	4	3
1	7	6	5	4	3	2

Figure 5.5: The cycloheptane, with its symmetry group

(BUHXAD) $(-32, 83, -52, -36, 76, -35, 1)$, and 57 (HYMINA) $(-21, 75, -22, -64, 55, 29, -53)$. A visual inspection revealed that instance 100, concept 6* in Table 5.7, looks very much like a half-chair, due to its two contiguous sp relations (see Figure 5.6).

As with the six-member carbocycles, the results of IMEM can be compared with those of AHP for each archetypal class (see Table 5.8):

twist-chairs IMEM discovers 56 twist-chairs, AHP 48. Of the 8 extra IMEM twist-chairs, 6 appear in AHP doublet clusters or as singletons, and instances 19 (CEBBUG) $(63, -88, 59, 18, -78, 85, -62)$ and 32, (DIPHEP10) $(67, -85, 55, 16, -74, 76, -59)$ are classified by AHP as chairs. Both of these have a τ_4 torsion angle just slightly in the +c range, causing them to fall under the $\mathcal{SDL}$ twist-chair concept.

chairs As mentioned above, AHP classify two IMEM twist-chairs as chairs. IMEM also adds instance 15 (BOLWUU), a member of a doublet cluster in AHP, to the chair class.

boats The boat clusters of AHP and IMEM are identical.

Figure 5.6 depicts the $\mathcal{SDL}$ image for the three archetypal concepts discussed above, and for the half-chair instance.

Exo- and heterocyclic seven-member rings

In a second study of seven-member rings, Allen et al. (1994) (henceforth AHPV) considered exo- and heterocyclic rings. Figure 5.7 illustrates three of these structures which are analyzed here by IMEM. Note that their topological symmetry is constrained by the presence of

Name	Standard concept							$\mathcal{SDL}$ concept						
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7
chair	64	-84	66	0	-66	84	-64	+c	-c	+c	sp	-c	+c	-c
tc	54	-72	88	-39	-39	88	-72	+c	-c	+c	-c	-c	+c	-c
boat	-58	-31	70	0	-70	31	58	-c	-c	+c	sp	-c	+c	+c
tb	-64	-18	75	-18	-64	45	45	-c	sp or -c	+c	sp or -c	-c	+c	+c

Table 5.6: Archetypal cycloheptanes, from (Allen et al. 1993a, Table 1), with corresponding $\mathcal{SDL}$ concepts.

Id	Discovered concept							Standard name	Frequency
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7		
1	+c	-c	+c	-c	-c	+c	-c	tc	56
2	+c	+c	-c	sp	-c	+c	-c	chair	29
3	-c	-c	+c	sp	-c	+c	+c	tb,boat	10
4	-c	sp	+c	sp	-c	+c	sp	boat-like	2
5*	-c	-c	+c	-c	-c	+c	+c	boat-like	1
6*	-c	sp	sp	+c	-c	+c	+c	novel	1
7*	sp	-c	+c	-c	-c	+c	-c	tc-like	1

Table 5.7: The cycloheptane conformations discovered by IMEM. *: individual concepts.

	AHP		IMEM	
Class	Id	Np	Id	Np
twist-chair	1 3	48	1	56
chair	2	30	2	29
boat/twist-boat	4	10	3	10

Table 5.8: Concept discovery results for the cycloheptanes: AHP vs. IMEM

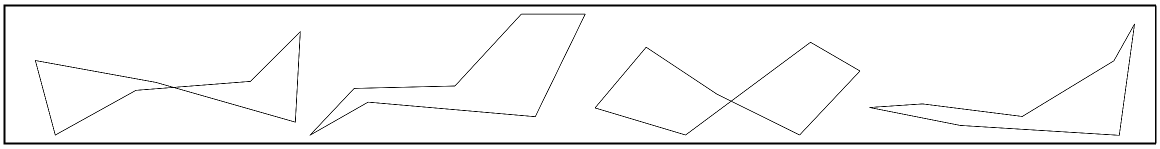
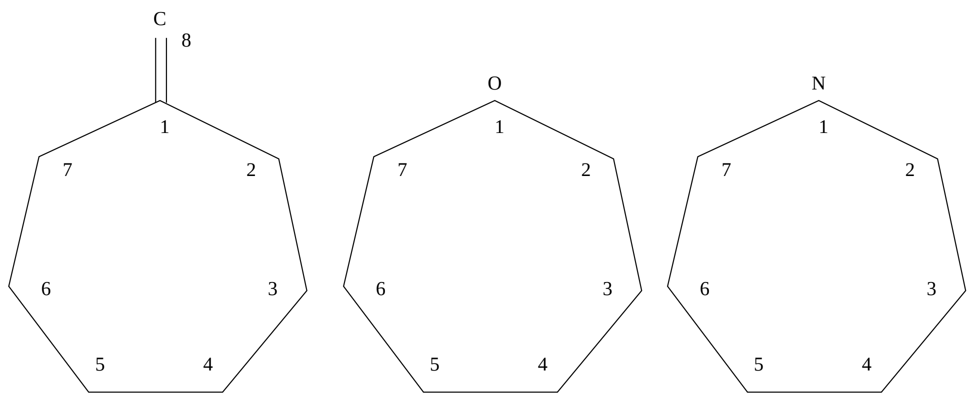


Figure 5.6: Some cycloheptane concepts discovered by IMEM. From left to right: twist-chair (concept 1 of Table 5.7), chair (2), twist-boat (3), and half-chair (6*).

II	V	VI



1 2 3 4 5 6 7 8	1 2 3 4 5 6 7	1 2 3 4 5 6 7
1 7 6 5 4 3 2 8	1 7 6 5 4 3 2	1 7 6 5 4 3 2

Figure 5.7: Exo- and heterocyclic seven-member rings, with symmetry groups

an exocyclic atom at C1 (II), or a heteroatom at position 1 (V, VI). The archetypal classes for these structures are the same as for the cycloheptane structure (Table 5.6), except that there are four possible forms of each archetype. These are produced by an axis of symmetry passing through one of the pairs of atoms 4-5, 3-4/5-6, 2-3/6-7, or 1-2/7-1 (Allen et al. 1993a).

The IMEM method was applied to three datasets: II (64 examples), V (50 examples) and VI (26 examples), using a planarity value of $\gamma = 15°$. AHPV applied a Jarvis-Patrick clustering to these three datasets. The cluster population results for IMEM and AHPV are given in Table 5.9; the two methods are seen to be in close agreement.

5.2.3 Steroid C17 sidechain

The six-member carbocycle and the cycloheptane datasets were chosen to illustrate high topological symmetry (orders 12 and 14, respectively) combined with the conformational constraints imposed by ring cyclicity. The steroid C17 sidechain (IV in Allen et al. 1991d) is a structure unconstrained by ring cyclicity, and its chain form leads to low-order topological symmetry (see Figure 5.8). The protein motifs considered in Chapter 6 have a similar form.

The Klyne-Prelog relations are also used to represent steroid C17 sidechain conformations. However, a planarity value of $\gamma = 15°$, as used for the previous two datasets, was

Class	AHPV	IMEM
	II	
TC2	14	15
TC3	15	15
C1	12	12
TC4	8	8
C2	2	2
	V	
TC3	17	17
TC2	12	16
C2	8	6
C3	5	4
	VI	
TC3	11	11
C1/TC4	6	6

Table 5.9: Exo- and heterocyclic 7-rings: AHPV vs. IMEM

found to be inappropriate. In the steroid sidechains, many of the 180° values in the second column of Table 5.10 fluctuate well outside of an ap range of $[165, 195]$, as defined by $\gamma = 15°$. Torsion angles in ring systems are more constrained than in fully flexible acyclic chains. Thus it is quite sensible to allow a wider definition of planarity for the steroid sidechain structure.

The biological function of a steroid structure is highly correlated with its conformation. The conformations of the steroid C17 sidechain have been analyzed manually by Duax et al. (1980), leading to a set of archetypal classes. These are reproduced in (Allen et al. 1991d) and here in Table 5.10. The corresponding $\mathcal{SDL}$ concepts, using a planarity value of $\gamma = 30°$, are in the rightmost column of the table. In contrast to the previous two datasets analyzed in this chapter, there is no equivocation of canonical forms in $\mathcal{SDL}$ for the C17 sidechain. Each class has a unique $\mathcal{SDL}$ description. Processing a dataset of 151 C17 sidechain examples with IMEM yielded 12 concepts with two or more instances, and four individual concepts (see Table 5.11). All of the singletons — instances 44 (CSD refcode `CHOCAL`), 79 (`CHOMYS`), 43 (`CHOBRH10`), and 69 `CHOLEU01` — were subsequently found to have crystallographic coordinates in error, with the distance between carbons 8 and 9 in the first three being within bonding distance at 1.09, 1.73, and 1.75 Å, and the distance between carbons 7 and 8 in instance 69 being out of bonding range at 0.91 Å. These individuals are omitted from Table 5.11. Concepts 9 through 12, covering 14 instances, are minor variations on the major classes, as noted in the table. As Table 5.10 illustrates, there are two possible A and D2 classes in $\mathcal{SDL}$. This is also illustrated by the empirical analysis of Table 5.11.

Allen et al. (1991d) processed a set of 101 examples of steroid sidechains using single-linkage and Jarvis-Patrick algorithms. They considered both *chiral* and *achiral* clustering, where the latter regards enantiomorphs as equivalent. Allen et al. (1995) (henceforth ABW) repeated the achiral experiment, using an augmented set of 151 examples and single-linkage clustering; they reported ten clusters with two or more instances, covering 148 of the 151 examples.

The steroid C17 sidechain results of ABW can be compared with those of IMEM (see Table 5.12). The two methods produce similar results for the six archetypal forms. A comparative analysis between IMEM and ABW results can be given for each archetypal concept:

A ABW assign 112 instances to the A class, IMEM 105. Of the 7 extra instances in ABW, one is `CHOMYS` which, as discussed above, has suspicious coordinates. Instances 23 (`BOGBUU`), 28 (`BUGLEU`), and 92 (`DEDSOU`) are assigned by IMEM to the D2 class, since τ_2 is well out of the ap range at -110.9, -120.2, and -99.9. Instance 116 (`GASFAH`) is assigned to the B concept, since τ_4 is out of the ap range at 122.0. The remaining two instances appear in IMEM concept 12.

B The B clusters are identical, except that IMEM adds instance 116, as noted above.

C The C clusters are identical, except that ABW assign instance 43 to this class. As reported above, the coordinates for instance 43 may be in error.

D1 The D1 clusters are identical in ABW and IMEM.

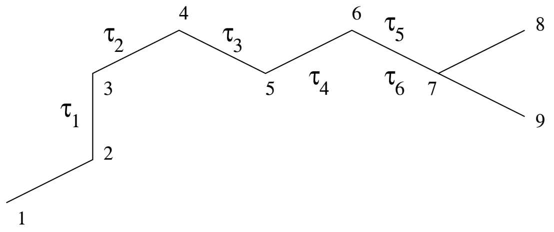
D2 As noted above, IMEM assigns instances 23, 28, and 92 to the D2 class. Otherwise the clusters are identical.

D3 The D3 clusters are identical in ABW and IMEM.

Thus the results of ABW and IMEM are in close agreement.

5.2.4 Hexopyranose sugars

The previous three datasets illustrated the conformational clustering of molecular graphs possessing some atomic permutation symmetry. The hexopyranose structure has 11 parts and is asymmetric, due to the exocyclic carbon on C5 (Figure 5.9). Unlike the previous



1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 9 8

Figure 5.8: The steroid C17 sidechain, with its symmetry group

Name	Standard concept						$\mathcal{SDL}$ concept					
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6
A	180	180	180	180	180	3 to 120	ap	ap	ap	ap	ap	sp or +c
B	180	180	180	40 to 70	180	42 to 85	ap	ap	ap	+c	ap	+c
C	180	180	56 to 85	180	180	-39 to -64	ap	ap	+c	ap	ap	-c
D1	180	60 to 73	180	-67 to -99	180	-39 to -64	ap	+c	ap	-c	ap	-c
D2	180	60, 86	180	180	180	-3, -59	ap	+c	ap	ap	ap	sp or -c
D3	180	57, 65	180	61, 64	180	56, 70	ap	+c	ap	+c	ap	+c

Table 5.10: Archetypal C17 sidechains, from ADT3, Table 1, and (Duax et al. 1980).

Id	Discovered concept						Standard name	Frequency
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6		
1	ap	ap	ap	ap	+c	ap	A	93
2	ap	ap	ap	ap	ap	sp		12
3	ap	ap	ap	-c	-c	ap	B	7
4	ap	ap	-c	ap	+c	ap	C	4
5	ap	+c	ap	-c	-c	ap	D1	3
6	ap	+c	ap	ap	-c	ap	D2	8
7	ap	-c	ap	ap	ap	sp		2
8	ap	+c	ap	+c	+c	ap	D3	4
9	ap	+c	ap	ap	+c	ap	D2-like	5
10	ap	ap	+c	ap	+c	-c	C-like	4
11	ap	ap	+c	ap	ap	+c	C-like	2
12	ap	ap	ap	ap	+c	-c	A-like	3

Table 5.11: Steroid C17 sidechain conformations discovered by IMEM.

	ABW		IMEM	
Class	Id	Np	Id	Np
A	1	112	1 2	105
B	7	6	3	7
C	9	5	4	4
D1	2	3	5	3
D2	6	7	6 7	10
D3	3	4	8	4

Table 5.12: Concept discovery results for the steroid C17 sidechains: ABW vs. IMEM. Id: concept identifier; Np: combined membership of concepts.

sections, this section also considers multi-layer taxonomies. These are important as intermediate concepts can highlight the similarities and differences between molecular structures (Maggiora and Johnson 1990).

In this experiment, the Klyne-Prelog relations (sp, +c, etc.) are not used. Instead, a binary relation of non-bonded contact or *proximity* is used; this relation computes whether or not two atoms are within van der Waals bonding distance of each other. Letting $\mathtt{vrad}(a)$ be the van der Waals radius[6] of an atom of type a, the `proximal` relation is defined as:

$$\begin{aligned}\mathtt{proximal(x,y)} \quad \stackrel{\text{def}}{=} \quad & \mathtt{not\ bonded(x,y)} \wedge \\ & \mathtt{d(second(x),second(y))} \leq \mathtt{vrad(first(x))} + \mathtt{vrad(first(y))} + \beta\end{aligned}$$

that is, two atoms are proximal if they are not bonded, and their interatomic distance is less than the sum of their van der Waals radii plus a constant factor β. A family of proximity relations arises by varying β. Taking the van der Waals radii of carbon and oxygen to be 1.70 and 1.52 Å, the upper bounds of some proximity relations are shown in Table 5.13. A value of $\beta = 0.07$ is used for the experiments reported here so, for example, a carbon and an oxygen are proximal if their distance is between 1.76 and 3.29 Å. Note that the `proximal` relation depends on the types of the atoms given as parameters.

In 3D, the pyranose ring exists in the chair conformation (see Section 5.2.1), and not as a flat ring. The chair form, however, does not uniquely identify the three–dimensional structure of a hexopyranose sugar. Each of the 5 ring carbon atoms is a *stereocenter*: the substituent atoms can have two different and nonsuperposable *configurations*. There are thus exactly 32 (2^5) possible configurational classes, or *stereoisomers* of hexopyranose

[6] Appendix E of (CCDC 1994) gives a detailed table of van der Waals radii.

sugars. Unlike constitutional isomers, stereoisomers have the same molecular graphs, with a rigid 3D part to which the substituents of stereocenters are related. Unlike conformations, different stereoisomers do not easily interconvert, as this requires the breaking and re-formation of bonds. Chemists have devised a precise classification for the stereoisomers of hexopyranose sugars (see Solomons 1988). They are described by the R and S relations. A sugar is classified as α or β depending on the configuration at C1, and a sugar is D or L depending on the configuration at C5. Each stereoisomer has different functional properties. In some sugars the chair form can invert, producing an enantiomorph. Thus, in addition to the basic 32 configurational classes, there are 32 possible conformational variants given by the enantiomeric chair forms.

A dataset of 249 hexopyranose structures, described by Allen and Fortier (1993), was processed using IMEM. A similarity threshold of $\tau = 0.75$ was used to encourage the formation of a multi-layer concept taxonomy, that is, concepts with fewer parts than the training examples. The atomic permutation symmetry group was not used for similarity valuations, as here we are looking for partial R-monomorphisms rather than R-isomorphisms. In addition, IMEM only searched for *connected* images, where every atom has a transitive path to every other atom via the `bonded` relation. The concept taxonomy of Figure 5.10 was created from the 249 examples. When all immediate successors of a concept are individuals, these are not displayed in the taxonomy. The taxonomy clearly shows 14 concepts (at the leaves) present in the dataset. Comparing these concepts with crystallographers' hand labellings of the 249 examples revealed that IMEM exactly reproduced the standard chemical classification of the pyranose sugars. This classification was achieved using the simple and intuitive relation of proximity.[7]

The numbers, in brackets, in each concept of Figure 5.10 are the numbers of parts in the image term. The two concepts `uniq-31` and `uniq-8` both have 8 parts: the central pyran ring and substituents O3 and C6. In `uniq-31` C1 and O3 are proximal, and in `uniq-8` C1 and O3 are not proximal. The atoms C1 and C6 are proximal in the individual `BIKWOH10` and not in `uniq-31` or `uniq-8` (therefore, all other 248 examples), so it is not subsumed by either of these concepts.

After clustering, the 14 discovered concepts were inspected to see which proximity values were invariant. It was found that only 7 interatomic proximity values varied across the

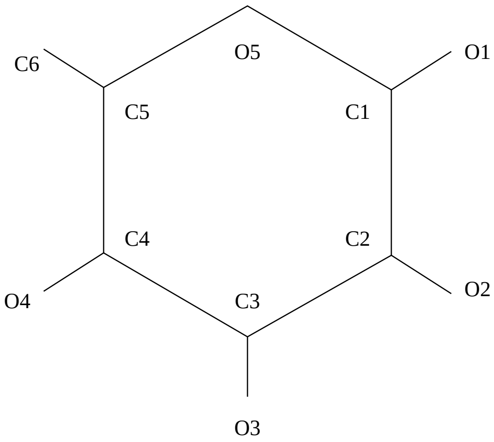
[7] Instance 128 (CSD refcode `FUSICO10`) — an α-D-glucose — was removed from the training set after an initial run because it caused the α-D-glucose concept to be split. Analyzing the reason for this, it was found that instance 128 contains an unusually long C5-C6 bond (1.762 Å), meaning that these two atoms were considered proximal (and not bonded). Its coordinates are probably in error.

discovered concepts. These are the proximity relations between O5-O2, O5-O4, C1-O3, C1-C6, C2-O4, C3-O1, and C4-O2. Among the instances of any particular discovered concept, all other proximity values are equivalent. Furthermore, the O5-O2 and C4-O2 proximities, and the O5-O4 and C2-O4 proximities were found to be functionally equivalent. Thus a total of 5 interatomic proximity values, listed in Table 5.15, might be used to cluster the 249 pyranose sugars. For example, referring to Table 5.15, δ_3 is the attribute for the proximity value between C1 and O3. Table 5.14 uses these attributes to give a compact structural sequence description to the concept. A "l" is an abbreviation for `true`, a "o" for `false`.

The attributes of Table 5.15 can be correlated with the standard D, L, α, β attributes for the pyranose sugars. It can be seen from Table 5.14 that `BIKWOH10`, the only L-form sugar in the dataset, is also the only one where C1 and C6 are proximal (attribute δ_4). This is perfectly reasonable, considering that the configuration at the stereocenter C5 determines the D or L designation for a sugar, and that C6 is the substituent of C5. Thus the attribute δ_4 determines the D or L designation. Earlier it was mentioned that the configuration at the stereocenter C1 determines the α or β designation. It appears that one attribute, δ_5, can distinguish between α and β forms (see, for example, `uniq-1` and `uniq-19`). The three remaining attributes, δ_1, δ_2, and δ_3, might be used to designate the 8 major sugar classes (glucose, galactose, mannose, etc.). The 5 qualitative attributes will give rise to a total of 32 (2^5) concepts.

An analysis was performed to determine the sensitivity of the clustering to the van der Waals proximity parameter β. It was found that values of β in the range of $[0.06, 0.09]$ produced perfect clusterings. Below this range, at $\beta = 0.05$, instance 121 (refcode `FITKAU`) becomes a singleton concept. An inspection of this structure revealed that it has an O2-O3 distance of 3.09 Å, sufficient to indicate proximity (see Table 5.13, fourth column, last row). In all other β-D-glucose instances, O2 and O3 are proximal. At $\beta = 0.0$, the same taxonomy as for $\beta = 0.05$ is produced, with the exception that two extra concepts are formed. One is an α-D-galactose split (2 instances), the other is a β-D-glucose split (3 instances). Above the $[0.06, 0.09]$ range, at $\beta = 0.1$, instance 70 (refcode `CELGIJ`) becomes a singleton concept. In it, O1 and C4 are proximal, with a distance of 3.32 Å (see Table 5.13, last column, first row). The O1 and C4 atoms are not proximal in all other α-D-glucose instances. Thus all values of β in the range $[0.00, 0.10]$ produce reasonable clusterings of the hexopyranose dataset. As β increases past this range, clustering quality slowly but progressively deteriorates.

Allen and Fortier (1993) cluster the data into the same categories as those of Table 5.14. They use as numerical features 6 intra-annular (e.g., O5-C1-C2-C3) and 5 improper (e.g.,



`O5 C1 C2 C3 C4 C5 O1 O2 O3 O4 C6`

Figure 5.9: The hexopyranose, with its symmetry element

O5-C1-O1-C2) torsion angles. Chemical expertise was necessary to determine these quantitative parameters from the many possible independent improper torsion angles. Only the 5 improper torsion angles are necessary to determine configuration, and these might be used as equivalents for the standard R and S descriptors of stereochemistry. To test this hypothesis using IMEM, two relations were defined (see Figure 5.11). These 4-ary relations are similar to the 5-ary R and S relations of stereochemistry, with the only difference being that a substituent hydrogen is assumed present. They are also very similar to the Klyne-Prelog relations, except that they are not defined among chains of atoms. Rather, they are defined between three atoms and a substituent (see Figure 5.11), as in the improper torsion angles used by Allen and Fortier, e.g., C5-C4-O4-C3 in Figure 5.9. These R and S relations also led to perfect partitionings of the hexopyranose dataset. Two experiments were performed: a chiral and an achiral clustering.[8] In the chiral approach, IMEM created 16 versus 14 concepts. The extra two concepts represented enantiomorphs of β-D-glucose (9 instances) and α-D-glucose (1 instance). These enantiomorphs were not discovered using the `proximal` relation, as proximity is invariant under mirror inversion. In the achiral approach, 13 rather than 14 concepts were discovered, with the D and L forms of the α-idose sugars merged into a single concept. This behaviour is chemically correct, as a D-form sugar has the same configuration as the enantiomorph of its L-form counterpart.

[8] Recall that achiral clustering views enantiomorphs as structurally equivalent concepts.

Atom	Atom	$\beta = 0.00$	$\beta = 0.05$	$\beta = 0.07$	$\beta = 0.10$
C	O	3.22	3.27	3.29	3.32
C	C	3.40	3.45	3.47	3.50
O	O	3.04	3.09	3.11	3.14

Table 5.13: Upper-bound distances for non-bonded contact or proximity between different pairs of atoms, under different values of β.

Id	Discovered concept					Standard name	Frequency
	δ_1	δ_2	δ_3	δ_4	δ_5		
`uniq-1`	o	o	o	o	o	β-D-glucose	104
`uniq-19`	o	o	o	o	l	α-D-glucose	73
`uniq-16`	o	l	o	o	o	β-D-galactose	25
`uniq-7`	o	l	o	o	l	α-D-galactose	19
`uniq-10`	l	o	o	o	l	α-D-mannose	8
`uniq-14`	l	l	o	o	l	α-D-talose	3
`uniq-17`	o	o	l	o	l	α-D-allose	2
`uniq-22`	o	l	l	o	o	β-D-gulose	3
`BIKWOH10*`	o	o	o	l	o	α-L-idose	1
`uniq-26`	l	o	o	o	o	β-D-mannose	3
`COKBIN*`	o	o	l	o	o	β-D-allose	1
`uniq-34`	o	l	l	o	l	α-D-gulose	2
`uniq-33`	l	o	l	o	l	α-D-altrose	3
`PAIDOP*`	l	l	l	o	l	α-D-idose	1

Table 5.14: The hexopyranose conformations discovered by IMEM. *: individual concepts.

	O1	O2	O3	O4	C6
O5		δ_1		δ_2	
C1			δ_3		δ_4
C3	δ_5				

Table 5.15: Pertinent proximity values for the hexopyranose dataset

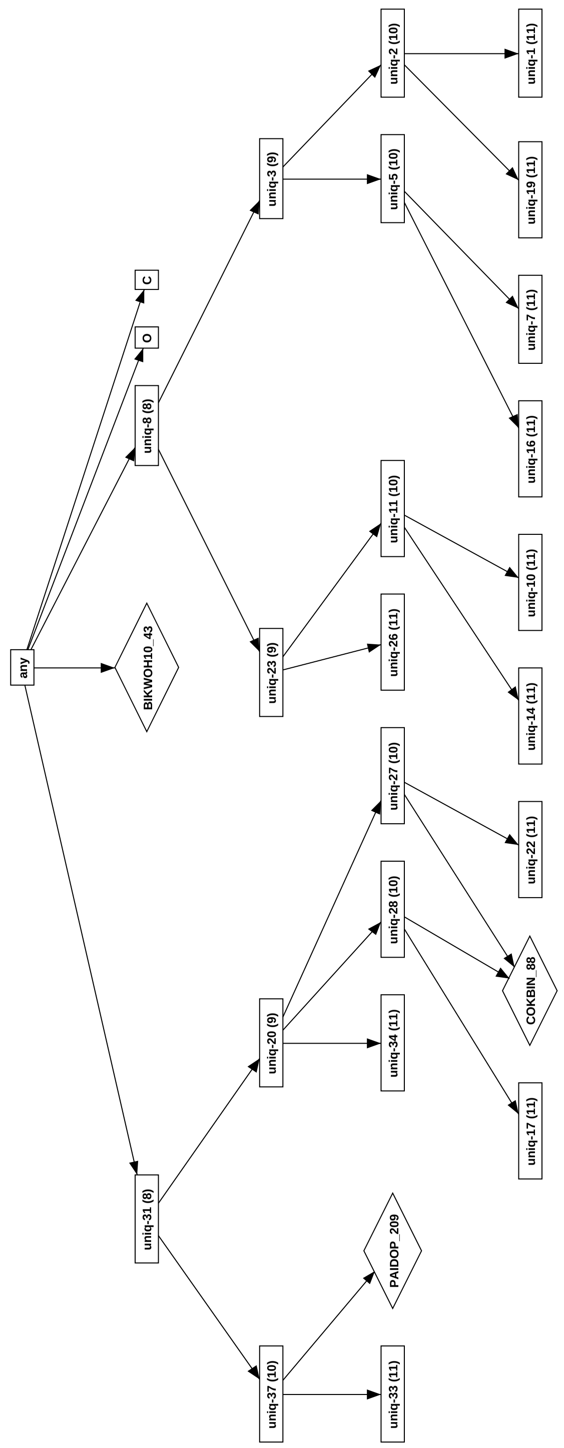


Figure 5.10: The hexopyranose concept hierarchy, after incorporation of 249 instances. Individual concepts are in diamonds.

```
R(p,q,r,s)  def=

      if (bonded(p,q) and bonded(p,r) and bonded(p,s)) then
           t(second(p),second(q),second(r),second(s)) ≥ 0

S(p,q,r,s)  def=

      if (bonded(p,q) and bonded(p,r) and bonded(p,s)) then
           t(second(p),second(q),second(r),second(s)) < 0
```

Figure 5.11: The modified R and S relations of stereochemistry, used to describe configurational aspects of the hexopyranose sugars.

5.3 Discussion and future work

This chapter has applied the IMEM structured concept discovery method to several datasets drawn from the CSD. These were chosen to illustrate different points: the conformational clustering of cyclic structures (datasets 1 and 2), of extended chain structures (dataset 3), and of structures whose 3D shapes are determined by stereochemical effects (dataset 4). In each case, the discovered concepts were compared with published results from numerical clustering techniques and also with accepted chemical classifications. For all datasets, the concepts discovered by IMEM are chemically sensible, and this has been illustrated by a rigorous comparison with archetypal conformation classes. For the hexopyranose dataset, a technique for stereochemical partitioning, not based directly on standard stereochemical descriptors, proved to be effective. All discoveries were made using simple and chemically intuitive qualitative relations such as Klyne-Prelog torsion angle partitioning, van der Waals proximity, and R and S stereochemical relations.

In several cases, IMEM discovered possible errors in the reported conformational data, expressed by outlying individual concepts (CSD refcodes `BABPIP`, `BEWNOG`, `SILYIV`, `CHOCAL`, `CHOMYS`, `CHOBRH10`, `CHOLEU01`, and `FUSIC010`). These sort of deviant singletons are revealed because IMEM, for all datasets, simultaneously required connectivity to be preserved. This would be difficult in the scheme of Allen et al. because their distance metrics cannot easily cope with different quantitative attributes measured in different units (i.e., degrees and Angstroms).

The IMEM approach has discovered concepts which correspond to actual low-energy conformers. Similar to the numerical clustering approach of Allen et al., this was done with no *a priori* knowledge of molecular dynamics or potential energy functions. The success of both approaches reflects and depends upon an ability to adequately represent and relate conformation classes. The numerical approach uses quantitative numerical features and distance metrics, whereas IMEM uses qualitative relations and logical subsumption. Both techniques will have difficulties if conformations are not adequately represented in a training set, or if there there are no apparent populated peaks in conformation space for the molecule in question. However, in the latter case, IMEM may still be able to identify similar 3D substructures in examples. In addition, both techniques will often be able to express 3D arrangements which do not correspond to true conformation classes. Thus an interesting idea for future work is to use least-square fitting and energy minimization techniques to further validate $\mathcal{SDL}$ conformations; removing spurious concepts which may be visually similar to others, or which may represent rare conformers or instances along an interconversion pathway between conformations. Also, $\mathcal{SDL}$ concepts could be used as starting points for energy minimization work, perhaps removing the need for searches over the complete conformational space.

The IMEM method has addressed the central problem with the conformational clustering scheme of Allen et al. in that it removes the need for substantial user intervention during the discovery process. In this chapter, IMEM was not guided by *a priori* knowledge of the number and form of classes to expect in a dataset. Also, it appears that less *a posteriori* processing of the clustering results is necessary to extract meaningful classes. The numerical clustering method creates many singletons and doublet clusters, and these are excluded from subsequent analyses using expert intuition and not by an autonomous process.

The IMEM method does not, however, completely remove the need for user intervention in the clustering process. Intervention is required in order to determine the hypothesis space for discovery, that is, to determine the relations used for expressing $\mathcal{SDL}$ concepts. However, this chapter has provided empirical evidence for the claim that a small library of parameterized relations could suffice to describe conformations of other cyclic and extended chain forms.

The clustering method of Allen et al. is best suited to the task of conformational clustering, where all examples have the same constitution and are described by the same number of quantitative features. Although not illustrated in this chapter, IMEM is in theory capable of more general discovery tasks, where recurrent patterns among disparate molecular graphs can be sought. This is part of the *pharmacophore discovery* task: identifying the common structural features in drugs of known and related activity (Golender and Rozenblit 1983). Provided that pharmacophores of interest could be represented adequately by $\mathcal{SDL}$

concepts, the IMEM structured concept discovery method could be applied.

Under this general paradigm of recurrent structure identification falls the work of Wilcox and Levinson (1986) in discovering taxonomies of 2D molecular graphs and using these to index a database for efficient substructure searching. Their work is important in three respects. First, the similarity between 2D graphs is not based on the number of common features or *screens*, as in the CSD similarity retrieval method, but is instead truly topological in nature. Such methods for 2D similarity comparison based on maximal common substructures are gaining increasing interest (Willett 1991). Second, the screening search in the CSD is simply a restricted form of a topological most general subsumee (MGS) search, where the concept taxonomy is restricted to a single layer. Third, screens are produced automatically by a learning procedure. The $\mathcal{SDL}$ scheme extends this work by providing a unified 2D/3D knowledge representation, and by expressing qualitative and transformation-invariant 3D relations. Allen et al. (1993b) indicate that results from conformational clustering can be embodied as 3D screens for efficient 3D similarity retrieval. Thus an interesting idea for future research is to explore the use of the $\mathcal{SDL}$/IMEM scheme for this purpose.

In the opening to Part II of this thesis it was stated that discovered concepts could be used as templates or conformational units for structure generation and small molecule model building. However, in this chapter no attempt has been made to assess the quality of discovered concepts for such purposes. In the next chapter, this assessment is made for protein motifs, where an interatomic distance root-mean-square metric is used to evaluate intracluster similarity. In the present chapter concepts were evaluated by comparing them with known conformations and with the results of numerical clustering experiments.

Finally, although it cannot yet be claimed that IMEM applies equally well to all conformational clustering tasks, this chapter has successfully considered a variety of molecular graphs illustrating different aspects of molecular conformation. There are certainly many other graphs for which knowledge of conformational preferences remain buried within the entries of the Cambridge Structural Database, for example, higher-order and fused ring systems, hydrogen-bonded motifs, amino acid rotamers, zeolite structures, and so on. Molecular database mining tools such as IMEM can surely help to uncover these concepts.

Addendum

A system called IMEM/CSD has been installed at the Cambridge Crystallographic Data Centre, and will reproduce the results presented in this chapter.

Chapter 6

Knowledge Discovery in the Protein Data Bank

The investigation of relations between protein tertiary structure and amino acid sequence is a topic of tremendous importance in molecular biology. The automated discovery of recurrent patterns of structure and sequence is an essential part of this investigation. These patterns, known as *protein motifs*, are abstractions of fragments drawn from proteins of known sequence and tertiary structure. This chapter has two objectives. The first is to introduce and define protein motifs, and provide a survey of previous research on protein motif discovery. The second is to present and apply a novel approach to protein motif representation and discovery, which is based directly on the theory developed in Part I of this thesis. A large database of protein fragments is processed using this approach, and several interesting and significant protein motifs are discovered.

6.1 Introduction and definitions

The Protein Data Bank (Bernstein et al. 1977) is the main source of data regarding the 3D structure of proteins. Each PDB entry contains 3D atomic coordinate data for a protein structure. At the middle of 1994, the PDB contained roughly 1000 entries, and it is continually expanding. A task of growing importance is to mine the PDB for recurrent sequence and structure motifs. These generalized patterns can facilitate efficient information retrieval and data incorporation, providing a conceptual framework to which new structural data can be related. They can also be used for prediction of protein conformation from constitution, since they often represent preserved 3D structural features. It is necessary to understand protein 3D structure before protein function and activity can be understood at the molecular level.

Proteins are macromolecules comprising chains of structural building blocks known as *amino acids*. Amino acids have a *backbone* and a *side chain*, and are classified into 20 groups based on the topological structure of their side chain. Each amino acid can be denoted by a single letter name. Proteins can be described at various levels of abstraction. The *primary structure* of a protein is given by its linear (1D) sequence of amino acids. The primary structure dictates the 3D structure of the protein in solution, although the rules governing this determination have not yet been discovered. The *secondary structure* of a protein is given by a sequence of structural identifiers, such as `H` (alpha–helix), `E` (strand), or `T` (turn). Equivalently, a sequence of n elements is often described by n such structural identifiers: the format produced by the DSSP secondary structure assignment program (Kabsch and Sander 1983). Protein *tertiary structure* or *conformation* is described by the positions of all atoms of the macromolecule in 3D space (see Figure 6.1).

In addition to the levels of secondary and tertiary structure, there are a variety of abstract representation forms for protein 3D structure (Schulz and Schirmer 1979). One such representation, encountered frequently throughout this chapter, is the *protein virtual backbone*. For a chain of length n, this comprises n representative points, one for each residue in the chain. The chosen representative point is often the position of the Cα atom (one of the amino acid backbone atoms) of the residue. In a protein virtual backbone, two points are *contiguous* or *virtually bonded* if their residues are adjacent in the amino acid sequence. The *virtual bond angle* (VBA) is the bond angle among three contiguous points. The *virtual bond dihedral angle* (VBDA) is the torsion angle among four contiguous points.

A *protein fragment* is an observed pattern of amino acid residues, for example, a region of (1D) primary structure, or of (3D) tertiary structure. A *protein motif* is an abstraction of one or more fragments. Protein motifs can be classified into four categories. *Sequence* motifs are linear strings of residue identifiers with an implicit topological ordering. *Sequence–structure* motifs are sequence motifs with predefined secondary structure identifiers attached to one or more residues in the motif. The sequence is assumed to be predictive of the associated structure. *Structure* motifs are 3D structural objects, described by positions of residue objects in 3D Euclidean space. Structure motifs are free of sequence information, although most research enforces contiguity of structure motif components. Finally, *structure–sequence* motifs are combined 1D-3D structures that associate sequence information with a structure motif. Structure–sequence motifs need not indicate a fixed direction of implication between structure and sequence. Figure 6.2 illustrates these four types of protein motifs, along with some subclassifications which are elaborated upon later

```
TTCCPSIVARSNFNVCRLPGTPEAICATYTGCIIIPGATCPGDYAN
-EE-SSHHHHHHHHHHHTTT--HHHHHHHHS-EE-SSS---GGG--
```

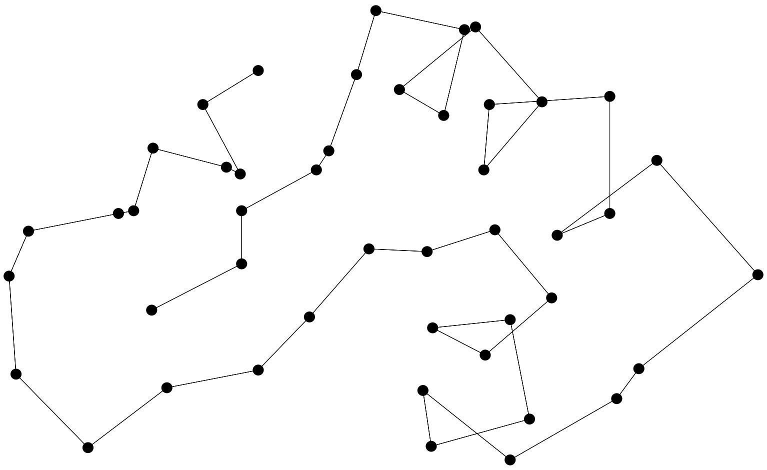


Figure 6.1: Representations of structure (primary, secondary, tertiary) for a small protein (Crambin: PDB code 1CRN). Top: primary structure given by a sequence of 46 amino acid identifiers. The primary sequence is followed by a sequence of 46 secondary structure identifiers (DSSP format: a "-" indicates unassigned structure for the corresponding residue). Bottom: a 2D visual projection of the protein's tertiary structure (virtual backbone representation: each dot is the location of a Cα atom).

in the chapter. The first three motif types are discussed by Thornton and Gardner (1989); the term "structure–sequence motif" is my own, and will be presented in this chapter.

Machine discovery of protein motifs of various types is currently an area of intense interest in molecular biology. One of the objectives of this chapter is to bring this important application domain to the attention of the machine learning community. Section 6.2 of this chapter will survey previous research on protein motif discovery according to the above categorization of motifs. The second aim of this chapter is to present and apply a new approach to protein motif discovery, which is based on the $\mathcal{SDL}$ knowledge representation ideas (Chapter 3) and the IMEM structured concept formation method (Chapter 4). The $\mathcal{SDL}$ description logic facilitates reasoning about subsumption of motifs — an ability not found in other protein motif discovery systems — while the concept formation method creates a concept taxonomy of generalized motifs. Section 6.3 presents some promising results obtained on two large datasets of protein fragments.

6.2 Discovery of protein motifs

The field of empirical machine discovery encompasses the theories and autonomous processes involved in finding novel regularities, concepts, or dependencies in data. It is convenient to

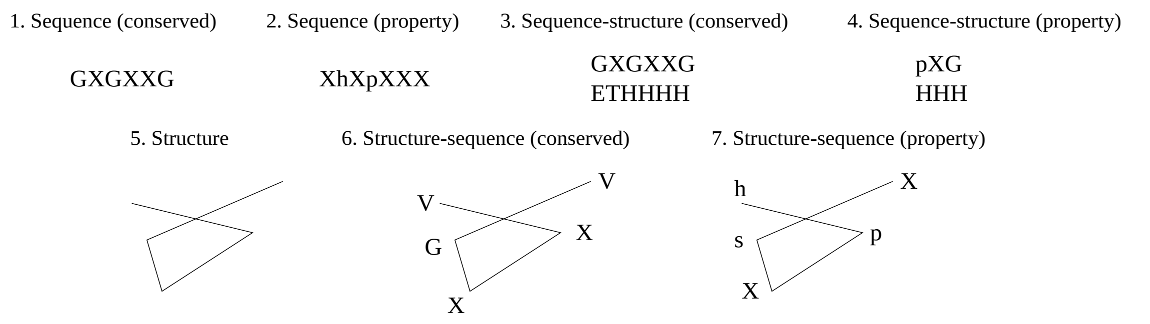


Figure 6.2: Various types of protein motifs. Legend; X: any residue, G: glycine, V: valine, H: helix, E: β-strand, T: turn, p:polar, s: small, h: hydrophobic.

identify a protein motif with a concept, having a formal intensional description in addition to an extensional meaning. In order to apply concept discovery techniques, a rigorous mathematical semantics for a motif is necessary to determine whether an observation lies within that concept's extension.

There has been a considerable amount of research on machine discovery of protein motifs. Most, but not all, of this work relies on some form of numerical clustering: fragments are described by a set of numeric features, a distance metric on these descriptions is defined, and motifs emerge as cluster centroids during the clustering process. The technique presented in this chapter uses, by contrast, a conceptual clustering technique; fragments are structured objects, a measure of relational or structural similarity is defined, and motifs are most specific generalizations of fragments.

In this section I will describe and compare a handful of methods in terms of their representation theory, the type of motif under consideration, the semantics given to motifs, the motif evaluation mechanism employed, and the pragmatics of discovered motifs. Though the results presented in Section 6.3 will be concerned mainly with structure–related motifs, it is informative to review sequence motif work as a backdrop for discussing the more general structure–sequence motif.

6.2.1 Sequence and sequence–structure motifs

Protein sequence motifs are the most commonly encountered motif type in the molecular biology literature. It is generally assumed that similarities in protein sequence are indicative of structural and functional similarity. Thus the discovery of sequence motifs from structurally similar proteins or protein fragments is an important method for uncovering relationships between protein structure and sequence. Protein sequence motifs can facilitate the incremental acquisition and indexing of sequence data into knowledge bases organized according to sequence similarity (Taylor 1986). Large libraries of sequence motifs have been developed (Pongor et al. 1993; Bucher and Bairoch 1994).

Sequence motifs can be discovered from a maximal alignment of one or more protein sequences, followed by the abstraction of residues at aligned positions. There is an extensive literature on the comparison of sequence motifs: see Lathrop et al. (1993) for a good survey of this work. Conserved residues are those identical at corresponding alignment positions. It is uncommon to find long contiguous sequences of conserved residues in non–homologous proteins (Sternberg and Islam 1990), hence the need arises to construct histograms of residue distribution at alignment positions to produce a consensus sequence motif. Taylor (1986) uses a more general abstraction scheme; a domain theory is used to classify amino acids into nondisjoint groups based on physicochemical properties such as hydrophobicity and polarity which are expected to have an influence on protein folding. Following Rooman and Wodak (1991), I shall refer to sequence motifs containing property identifiers as *property* motifs (see Figure 6.2).

The PIMA method of Smith and Smith (1990) (also see Lathrop et al. 1993) discovers property sequence motifs which cover an entire set of sequences from functionally related proteins. Smith and Smith use a physicochemical classification of amino acids somewhat different from Taylor's (1986). Discovered patterns can contain special "gap" identifiers, which can function as 0 or 1 residues. PIMA is a supervised learning method, since the class of related sequences (e.g., from the protein kinase and serine protease family) is delimited prior to the discovery of a covering sequence motif.

Much of the work on protein secondary structure prediction is based on the *a priori* definition of sequence motifs that are predictive of a certain type of secondary structure identifier. These sequence–structure motifs (referred to by Thornton and Gardner (1989) as "structure-related sequence motifs") have an inherent directionality of implication from sequence to structure. The work of Rooman, Wodak, and colleagues has been influential in establishing important sequence–structure predictivity results. Rooman and Wodak (1988) associate with each amino acid in a motif a standard secondary structure identifier (e.g., Figure 6.2, motif 3). A discovery procedure develops short sequence regular expressions with structure associations. It is demonstrated that, while not enough associations are derived to predict a complete protein structure, a number of reliable and predictive motifs do exist. In a similar study, Rooman et al. (1990b) replace the standard secondary structure identifiers with elements produced by structure motif discovery (Section 6.2.2).

Rooman and Wodak (1991) present a sequence–structure study which discovers more general property sequences in addition to conserved sequences. The predictive power of derived property sequence–structure motifs is high ($\sim$ 80%), however overall secondary structure prediction scores remain low due to the inapplicability of specific motifs at many positions of the amino acid sequence. A property sequence–structure motif only contains properties, and cannot contain specific residue identifiers. The $\mathcal{SDL}$ representation language can express motifs with combined sequence types.

6.2.2 Structure motifs

The accurate prediction of protein tertiary structure from amino acid sequence, while theoretically possible, has remained one of the great open problems in molecular biology. Though the prediction problem is typically reduced to a more manageable one — the prediction of secondary structure from sequence followed by the packing of secondary structures into 3D — recently there has been increasing interest in the automated discovery and use of structural elements less coarse than the standard secondary structures. There are three main reasons for this development. First, there is a wide discrepancy between different methods for secondary structure assignment from tertiary structure (Zhang et al. 1993). A prediction system relying on one assignment method for training and evaluation is modelling to some extent its particular characteristics (Colloc'h et al. 1993). Second, unidentifiable folding patterns are usually classified as "random coil," even though these regions are neither random nor undefinable (Prestrelski et al. 1992). Finally, the packing of secondary structure elements is itself a non–trivial task and is dependent on accurate secondary structure predictions. By contrast, structure motifs are building blocks that can be used to precisely describe the tertiary structure of a new protein (Jones and Thirup 1986; Claessens et al. 1989; Richards and Kundrot 1988; Unger et al. 1989; Finzel et al. 1990; Levitt 1992).

Structure motifs are typically discovered by numerical clustering. These procedures require a fixed set of numeric parameters to describe observations, and a distance metric over these multidimensional vectors. In all structure motif discovery research surveyed in this section, the initial numeric parameters are simply the list of 3D coordinates of fragment components. The "visual" similarity of two fragments is thus highly dependent on the coordinate frame of the protein(s) from which the fragments were extracted. This problem can be circumvented in three ways: 1) by standardizing fragment descriptions before comparison; 2) by using numeric features invariant with respect to Euclidean transformations (rotations and translations), or 3) by first performing an optimal alignment of the fragments in 3D space before their comparison. There are drawbacks with each approach; the first implicitly forces a fragment alignment that is possibly suboptimal, the second is sensitive to the chosen features, and the third complicates the semantics of a motif. Further weaknesses with these approaches are discussed at the end of this section.

Hunter and States (1991) standardize fragments by placing their center of mass at the origin, orienting the x, y, and z axes relative to the principal axes of the moment of inertia tensor of the fragment. The authors note various problems with this standardization method; foremost is the fact that small changes in fragment data can flip the choice of an axis, confounding the motif-fragment relationship. Bayesian classification techniques are used to produce clusterings which are evaluated according to Bayes' formula with a prior distribution favouring fewer classes; given two clusterings, each with the same number of classes, the method will prefer the clustering which has a tighter fit to the data. Motifs are represented by a probability distribution over Cartesian coordinates for the backbone atoms of each residue. Thus motifs are probabilistic concepts; fragments fall into a class with a certain probability. In contrast to other approaches reviewed below, amino acids are represented by *line* rather than *point* data, since all backbone atoms are used.

Both Prestrelski et al. (1992) and Zhang et al. (1993) use the invariant feature approach to structure fragment comparison. In (Prestrelski et al. 1992), fixed-length (n) virtual backbone fragments are represented by a sequence of $(n-4+1)$ VBDAs and $(n-4)$ so–called "linear distance" (LD) parameters. The latter measures the sum of distances from a fragment component to the next four components in the fragment. It can easily be shown that these features are invariant under Euclidean transformations. A distance metric weights the relative contribution of the VBDA and the LD components of the feature vector. A simple clustering procedure, applied to 2378 fragments of length $n = 8$, revealed 113 equivalence classes. Motifs representing these classes are simply the average values of each of the nine features. The motif library is evaluated according to its ability to cover various known secondary structural phenomena such as helical or extended strand structures.

Zhang et al. (1993) use a different set of invariant features. A contiguous fragment of length n, drawn from a protein virtual backbone, is described by $(n-3)$ VBAs, $(n-4+1)$ VBDAs, and roughly $(n^2/2)$ intra–fragment distances between noncontiguous fragment components. A set of fragments is divided into two training sets. Fragments from one set are presented to an auto–associative neural network. After training, the fragments are once again presented to the network, with the value of each of the 8 hidden units extracted as feature vectors of lower dimensionality. These modified features vectors are subsequently clustered into a fixed number of classes using a numerical clustering procedure. An interesting scheme is employed to evaluate a given clustering; the second training set is also clustered, and the correlation between the two independent clusterings is examined. A higher average correlation, for example, indicates a better ability to reproduce a given categorization on an independent training set.

Rooman et al. (1990a) also use an invariant representation scheme. The similarity between two fragments is computed by a Cα distance root-mean-square (DRMS) metric[1] (Rooman et al. 1990a). Similar to the work of Prestrelski et al. (1992) and Zhang et al. (1993), fixed-length contiguous fragments from a protein virtual backbone were used. A fragment is an instance of a class by virtue of being within a DRMS distance threshold from the prototypical motif of that class. Similar to the work of Hunter and States, a structure motif is a 3D coordinate description, and can be depicted and perhaps evaluated by a chemist using molecular visualization software. A fragment clustering yields a dendogram of cluster merges; this structure must be pruned to produce a library of motifs. Rooman et al. (1990a) discuss different pruning techniques, and point out the dependence of any technique on the pragmatics of the motif library. In a first experiment, a fixed number (four) of general motifs is sought after, for each fragment length from four through seven. These motifs are correlated with four standard secondary structures. The four discovered classes are used in the sequence–structure analysis (Rooman et al. 1990b) discussed in the previous section. In a second experiment, Rooman et al. (1990a) cluster heptamers and only retain motifs which contain, on average, 50 members. Some of these motifs are subjected to a structure–sequence analysis (Section 6.2.3).

Unger et al. (1989) first perform an optimal alignment of two fragments in 3D before their comparison, using a best molecular fit routine (e.g., Kabsch 1976). Virtual backbone, described by Cα positions of residues, are clustered using a k-nearest neighbor algorithm. Whereas Rooman et al. (1990a) use the DRMS metric, based on intra–fragment distances, Unger et al. first optimally align the structures and use the "aligned" root-mean-square

---

[1]

$$\mathrm{DRMS}(X,Y) = \sqrt{\frac{1}{n(n-1)/2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\mathtt{d}(X_i,X_j)-\mathtt{d}(Y_i,Y_j))^2}$$

where X and Y are two protein fragments of the same length n, indexed by i and j, respectively. Recall that $\mathtt{d}(x,y)$ is the Euclidean distance between points x and y.

deviation (ARMS) between the two aligned point sets[2] (Cohen and Sternberg 1980) as a measure of similarity. Various criteria are used to evaluate the proposed clustering, including the production of a reasonable number of discovered motifs, and a reasonable goodness of fit to the training fragments. Matsuo and Kanehisa (1993) also optimally align fragments prior to clustering them.

An important aspect of protein structure is its hierarchical organization (Schulz and Schirmer 1979). Proteins can be described by structural domains, comprising secondary structures, which in turn are composed of amino acids, and so on. Onizuka et al. (1993) introduce a formal representation scheme capable of describing contiguous fragments of variable length in terms of shorter fragments. The invariant numeric parameters, for fragments of various lengths, are clustered into 16 groups each. The main contribution of Onizuka et al. is in the use of a hierarchical representation scheme, a topic that will be revisited later in this chapter.

Nussinov and Wolfson (1991) use large–scale structural comparisons to discover protein structure motifs. Unlike the approaches surveyed above, a protein is not divided into contiguous training fragments; rather, two proteins are compared in their entirety to identify atoms that are superposable in 3D. A set of such points forms a structural motif. A technique from computer vision known as *geometric hashing* is used to compute the geometric alignment parameters which produce the most superposable atoms. Two interesting features of this method are that motifs need not refer to contiguous sequential regions, and that they can contain arbitrarily many components.

Table 6.1 summarizes several structure motif discovery results. The first column indicates the method for computing similarity between two fragments; whether they are first standardized, aligned during comparison, or whether features invariant under Euclidean transformations are used. The second column indicates the training fragment size; there is some consensus on a length from six to eight. The third column indicates the number of training fragments used. Finally, the last column indicates the number of retained motifs. It

---

2

$$\mathrm{ARMS}(X,Y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \mathtt{d}(X_i, Y_i)^2}$$

where X and Y have been aligned to minimize this expression. Cohen and Sternberg (1980) relate DRMS and ARMS for protein fragments by the approximation

$$\mathrm{DRMS}(X,Y) \approx 0.75 \times \mathrm{ARMS}(X,Y) + 0.199$$

Paper	similarity	fragment length	# fragments	# motifs
Zhang et al. 1993	invariant	7	6422+6242	6
Prestrelski et al. 1992	invariant	8	2378	113
Hunter and States 1991	standardized	5/6	9556/9491	27/—
Rooman et al. 1990a	invariant	4–7	13000	4
Rooman et al. 1990a	invariant	7	13000	> 10
Onizuka et al. 1993	invariant	5–129	—	16
Nussinov and Wolfson 1991	alignment	full protein	2	1†
Unger et al. 1989	alignment	6	13000	103
Matsuo and Kanehisa 1993	alignment	7	15320	37

Table 6.1: Structure motif discovery results (—: not reported, †: per protein pair).

is apparent that there is little agreement on the optimal motif library size. As we have seen, however, different research has different pragmatic goals for the discovered motif libraries.

6.2.3 Structure–sequence motifs

Structure–sequence motifs assign both sequence and 3D coordinate information to residues. This motif type is different from the sequence–structure motif in that the motif itself must have an explicit 3D structure. The sequence–structure motifs described in the previous section do not fall into the category of structure–sequence motifs because the 3D structure is only implicit in the association of a residue with a structure identifier.

Structure–sequence motifs are currently receiving a great deal of attention in the molecular biology literature. The "inverse" structure prediction problem, where a sequence is predicted for a given structure, relies on a compact library of structure–sequence motifs. The sequence portion of these motifs can be a conserved or property sequence, and may be probabilistic in that propensities for different amino acids at each sequence position are indicated. A motif, which may be noncontiguous, is "threaded" by the amino acid sequence of unknown structure, with an evaluation function judging the goodness of fit of the input sequence to the motif sequence for a particular threading (Jones et al. 1992; Ponder and Richards 1987). One attraction of this approach is that the number of possible protein structure classes may not be very large — on the order of one thousand (Chothia 1992) — and that an exhaustive search over a compact library of structure–sequence motifs may be feasible.

The motifs used by threading research tend to be large–scale, encompassing an entire protein or core portion thereof. In the technique of Blundell et al. (1987), a common structural core for a set of proteins from a homologous family is first constructed. The sequences corresponding to this core for each training protein are then aligned. Conserved residues are retained and assigned to a common conserved sequence motif. Similar to the work of Nussinov and Wolfson (1991), the technique does not require a training set of fixed-length fragments. Blundell's group is concerned with *knowledge based protein modelling*: protein structure prediction, where the main source of information comes from exploration and abstraction of known structures. In this spirit, Sali and Blundell (1990) develop an elaborate scheme for the comparison of protein structures. The results of a comparison form a "generalized protein," which can be used in predicting the 3D conformation of an amino acid sequence. Similar to the work of Lathrop et al. (1987), proteins are described by a hierarchy, with each level being a sequence of typed elements. Elements of fragments are represented by a host of computed properties, rather than by a single identifier. Attributes of fragment elements can refer to other elements in the sequence, thus representing binary relationships such as hydrogen bonding between elements. Each of these properties and relationships is weighted, and contributes to an overall weighted distance metric.

Many researchers concerned with the discovery of structure motifs (see previous section) have attempted to generalize their techniques to produce structure–sequence motifs. This involves an analysis of the sequences within a discovered structure class. Unger et al. (1989) tabulate statistics on the frequency of each amino acid types at every position in a structure motif, producing a consensus sequence motif. Preliminary results indicate that the local 3D structure of a fragment can sometimes be predicted by assignment of the fragment to a motif based on these frequency tables. Zhang et al. (1993) also tabulate the relative frequency of each of the amino acids for the central residue in their six discovered structure motifs. Finally, Rooman et al. (1990b) obtain a property sequence motif for discovered heptamer structures. These sequences, however, are only weakly specific and are not likely to be predictive of the associated structure.

Each of these three simple extensions to structure motif discovery are restrictive in that a *single* property or conserved sequence must be associated with a structure; it is not possible to represent a structure which is associated with a disjunction of sequences. This is a problem because the consensus sequence of a discovered structure motif will tend towards the flat distribution. The discovery results presented in Section 6.3 of this chapter remove this restriction. Also, most structure–sequence work reflects a pragmatic bias to the prediction of structure from known sequence. As we have seen, an equally useful task is the prediction of sequence from hypothetical or known structure.

Paper	Motif type	Semantic Theory	
		Sequence	Structure
Taylor 1986	sequence	model	—
Smith and Smith 1990	sequence	similarity	—
Rooman and Wodak 1988	seq-struct	model	—
Rooman and Wodak 1991	seq-struct	model	—
Rooman et al. 1990b	seq-struct	model	—
Hunter and States 1991	struct	—	probability
Rooman et al. 1990a	struct	—	similarity
Nussinov and Wolfson 1991	struct	—	similarity
Prestrelski et al. 1992	struct	—	similarity
Onizuka et al. 1993	struct	—	similarity
Matsuo and Kanehisa 1993	struct	—	similarity
Unger et al. 1989	struct-seq	probability	similarity
Blundell et al. 1987	struct-seq	model	similarity
Sali and Blundell 1990	struct-seq	similarity	similarity
Zhang et al. 1993	struct-seq	probability	similarity
Conklin et al. 1993b	struct-seq	model	model

Table 6.2: A comparison of different protein motif representations (—: not applicable to this type of motif).

6.2.4 Discussion

Table 6.2 classifies the protein motif discovery work discussed above according to three dimensions. Column 2 indicates the motif type. Columns 3 and 4 indicate the semantic theory which dictates the meaning of a motif, and hence the motif-fragment relationship. In a *model-theoretic* semantics, the extension of a motif is a set — the set of all fragments with the same properties and relationships as the motif. In a *probabilistic* framework, a motif also denotes a set, but here the elements of the set have a probability of occurrence. A *similarity* semantics can be given in two ways, one where the motif is assigned a distance threshold δ (as in the work of Rooman et al. 1990a), another where there is no bound and the motif denotes something similar to a "fuzzy" set. Researchers in protein motif discovery are often not clear about which semantics is intended. Table 6.3 summarizes the three main semantic theories of protein motifs. For example (row 1), if a motif X has a similarity (δ) semantics, then a fragment Y must meet the condition $d(X,Y) \leq \delta$, where δ is a researcher-specified constant.

Table 6.2 shows that, in the work surveyed, protein motifs have not been given a common semantics. Structure motifs have usually been represented using prototypes that have a similarity semantics, whereas sequence motifs have usually been represented by logical definitions with a model-theoretic semantics. Structure–sequence motifs often inherit a

Semantic theory	# truth values	X	# values for truth	notation
similarity (δ)	1	prototype	2	$d(X,Y) \leq \delta$
similarity	many	prototype	—	$d(X,Y)$
probabilistic	1	probability distribution	many	$p(Y\|X)$
model	1	logical sentence	2	$\models Y \Rightarrow X$

Table 6.3: The relationship between a motif X and a fragment Y in various semantic theories.

confused dual semantics.

Most protein motif discovery work uses a restricted semantics for structure motifs, in that a motif can only subsume a fragment of the same length. This leaves little flexibility for reasoning about relative generality of motifs, an important capability (Rooman and Wodak 1991). Unger et al. (1989) note that the ARMS (and DRMS) measures of structural similarity can give counterintuitive results, not necessarily reflecting topological or qualitative similarities. The $\mathcal{SDL}$ language uses a model-theoretic semantics for both structure and sequence motifs and addresses many of the issues raised in this survey.

6.3 Discovery in the Spatial Description Logic

The first part of this chapter presented the important application of protein motif discovery, and surveyed a number of existing approaches. Many issues were raised and discussed, including the semantics given to motifs, the motif evaluation mechanism employed, and the pragmatics of discovered motifs. This part of the chapter will present and evaluate a new approach to protein motif representation and discovery which is based on the $\mathcal{SDL}$ knowledge representation and the IMEM discovery theory presented in Part I of this thesis.

6.3.1 Protein motif representation

In Chapter 5, the $\mathcal{SDL}$ representation language was used to describe small molecular structures and conformation classes. Components of $\mathcal{SDL}$ images were atom identifiers paired with Cartesian coordinates. To represent protein motifs a slightly different approach has been used. This has been outlined briefly in (Conklin et al. 1993b) and is presented here in full detail.

The main difference in the two approaches concerns the coordinate space into which parts are placed in $\mathcal{SDL}$ images. Protein fragments, like small molecules, must preserve

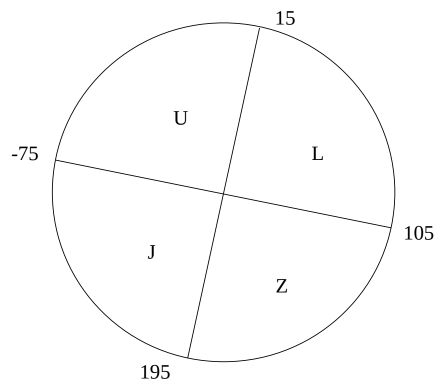


Figure 6.3: The Δ relations, which partition torsion angle space to form four qualitative relational attributes.

connectivity relationships. In the CSD datasets, these were inferred using a test on Euclidean distance ranges. For protein structure motifs, a similar technique would require a reliable range on Cα distances that admitted all pairs of virtually bonded residues and discarded all others. There is an easier solution in $\mathcal{SDL}$. In place of a 3D coordinate space, a 4D coordinate space is used. The first component of this space takes on integer values, and is used to represent topological order and contiguity along the amino acid sequence. The definition of contiguity in this coordinate space is

$$\texttt{bonded}(\texttt{x},\texttt{y}) \quad \stackrel{\text{def}}{=} \quad \texttt{first}(\texttt{second}(\texttt{x})) - \texttt{first}(\texttt{second}(\texttt{y})) = 1$$

that is, two residues are (virtually) bonded if their topological distance along the first dimension is 1. Note, however, the asymmetry of this relation: this is quite desirable as protein sequences have an implicit direction which must also be preserved.

In addition to the (2D) contiguity relation, a scheme similar to the Klyne-Prelog one used in Chapter 5 can be used for conformational descriptions of protein fragments. The Δ relations, so named to distinguish them from the Klyne-Prelog relations, use a different partitioning of torsion angle space. The functional relation Δ takes on four discrete values: U, L, Z, and J (see Figure 6.3 and Figure 6.4). The idea of partitioning the space of VBDAs to create structure descriptors was proposed independently by both Conklin et al. (1993b) and Ring et al. (1992). The particular partitioning of Δ is due to Ring et al., and is based on a statistical analysis of a large number of protein tetramers. It is attractive for protein motifs since repetitions (called *structural sequences*) of L and J identifiers correspond closely to the standard α and β secondary structures, respectively (Levitt and Greer 1977).

```
U(p,q,r,s) ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
      t(rest(second(p)),rest(second(q)),rest(second(r)),rest(second(s))) ≥ −75 or
      t(rest(second(p)),rest(second(q)),rest(second(r)),rest(second(s))) < 15

L(p,q,r,s) ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
      15 ≤ t(rest(second(p)),rest(second(q)),rest(second(r)),rest(second(s))) < 105

Z(p,q,r,s) ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
      105 ≤ t(rest(second(p)),rest(second(q)),rest(second(r)),rest(second(s))) < 195

J(p,q,r,s) ≝

      if (bonded(p,q) and bonded(q,r) and bonded(r,s)) then
      195 ≤ t(rest(second(p)),rest(second(q)),rest(second(r)),rest(second(s))) < 285
```

Figure 6.4: Definitions for the Δ relations. `t` is the torsion angle function. The `rest` function drops the first element from a list, and is used to access Cartesian coordinates.

Another difference between the CSD and PDB experiments concerns the much more extensive use of background knowledge in the PDB experiments. This background knowledge is used to express the components of structure-sequence motifs in $\mathcal{SDL}$. As background knowledge, I have used the physico-chemical amino acid properties of Taylor (1986). The amino acid classes of hydrogen bond acceptor and donor, used by King and Sternberg (1990), were also used. The $\mathcal{SDL}$ program for these properties is given in Table 6.4. Appendix D gives one-letter abbreviations for all properties and amino acids.

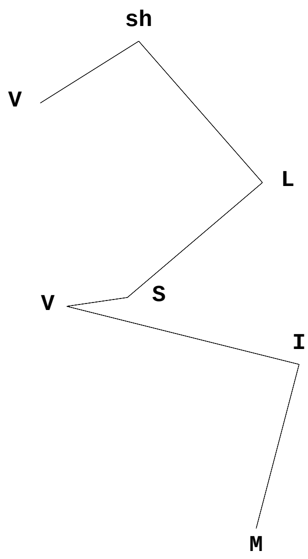
Figure 6.5 shows an example motif using these abbreviations, and its translation into a structured concept. It will be convenient for the next section to have in hand a textual notation for a motif: such a notation is illustrated at the bottom of Figure 6.5. A motif of length n will be written as a list of n conjunctive primitive concepts, each enclosed in brackets, followed by a structural sequence of length $n-3$. The kth element in this structural sequence will denote the Δ relation among components k through $k+3$.

6.3.2 Methods

A large database of 95 proteins was used to construct two datasets of fragments for conformational clustering. These proteins are an amalgamation of those used by Hunter and States (1991) and Rooman and Wodak (1991), and each has a resolution better than 2.5 Å

```
defprimconcept amino-acid any;

defprimconcept hydrophobic amino-acid;
defprimconcept hydrophylic amino-acid;
defprimconcept small amino-acid;
defprimconcept tiny small;
defprimconcept polar amino-acid;
defprimconcept charged polar;
defprimconcept positive charged;
defprimconcept negative charged;
defprimconcept aromatic hydrophobic;
defprimconcept aliphatic hydrophobic;

defprimconcept ALA (and hydrophobic tiny);
defprimconcept ARG (and positive hbonddon hydrophylic);
defprimconcept ASN (and small polar hbondacc hbonddon hydrophylic ASX);
defprimconcept ASP (and small negative hbondacc hydrophylic ASX);
defprimconcept CYS (and polar tiny hbonddon hbondacc);
defprimconcept GLN (and polar hbonddon hbondacc hydrophylic GLX);
defprimconcept GLU (and negative hbondacc hydrophylic GLX);
defprimconcept GLY (and hydrophobic small);
defprimconcept HIS (and aromatic positive hbonddon);
defprimconcept ILE aliphatic;
defprimconcept LEU aliphatic;
defprimconcept LYS (and hydrophobic positive hbonddon);
defprimconcept MET hydrophobic;
defprimconcept PHE aromatic;
defprimconcept PRO small;
defprimconcept SER (and tiny polar hbondacc hbonddon hydrophylic);
defprimconcept THR (and small polar hbondacc hbonddon hydrophobic);
defprimconcept TRP (and aromatic polar hbonddon);
defprimconcept TYR (and aromatic polar hbonddon hbondacc);
defprimconcept VAL (and aliphatic small);

defprimconcept hbonddon amino-acid;
defprimconcept hbondacc amino-acid;

defprimconcept GLX amino-acid;
defprimconcept ASX amino-acid;
defprimconcept UNK amino-acid;
```

Table 6.4: The $\mathcal{SDL}$ representation of the background knowledge used to represent the parts of protein motifs.



λ x.$\exists^*$ p1 p2 p3 p4 p5 p6 p7

```
    π(x,p1) ∧ ... ∧ π(x,p7) ∧
    Valine(p1) ∧
    small(p2) ∧ hydrophobic(p2) ∧
    Leucine(p3) ∧
    Serine(p4) ∧
    Valine(p5) ∧
    Isoleucine(p6) ∧
    Methionine(p7) ∧
    bonded(p1,p2) ∧
    bonded(p2,p3) ∧...∧
    L(p1,p2,p3,p4) ∧
    L(p2,p3,p4,p5) ∧
    L(p3,p4,p5,p6) ∧
    L(p4,p5,p6,p7)
```

(V)(sh)(L)(S)(V)(I)(M) LLLL

Figure 6.5: The logical equivalence semantics for a protein motif which preserves the `bonded` and Δ relations. Left: a depiction of a protein motif. Right: the translation of the motif into a structured concept. All negated `bonded` and Δ literals are omitted from the translation. Bottom: the motif in textual notation.

(see Table 6.5). They also span a variety of distinct protein structural families. Care was taken to omit from the amalgamated database protein chains with identical sequences. To create the first dataset, the 2D query of Figure 6.6 was applied to the database, producing a set of 16761 virtual backbone heptamer fragments.[3] Note the asymmetry in this query (unlike those of Chapter 5); this is a result of the directed edges in the 2D graph, due to the asymmetry of the `bonded` relation. A second 2D query was used to construct a dataset of 15448 20-mer fragments; this is identical to the heptamer query except for its length (and symmetry element). All fragments in both datasets preserved the `bonded` and Δ relations. The order of both datasets was randomized. Enantiomorphs were not considered to be equivalent (recall Section 5.1). The next section will report on the conformational clustering of the 7-mer and 20-mer datasets, but first I shall discuss some interesting issues that came up during the protein motif experiments.

[3]Unlike the CSD, the PDB is not bundled with sophisticated searching routines. I wrote some programs specialized to execute the heptamer and 20-mer queries on the PDB.

155C	1GP1 (A)	1SBT	2CCY (A)	2PKA (A)	3EST	4FD1
1ACX	1HIP	1SGT	2CDV	2PKA (B)	3FXC	4FXN
1BP2	1HMZ (A)	1SN3	2CGA (A)	2PRK	3ICB	4HHB (A)
1CC5	1HMZ (C)	1TIM (A)	2CNA	2RHE	3INS (A)	4HHB (B)
1CCR	1LZ1	1TON	2CTS	2SGA	3INS (B)	4LDH
1CPV	1MBC	1TPP	2CYP	2SNS	3RP2 (A)	4MDH (A)
1CRN	1MBD	1UBQ	2GN5	2SOD (O)	3SGB (E)	5CHA (A)
1CSE (E)	1MLT (A)	2ABX (A)	2LH4	2STV	3SGB (I)	5CPA
1CTF	1NXB	2ACT	2LHB	2YHX	3TLN	5PTI
1CY3	1PAZ	2ALP	2LZM	3BCL	3WGA (A)	5RXN
1ECD	1PCY	2APP	2MHR	3C2C	451C	6TMN (E)
1FB4 (H)	1PPT	2APR	2MT2	3CYT (O)	4ADH	7RSA
1FB4 (L)	1PSG	2AZA (A)	2OVO	3DFR	4APE	8CAT (A)
1GCR	1RHD	2B5C	2PAB (A)	3EBX	4DFR (A)	9PAP
1GD1 (O)	1RN3	2CAB				

Table 6.5: The proteins used to build the fragment datasets. Chain identifiers, where applicable, are in brackets.

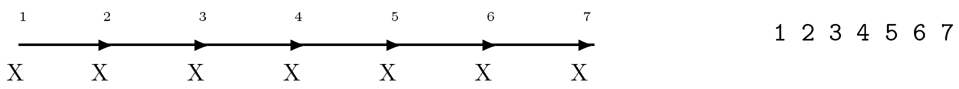


Figure 6.6: The heptamer fragment, with its symmetry element

Structure and sequence indexing in $\mathcal{SDL}$

A slight modification to the basic IMEM method was made: when a new structure-sequence motif was created, two extra concepts were formed. These two extra concepts are used for efficient query evaluation. One is a pure sequence concept which is created by dropping the Δ relations from the image term of a new structure-sequence motif. The other is a pure structure concept which is created by changing all primitive terms in the motif to X (i.e., any amino acid is permissible at that location). Figure 6.7 illustrates this process. In a properly-classified knowledge base it is easy to answer the query "what fragments have the same sequence as **3**" (answer: `inst`(**4**)), or "what other motifs have the same sequence as **3**" (answer: `fringe`(**4**)), or "how many fragments have the same structure as **3**" (answer: $|$`inst`(**5**)$|$). The need for efficient answers to these queries will become apparent in the following discussion on motif evaluation.

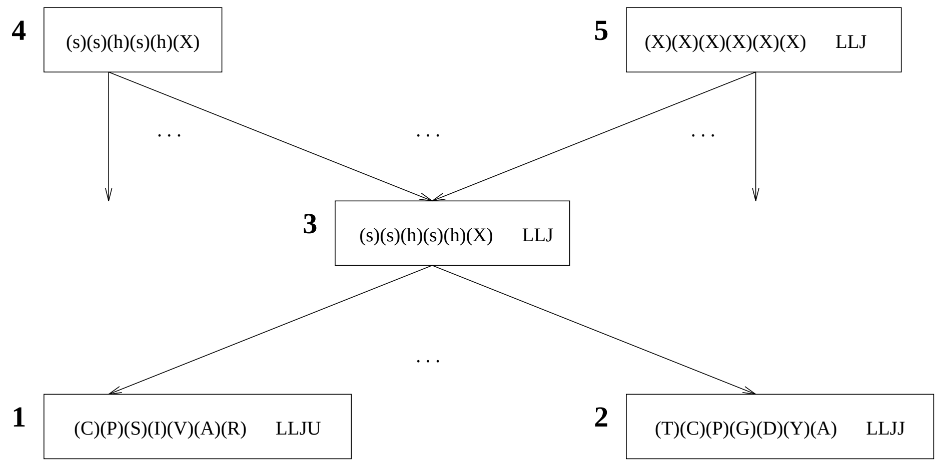


Figure 6.7: Protein motif sequence and structure indexing in $\mathcal{SDL}$. All motifs are in abbreviated form (see Figure 6.5). The least common subsumer of fragments **1** and **2** is motif **3**. Concepts **1**, **2**, and **3** preserve the `bonded` and Δ relations. The Δ relations are dropped from **3** to produce motif **4**, which is a pure sequence motif. The sequence of **3** is maximally generalized to produce the sequence of **5**, a structure-sequence motif. The "..." indicate that the motifs may subsume other motifs and fragments.

Motif evaluation

In Chapter 5, the concepts discovered by IMEM could be validated against known conformational classes and the numerical clustering results of Allen et al. For protein structure-sequence motifs there is no such evaluation scheme due to the novelty of the discovery task. Many motifs can be created, each representing a recurrent association between sequence and structure. Which ones represent interesting discoveries? For the PDB experiments, I decided to explore whether some of the motifs had a sequence which was predictive of local tertiary structure. To this end, those motifs for which the sequence is probably not predictive were filtered out from the original set. The indexing scheme described above greatly facilitated this process, since it gives efficient and accurate counts of independent sequence and structure motif instances.

Each discovered motif was passed through three successive filters. First, the ratio M_+/N — where N is the number of fragments subsumed by the sequence portion of the motif, and M_+ is the number of fragments subsumed by the structure–sequence motif — must be greater than 0.8. This ensures that more than 80% of the instances of the sequence motif have the same structure, and therefore that the sequence may be predictive of the corresponding structure. Second, the mean DRMS of all instances of the motif must be less than 0.95 Å. [4] This is done because it does not necessarily follow that motifs, qualitatively similar according to the Δ relations, are also quantitatively or "visually" similar. This filter ensures that the structure portion of the motif is a good 3D building block. Finally, the third filter applies a chi-square test of statistical independence to the discovery. This is a standard technique for evaluating the significance of a suspected association (Zembowicz and Zytkow 1992). A 2×2 contingency table for the motif is constructed (see Table 6.6), and a Bonferroni-adjusted chi-square test at a significance level of 0.01 is applied.[5] This test gives support to the conclusion that the sequence and structure portions of the motif are significantly correlated and not merely random associations. It satisfies Piatetsky-Shapiro's criteria for a "rule interest" measure (also see Section 4.4) and has some interesting properties. For example, all other things being equal, a sequence which occurs twice in a very common structure is less interesting than one which occurs twice in a rare structure. If a motif C passes through all three filters, then all motifs D such that $C \succeq_{\mathcal{T}} D$ are not considered. In this manner, the *most general* motifs satisfying all three filters are retained.

Similarity valuation

For the heptamer experiments, a small modification was made to the way similarity is measured (Chapter 4). In the heptamer experiments, I wanted to create structure-sequence motifs only from structurally equivalent fragments. This is not enforced by using a similarity threshold $\tau = 1.0$, since then two fragments must have identical parts to be candidates for concept formation. Instead, it is accomplished by modifying Expression 4.4 to consider only R-isomorphisms rather than R-monomorphisms. This modification does not affect the satisfaction of the similarity axioms in Chapter 4. For the 20-mer experiments, part deletion is allowed, and the original similarity definition of Expression 4.4 is used.

[4] Unger et al. (1989) consider two hexamer fragments X and Y to be compellingly similar if $\mathrm{ARMS}(X, Y) < 1.0$ Å. Substituting this value into the approximation given in Footnote 2 yields 0.95 Å.

[5] In a first pass through the motifs, only those which refute the null hypothesis of statistical independence at the 0.01 level of significance ($\chi^2 > 6.64$) are retained. However, the null hypothesis will be falsely rejected — a Type I error — for 1 in 100 tests. To preserve experiment-wise significance levels, it is therefore necessary to modify the chi-square test with a Bonferroni adjustment (Harris 1985). This divides the significance level by the number n of tests made, and produces a new critical value for χ^2. If n motifs pass the three filters then these motifs are sent through the third filter again, using an adjusted significance level of $0.01/n$.

Environments

A technique which was implicitly applied in the hexopyranose experiments of Chapter 5, and which is explicitly applied here to the 20-mer dataset, is the *method of environments.*[6] This technique restricts the number of possible potential monomorphisms between two concepts for subsumption and similarity computations during the conformational clustering process.

Under the method of environments, the only acceptable partial monomorphisms are those which are restrictions of an element of the atomic permutation symmetry group.[7]

As an illustration of this method, consider the heptamer query graph of Figure 6.6. Furthermore suppose, as was done in the hexopyranose experiments, that we are only interested in *connected* monomorphisms (a connected monomorphism maps a connected graph onto another connected graph). Using the method of environments, there are 7 partial monomorphisms of size 1, 6 of size 2, 5 of size 3,..., and 1 of size 7, leading to a total of 28 possible (connected) partial monomorphisms between two heptamer fragments. Without environments, there are 49 partial monomorphisms of size 1, 36 of size 2, 25 of size 3,..., and 1 of size 7, leading to a total of 140 (connected) partial monomorphisms: a substantial increase.

Referring to the symmetry group of Figure 6.6, Figure 6.8 shows a partial monomorphism which is disallowed under the method of environments. The figure depicts a partial monomorphism which is not a restriction of the sole element of the symmetry group. Disallowing this sort of matching is desirable in other respects: consider the case where both fragments are drawn from the same protein, with sequence indices as shown in Figure 6.8. The monomorphism in the figure is then matching identical entites. This may result in a spurious concept being created, since the concept does not necessarily represent a recurrent motif.

The chemical rationale behind the method of environments is that *within the environment defined by a specific query graph, only certain atoms should be put into correspondence.* These correspondences are precisely those which are restrictions of elements of the atomic permutation symmetry group.

---

[6]The term *environment* was suggested to me by Frank Allen.

[7]Defining the *graph* of a function f as $\{(x, f(x)) \ : \ x \in \mathrm{dom}(f)\}$, a function g is a *restriction* of f if and only if $\mathrm{graph}(g) \subseteq \mathrm{graph}(f)$.

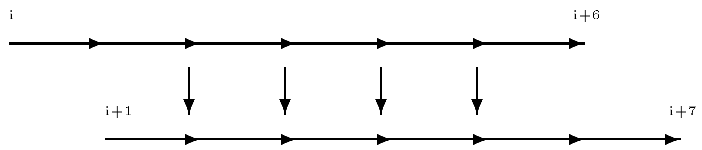


Figure 6.8: A partial monomorphism which is disallowed under the method of environments

	Structure		
Sequence	yes	no	Totals
yes	6	1	7
no	56	16698	16754
	62	16699	16761

Table 6.6: A sequence–structure contingency table used for the chi-square assessment of a discovered motif (motif 4451 of Table 6.7).

6.3.3 Experimental results

Heptamer motifs

The IMEM method, as described above, was applied to the heptamer dataset. A similarity threshold of $\tau = 0.55$ was used. The efficiency of classification in $\mathcal{SDL}$ degrades (at worst) linearly with the number of concepts in the taxonomy. Nevertheless, the fragment dataset is very large, and to reduce overall discovery time concept formation was disabled after 3000 fragments were processed. The remaining fragments were then classified: in this manner accurate statistics on the previously discovered motifs are obtained.

After incorporation of all 16761 heptamer fragments, 8309 unique concepts were created. After application of the three filters to the discovered motifs, 381 were retained. Together these 381 predictive motifs cover 1000 distinct fragments, or 6% of the fragment dataset. Some of the retained motifs are presented in Table 6.7. The mean DRMS for many motifs is very low. Some motifs (e.g., 4602 and 15665) have several specific conserved residues indicated, but some have only properties (e.g., 7529 and 14524). The sequences of several motifs are quite general: this indicates their wider applicability for tertiary structure prediction. Many of the structural sequences are `LLLL` — referring to the common helical structure — but several other structural sequences are also exhibited.

Id	Sequence	Structure	M_+	M_-	$\overline{\text{DRMS}}$	σ	χ^2
4451	(L)(h)(s)(yp)(ayp)(h)(V)	LLJU	6	1	0.76	0.73	1384.03
11925	(adps)(G)(adyp)(L)(s)(dp)(hs)	JJJU	3	0	0.61	0.29	782.81
12496	(s)(adps)(h)(P)(rdp)(Q)(V)	ZLZL	4	0	0.65	0.42	1152.21
11717	(r)(dhp)(G)(S)(L)(adps)(dhp)	JJUU	3	0	0.14	0.01	2183.61
10541	(rdp)(r)(adpt)(G)(s)(S)(h)	JZJU	4	0	0.68	0.45	954.00
15665	(l)(hs)(S)(h)(G)(S)(G)	JZUZ	3	0	0.45	0.24	1223.63
5302	(l)(adps)(A)(hs)(H)(C)(h)	LLJU	10	0	0.84	0.80	2694.99
4602	(G)(D)(S)(G)(G)(s)(h)	JZLZ	10	1	0.18	0.05	2106.76
14836	(h)(h)(T)(A)(hs)(H)(C)	LJUZ	10	0	0.21	0.06	3039.27
2641	(h)(G)(l)(hs)(S)(h)(G)	UZJJ	6	0	0.19	0.05	961.32
16290	(s)(l)(h)(l)(l)(c)(L)	JJJZ	9	0	0.29	0.16	497.47
2678	(h)(hs)(l)(T)(A)(h)(dp)	JUZJ	8	2	0.37	0.26	1210.11
2672	(r)(hs)(h)(T)(A)(hs)(H)	JUZJ	7	0	0.21	0.07	1326.82
16659	(L)(h)(hs)(h)(h)(K)(s)	LLLL	7	1	0.27	0.08	21.86
12404	(l)(hs)(A)(h)(h)(K)(p)	LLLL	6	0	0.20	0.08	23.09
15291	(l)(ayps)(ayp)(h)(h)(s)(cy)	LLLL	9	2	0.33	0.21	25.17
13958	(yp)(D)(h)(h)(h)(dhp)(l)	LLLL	9	0	0.38	0.25	34.64
3552	(X)(dyp)(rdp)(l)(X)(ayp)(hs)	LLLL	9	1	0.33	0.16	29.40
3485	(h)(s)(A)(X)(A)(L)(h)	LLLL	8	2	0.56	0.54	21.54
8914	(X)(A)(l)(ayp)(ayp)(A)(h)	LLLL	7	1	0.17	0.04	21.86
12323	(I)(h)(c)(h)(l)(h)(ayp)	LLLL	6	0	0.21	0.06	23.09
16173	(l)(h)(dhp)(hs)(l)(h)(ayn)	LLLL	6	0	0.21	0.06	23.09
7489	(K)(X)(h)(l)(ayn)(X)(hs)	LLLL	8	2	0.46	0.25	21.54
16151	(h)(A)(h)(hs)(c)(dhp)(hs)	LLLL	7	1	0.23	0.08	21.86
9105	(h)(s)(A)(h)(h)(dhp)(hs)	LLLL	15	2	0.31	0.10	47.50
12707	(l)(hs)(hs)(h)(hs)(A)(l)	LLLL	7	0	0.23	0.14	26.94
11350	(h)(s)(c)(I)(adp)(ayps)(h)	LLLL	6	0	0.68	0.32	23.09
13569	(G)(X)(D)(h)(h)(h)(dhp)	LLLL	7	1	0.52	0.34	21.86
9984	(hs)(ayps)(A)(h)(h)(adp)(h)	LLLL	8	2	0.24	0.06	21.54
7553	(s)(r)(h)(K)(dyp)(dp)(h)	LLLL	7	1	0.39	0.29	21.86
10792	(aps)(r)(l)(h)(adyp)(dhp)(h)	LLLL	6	0	0.26	0.10	23.09
7529	(s)(h)(l)(h)(adps)(dhp)(h)	LLLL	13	2	0.37	0.19	39.98
12155	(s)(h)(h)(h)(hs)(L)(dhp)	LLLL	7	1	0.37	0.29	21.86
9090	(s)(s)(A)(h)(adp)(ap)(A)	LLLL	7	0	0.21	0.08	26.94
14637	(h)(V)(s)(rdp)(h)(K)(X)	LLLL	7	0	0.23	0.08	26.94
7933	(hs)(h)(h)(adp)(A)(l)(h)	LLLL	7	1	0.27	0.11	21.86
14524	(h)(h)(ayp)(dhp)(l)(h)(ayp)	LLLL	14	3	0.46	0.26	39.59

Table 6.7: Examples of predictive protein structure–sequence heptamer motifs. Id: a unique identifier for the motif. M_+: the number of fragments subsumed by the motif; M_-: the number of fragments subsumed by the sequence, but not by the associated structure; $\overline{\text{DRMS}}$: mean DRMS (Section 6.2); σ: standard deviation; χ^2: chi-square value computed from the sequence–structure contingency table for the motif. Table 6.6 displays the contingency table for motif 4451. Appendix D gives one-letter abbreviations for all properties and amino acids.

Id	Sequence	Structure	M_+	M_-
2408	(h)(L)(R)(X)(G)(ayn)(Y)(X)(cy)(Y)(X)	ZULUZZLJ	2	0
2405	(X)(P)(adyps)(Y)(dyp)(G)(dp)(Q)(Y)(h)(L)	LUZLUJZJ	2	0
1947	(G)(S)(L)(I)(N)(ayp)(adyp)(W)(V)(V)(adps)(A)(A)(H)(C)(h)	JZUJLLZJZUJLL	3	0
1505	(h)(h)(l)(X)(dp)(s)(s)(h)(s)(Bayps)(Zayp)(dp)(dp)(X)(adyps)(dhp)	JZJJJLJZJZJZJ	3	0
2097	(ayp)(Bayps)(T)(L)(h)(ayn)(Y)(L)(X) (N)(P)(K)(K)(Y)(I)(P)(G)(T)(K)(M)	LLLLLLLUZLLLLJLZJ	2	0
1166	(N)(N)(h)(C)(C)(S)(Q)(rdp)(G)(dp)(C)(G)	ZJJJJLJZJ	3	0
1322	(G)(adyps)(L)(X)(adyp)(h)(S)(E)(Q)(E)(L)(l)(D)	JZJJJZLLLL	2	0
978	(T)(G)(S)(t)(D)(L)(W)(l)(h)(S)(T)(X)(X)	LJLJZZJZJL	2	0
1377	(l)(hs)(hs)(L)(h)(hs)(h)(h)(h)(h)(h)	LLLLLLL	6	0
2371	(s)(adp)(A)(cy)(r)(dp)(dhp)(h)(h)(s)(ayn)(X)(l)(h)	LLLLLLLLLL	3	0
1939	(s)(adp)(X)(s)(s)(h)(X)(ayp)(h)(l)(h)(ayp)(X)(s)(h)	JLLLLLLLLLJ	4	1
1120	(h)(p)(ayp)(h)(h)(hs)(X)(X)(X)(h)(dp)(h)(X)(h)(X)(adp)(X)	LLLLLLLLLLLLL	7	1
1509	(h)(X)(s)(hs)(h)(h)(X)(X)(h)(h)(h)(hs)(hs)(p)(X)(adp)(h)	LLLLLLLLLLLLL	5	1
2122	(s)(h)(hs)(p)(X)(dhp)(h)(h)(h)(s)(X)(h)(h)(h)(X)(X)(X)(h)(X)	LLLLLLLLLLLLLLL	5	0
297	(A)(dyp)(X)(aps)(h)(h)(h)(dp)(h)(V)(adp)(X)(h)(h)(A)(h)(s)	ZLLLLLLLLLLLL	4	0
1093	(c)(X)(X)(ayp)(h)(X)(hs)(ayp)(hs)(adps) (hs)(h)(h)(h)(t)(dhp)(K)(X)(dp)(X)	LLZLLLLLLLLLLLLLL	2	0
1776	(hs)(h)(X)(L)(l)(ayps)(X)(adyp)(h)(V)	JJZUJLL	6	0
2378	(X)(adyp)(h)(X)(h)(D)(T)(G)(adps)(s)(X)(L)(h)	ZJJZZLLJLJ	5	0
185	(h)(V)(l)(T)(A)(A)(H)(C)(h)	JZUJLL	6	0
180	(C)(X)(G)(D)(S)(G)(G)(s)(h)	LLZLZJ	8	1

Table 6.8: Examples of variable-length motifs

Variable-length motifs

Given the apparent success of the heptamer analysis, it is reasonable to ask: do there exist longer predictive motifs, and can these be automatically discovered? To answer these questions, I used IMEM to perform a conformational clustering of the 20-mer dataset. In contrast to the heptamer analysis, here IMEM was permitted to delete parts from fragments when creating least common subsumers. A similarity threshold of $\tau = 0.33$ was used. Only connected images were considered. Concept formation was disabled after 1000 fragments were processed. A total of 1001 motifs were discovered, and 189 of these passed through the three filters. Table 6.8 lists just a few of the discovered variable-length motifs. Motifs 2097 and 1093 have 20 parts, with several specific residues indicated in motif 2097. Motif 180 has a very specific sequence, yet covers eight fragments.

Motifs 180 and 185 are quite interesting in other respects. Motif 180 represents an abstraction of residues 191 through 199 in the family of *serine protease* proteins. This region represents the most highly conserved region in serine proteases, and contains the active-site serine residue 195 (Leytus et al. 1988). Motif 185, also conserved in the serine proteases, represents residues 51 through 59, and contains the active side histidine residue.

The PIMA method of Smith and Smith (1990) discovered a similar sequence motif, but there the class of training sequences was prepared carefully for discovery. Thus IMEM discovered, with no prior knowledge of protein families, a discriminating pattern responsible for biological activity. Equally interesting is the fact that motif 180 is not only a sequence motif: it has a 3D structure conserved to within 0.95 Å.

6.3.4 Discussion and future work

These results address a key question posed by Unger et al. (1989), showing that it is possible to automatically discover protein structural building blocks that carry sequence specificity. These structural building blocks are encapsulated in structure–sequence motifs and expressed as $\mathcal{SDL}$ concepts. Filters were used to prune the set of discovered motifs. Though the filters were tailored for sequence–structure predictivity, there is no reason why structure–sequence predictivity could not also be explored in this discovery framework. For such an experiment, it will probably be necessary to use long training fragments to allow for a wider distribution of structural sequences.

These results are important, as previously only a few predictive tertiary motifs have been identified, and there are few techniques for discovering them from data. Rooman et al. (1989) state that "presently, the only way to establish a fingerprint for a given structural motif is to construct it, test it in a larger database, and possibly modify it, until adequate discrimination power is achieved". Discovered tertiary motifs can be used for protein tertiary structure prediction and modelling, and increasing the length of discovered motifs may allow us to consider tertiary interactions which are nonlocal in the protein sequence. Interactions between distant residues, and the failing of structure prediction methods to take them into account, is one of the hypothesized reasons for the limited prediction success that has been achieved. In general, with longer motifs, more contiguous residues can be predicted, and less tertiary alignment of predicted portions needs to be performed. It is, however, too restrictive to discover motifs of only one size. An advantage of IMEM is that it can discover variable–length motifs.

In this chapter, a particular partioning of VBDA space has been used to describe the qualitative 3D structure of protein fragments and motifs. Although this partitioning has yielded results which clearly indicate correspondences between qualitative and visual similarity, it is conceivable that other partitionings could produce better results. Future research could address this topic, which is intimately related to the topic of constructive learning, briefly discussed in Chapter 2. A fully automated IMEM system should be able to propose its own ontology, or relations used to describe structured concepts.

The efficiency of the IMEM discovery algorithm depends on its incrementality combined with the use of an evolving concept taxonomy for fast close-match retrieval of concepts. Due to incrementality, a different ordering of the training data may produce quite a different set of predictive motifs. This effect has been confirmed empirically. Once a motif is discovered and classified, it hides its instances from future fragments which it does not subsume (although note that other motifs may subsume some of the instances). For example, in the experiments above, a sequence motif alone will hide some new fragments from subsumed structure–sequence motifs. Although not always desirable, the effect can be exploited by using sequence motifs suspected to be predictive (Rooman et al. 1989) as background knowledge. Driven by empirical data, the discovery system will elaborate these sequence motifs, and also confirm whether they occur in similar structures.

A visual inspection of several discovered motifs has revealed that pairs of motifs — while not subsuming each other — are often quite similar in their sequences. While they both may be predictive, their least common subsumer may not be. This points to the idea of extending $\mathcal{SDL}$ to deal with disjunction and negation to more concisely express the components of protein motifs. Indeed, parts of Taylor's (1986) original domain theory of amino acid types is expressed using disjunction and negation. Work on compiling a more extensive knowledge base of amino acids is in progress. It might also be worthwhile exploring other domain theories for amino acid types (e.g., Finzel et al. 1990; Smith and Smith 1990).

On a related note, previous structure–sequence discovery work has not explicitly represented the internal structure of amino acids. An extension of previous work is to base the discoveries of a system on the internal spatial structure of the amino acids. There are two approaches to such an extension. One is to code, as background knowledge, the definitions from manual amino acid rotamer classifications (Ponder and Richards 1987). Another approach is to use IMEM to autonomously discover its own rotamer classes. Both approaches require a knowledge representation, such as $\mathcal{SDL}$, capable of describing multilevel structured objects. Initial results on multilevel protein motif discovery with IMEM is reported by Conklin et al. (1994). Klingler and Brutlag (1994) also explore this topic, correlating amino acid sidechain conformations with helical secondary structures.

Hodgman (1989) distinguishes protein sequence motifs from sequence *patterns*, which are motifs which are linked together with a specification of maximum and minimum separation between motifs. It is possible that queries of the PDB, more sophisticated than the ones considered here, could be constructed to retrieve instances of such sequences which are then subjected to conformational analysis. Perhaps sequence alignment schemes could be used to propose interesting sequence patterns, and IMEM could identify their recurrent three-dimensional structures.

Finally, a more rigorous analysis of discovered motifs is necessary. This chapter has presented a method which discovers predictive protein motifs. Although a couple were analyzed in detail, a comprehensive comparison of discovered motifs with known functional motifs is outside the scope of this thesis, and is left for future research.

6.4 Conclusions

This chapter has presented and surveyed the important application domain of protein motif discovery. A novel representation and discovery scheme was applied, wherein all types of protein motifs have a model-theoretic semantics, and structured concept discovery is used to uncover recurrent motifs. The organization of a knowledge base with subsumption efficiently guides new training fragments to other similar fragments, which are generalized and classified into an evolving concept taxonomy. Some promising results on two large datasets of protein fragments were presented.

The main emphasis of this chapter was on structure–sequence motifs — patterns of 3D structure with attached sequence information. Until now, approaches to structure–sequence motif discovery were limited to the *a posteriori* construction of a conserved or property sequence motif for instances of a discovered structure motif. These approaches do not represent structure motifs associated with a disjunction of sequences, and will produce overly-general and weakly predictive structure prediction rules. The method presented in this chapter is one of the first to fully integrate structure and sequence motif representation and discovery. As this part of my dissertation has illustrated, the discovery of protein motifs is part of a broader research program which is concerned with the automated discovery of associations between molecular constitution and conformation in large molecular structure databases.

Chapter 7

Conclusions

This thesis has presented a novel technique for discovering recurrent structured concepts in structured data comprising objects with parts and their interrelationships. The technique was applied and evaluated on two large molecular structure databases, the Cambridge Structural Database and the Protein Data Bank. The problem solved in both cases was *conformational clustering*: given a graph representing some molecular constitution, group the examples into classes characterized by structured concepts. There are many recurrent conformational patterns which remain hidden in molecular structural databases. Conformational patterns can be used as structural templates for model building, and as rules for structure generation. They are an important part of the grammar of three-dimensional structural chemistry.

The discovery technique looks for similarities in a stream of examples, and expresses similar portions as generalized concepts. As discussed in the Introduction, a number of important issues had to be addressed during the development of this method. The *knowledge representation* issue was addressed in Chapter 3, where a new Spatial Description Logic $\mathcal{SDL}$ was developed. The logic belongs to the class of *direct* knowledge representation formalisms, which compute relations by directly inspecting models of structured objects or concepts. The semantics of $\mathcal{SDL}$ was given by a natural translation to first-order logic. The subsumption rules of $\mathcal{SDL}$ were shown to be sound and complete with respect to this semantics. The *similarity* issue was addressed in Chapter 4, where an axiomatic theory of similarity was presented. This theory dictates the desired properties of a similarity measure for structured concepts. The *evaluation* issue was addressed in Chapters 5 and 6. In Chapter 5 a complete clustering was evaluated according to its likeness to known and statistical classifications of the same data. In Chapter 6, single concepts — or structure-sequence protein motifs — were evaluated according to the ability of a sequence to predict an associated structure.

The main hypothesis of this thesis was that structured concept discovery techniques could be used to describe and discover interesting and useful classes of molecular conformations. Part I of this thesis presented the theoretical background and new methods used to construct a usable structured concept discovery system. Part II of this thesis applied and evaluated the system on molecular structure databases. Chapter 5 reported on the conformational clustering of several small molecular graphs, illustrating a wide variety of conformational phenomena. In all cases, the clusterings were chemically sensible. Chapter 6 reported on the conformational clustering of protein fragments using background knowledge. Stringent criteria were developed to filter out non-predictive motifs, and several predictive motifs were discovered. In addition to supporting the main hypothesis of this thesis, the results of Part II show that the theory and machinery described in Part I was general in scope and can be applied, with minor modifications, to two different types of molecular structure databases.

The methods presented in this thesis are a step towards the automation of the conformational clustering process; a quality deficient in earlier efforts. However, the programs are not yet fully automated. Substantial chemical knowledge goes into the construction of an original 2D query graph. A chemist will usually have in mind some prior assumptions for the conformational preferences of that graph. Also, in Chapter 5 it was necessary to parameterize some of the relations used to describe conformations, and a chemist must choose these parameters. Clearly what is desired in the future, however, is a move towards exploratory discovery systems, which explore the data and hypothesis space autonomously. These could propose parameters, choose relations for describing structures, and postulate interesting 2D molecular graphs for clustering. In the protein motif domain they could dynamically extend or focus attention on motifs known to be predictive.

The biological function of proteins and small organic molecules is directly related to their three-dimensional shape. One of the great challenges of modern science is the derivation and understanding of the laws by which molecules adopt their shapes. Computational knowledge discovery methods, which scan large molecular structure databases for pattern and regularity, will help to uncover these laws of nature.

Appendix A

Completeness of $\mathcal{SDL}$

Here is a proof, by structural induction, of step 2 in the completeness proof of Section 3.2.3:

$$\mathtt{exp}_{\mathcal{T}}(C) \succeq \mathtt{exp}_{\mathcal{T}}(D) \quad \Leftrightarrow \quad \overline{\mathtt{exp}_{\mathcal{T}}(C)}\ \theta\text{-subsumes}\ \overline{\mathtt{exp}_{\mathcal{T}}(D)}$$

Proof: let

$$\begin{aligned} \mathtt{exp}_{\mathcal{T}}(C) &= [I, R] \\ \mathtt{exp}_{\mathcal{T}}(D) &= [J, R'] \\ \overline{\mathtt{exp}_{\mathcal{T}}(C)} &= \lambda\mathtt{x}.(\exists^* \overline{y})\, \Pi(\mathtt{x}, \overline{y}) \wedge E \\ \overline{\mathtt{exp}_{\mathcal{T}}(D)} &= \lambda\mathtt{x}.(\exists^* \overline{y'})\, \Pi(\mathtt{x}, \overline{y'}) \wedge E' \end{aligned}$$

1. Consider the $\Rightarrow$ direction of the theorem. Assume $\mathtt{exp}_{\mathcal{T}}(C) \succeq \mathtt{exp}_{\mathcal{T}}(D)$. The proof that the consequent must be true is by structural induction.

a) Base case: C and D are primitive terms. Trivially shown to hold true. □

b) Induction step: C and D are image terms.

By the assumption that $\mathtt{exp}_{\mathcal{T}}(C) \succeq \mathtt{exp}_{\mathcal{T}}(D)$, $\exists f : I \xrightarrow{R} J\ \forall a \in I\ \mathtt{first}(a) \succeq \underline{\mathtt{first}(f(a))}$. Now, let v and w be the component-to-variable functions which were used to construct $\overline{\mathtt{exp}_{\mathcal{T}}(C)}$ and $\overline{\mathtt{exp}_{\mathcal{T}}(D)}$ (recall Section 3.2.2 and see the diagram below). Construct the substitution $\theta = v^{-1}; f; w$. Now, consider any literal l in E. If l is unary, say $l = r(a)$, with $r \in R$, then $\mathtt{first}(v^{-1}(a)) \succeq \mathtt{first}(f(v^{-1}(a)))$, and then $l\theta$ is in E' by the structural induction hypothesis. If $l = r(a_1, \ldots, a_n)$ then $r(v^{-1}(a_1), \ldots, v^{-1}(a_n))$ must have been $\mathtt{true}$, hence $r(f(v^{-1}(a_1)), \ldots, f(v^{-1}(a_n)))$ must also be $\mathtt{true}$, since f is an R-monomorphism. A similar argument applies when $l = \mathtt{not}\, r(a_1, \ldots, a_n)$. Hence for any literal l in E, $l\theta$ is in E'. □

2. Now consider the $\Leftarrow$ direction of the theorem. Assume $\overline{\mathtt{exp}_{\mathcal{T}}(C)}$ θ-subsumes $\overline{\mathtt{exp}_{\mathcal{T}}(D)}$. The proof that the consequent must be true is by structural induction.

a) Base case: C and D are primitive terms. Trivially shown to hold true. $\square$

b) Induction step: C and D are image terms.

By the assumption that $\overline{\mathtt{exp}_{\mathcal{T}}(C)}$ θ-subsumes $\overline{\mathtt{exp}_{\mathcal{T}}(D)}$, there exists an injective substitution $\theta : \overline{y} \rightarrow \overline{y'}$ such that $E\theta \subseteq E'$ (recall Section 2.2.3 and see the diagram below). Construct the monomorphism $f = v;\theta;w^{-1}$. By the structural induction hypothesis, f preserves subsumption between corresponding components. To show that f is an R-monomorphism, consider an n-ary $r \in R$ and components $i_1, \ldots, i_n$ in I. if $r(i_1, \ldots, i_n) = \mathtt{true}$, then $r(v(i_1), \ldots, v(i_n))$ is in E, and $r(\theta(v(i_1)), \ldots, \theta(v(i_n)))$ is in E'. Hence $r(f(i_1), \ldots, f(i_n))$ must have been $\mathtt{true}$. A similar argument applies when $r(i_1, \ldots, i_n) = \mathtt{false}$. Hence f is an R-monomorphism. $\blacksquare$

$$\begin{tikzcd}
\mathtt{exp}_{\mathcal{T}}(C) = [I, R] \arrow[rr, "f : I \xrightarrow{R} J = v;\theta;w^{-1}"] \arrow[d, "v : I \rightarrow \overline{y}"'] & & \mathtt{exp}_{\mathcal{T}}(D) = [J, R] \arrow[d, "w : J \rightarrow \overline{y'}"] \\
\overline{\mathtt{exp}_{\mathcal{T}}(C)} = \lambda \mathtt{x}.(\exists^* \overline{y})\ \Pi(\mathtt{x}, \overline{y}) \wedge E \arrow[rr, "\theta : \overline{y} \rightarrow \overline{y'} = v^{-1};f;w"] & & \overline{\mathtt{exp}_{\mathcal{T}}(D)} = \lambda \mathtt{x}.(\exists^* \overline{y'})\ \Pi(\mathtt{x}, \overline{y'}) \wedge E'
\end{tikzcd}$$

Appendix B

Subsumption in $\mathcal{SDL}$ is NP-complete

The fact that subsumption in $\mathcal{SDL}$ is NP-complete follows immediately from the observation that arbitrary undirected graphs can be expressed using a direct representation. This observation is somewhat interesting in its own right:

Definition 3 *The* unit sphericity $\mathrm{sph}(G)$ *of an undirected graph $G = (V, E)$ is defined as the minimum dimension n of a Euclidean space for which there exists a function $f : V \rightarrow \Re^n$ such that*

$$(v_1, v_2) \in E \quad \Leftrightarrow \quad \mathtt{d}(f(v_1), f(v_2)) \leq 1$$

Proposition 2 *For any undirected graph $G = (V, E)$,* $\mathrm{sph}(G) < |V|$ (Guttman 1979; Maehara 1984).

Theorem 3 *Subsumption in $\mathcal{SDL}$ is NP-complete.*

Proof: Let $G = (V_1, E_1)$ and $H = (V_2, E_2)$ be undirected graphs, with $|V_2| \geq |V_1|$. Embed G and H into $(|V_2| - 1)$-dimensional Euclidean space. Let the resultant images G', H' preserve the adjacency relation `adj`, where

$$\mathtt{adj(x,y)} \stackrel{\mathrm{def}}{=} \mathtt{d(second(x), second(y))} \leq 1$$

It is apparent that G is isomorphic to G' and that H is isomorphic to H'. In particular, G is a subgraph of H if and only if there is an `adj`-monomorphism from G' to H'. Thus subgraph isomorphism between undirected graphs, known to be NP-complete, is a special case of subsumption in $\mathcal{SDL}$. ■

Appendix C

Indexing theorem

The indexing theorem of Section 4.2 is restated here:

> Let $\mathcal{T}$ be an $\mathcal{SDL}$ terminology, C and D be any two concept terms such that $C \succeq_{\mathcal{T}} D$, and $S_{\mathcal{T}}$ be a similarity coefficient obeying the axioms of Table 4.1. For any concept term Q, if $D \succeq_{\mathcal{T}} Q$, then $S_{\mathcal{T}}(D,Q) \geq S_{\mathcal{T}}(C,Q)$. Furthermore, the inequality is strict whenever the relation $C \succeq_{\mathcal{T}} D$ is proper (i.e., $C \not\equiv_{\mathcal{T}} D$).

Proof: let $S_{\mathcal{T}}$ be a similarity coefficient obeying the axioms of Table 4.1. If $C \equiv_{\mathcal{T}} D$, then, by similarity axiom 2, $S_{\mathcal{T}}(D,Q) = S_{\mathcal{T}}(C,Q)$. The proof of the case where $C \not\equiv_{\mathcal{T}} D$ is by structural induction on concept terms.

a) Base case: C, D, and Q are primitive terms. By the assumptions that $C \succeq_{\mathcal{T}} D \succeq_{\mathcal{T}} Q$, and that $C \not\equiv_{\mathcal{T}} D$, and by the completeness of $\mathcal{SDL}$ (Appendix A), it follows that $\mathtt{exp}_{\mathcal{T}}(C) \subset \mathtt{exp}_{\mathcal{T}}(D) \subseteq \mathtt{exp}_{\mathcal{T}}(Q)$. Similarity axiom 4 states that $S_{\mathcal{T}}(C,Q) < S_{\mathcal{T}}(D,Q)$. $\square$

b) Induction step.

Recall that we are trying to show that $S_{\mathcal{T}}(C,Q) < S_{\mathcal{T}}(D,Q)$, given the structural induction hypothesis, and the assumptions that $C \succeq_{\mathcal{T}} D \succeq_{\mathcal{T}} Q$ and $C \not\equiv_{\mathcal{T}} D$. In the following, the relation sets of any two image terms is assumed to be a constant R. This can be done because similarity is defined only over image terms with the same relation sets. The assumption allows for two notational simplifications. First, we can assume that monomorphisms are between image terms, rather than symbolic images, and the R can be dropped from $\xrightarrow{R}$. Second, the image `overlay` operator (here abbreviated $\cup$) can be defined over image terms in the obvious way.

Let

$$S_{\mathcal{T}}(C,D) = S_{\mathcal{T}}(C,D,h)$$
$$S_{\mathcal{T}}(D,Q) = S_{\mathcal{T}}(D,Q,g)$$

that is, g and h are the functions which maximize (4.4) (see the diagram below). It is a simple observation that $h : C \to D$ and $g : D \to Q$ must be subsumption-preserving monomorphisms.

Now consider the monomorphism $f = h;g : C \to Q$, and let

$$D = X \cup X'$$
$$Q = Y \cup Y' \cup Y''$$

where X is the range of h, and Y is the range of f. It is a simple observation that $C \succeq_{\mathcal{T}} X \succeq_{\mathcal{T}} Y$, and that

$$S_{\mathcal{T}}(C,Q) = S_{\mathcal{T}}(C,Q,f)$$

By the structural induction hypothesis, and the fact that $C \succeq_{\mathcal{T}} X \succeq_{\mathcal{T}} Y$, it follows that $S_{\mathcal{T}}(C, Y) \leq S_{\mathcal{T}}(X, Y)$. By multiple application of the part replacement rule (similarity axiom 5), we have

$$S_{\mathcal{T}}(C, Y \cup Y' \cup Y'') \leq S_{\mathcal{T}}(X, Y \cup Y' \cup Y'')$$

Now $D = X \cup X'$ will have one of two forms: either X' is empty or nonempty. If X' is empty, then by the assumption that $C \not\equiv_{\mathcal{T}} X$, and by similarity axioms 4 and 5, the inequality above is strict:

$$S_{\mathcal{T}}(C, Y \cup Y' \cup Y'') < S_{\mathcal{T}}(X, Y \cup Y' \cup Y'')$$

or

$$S_{\mathcal{T}}(C, Q) < S_{\mathcal{T}}(D, Q)$$

If X' is nonempty, then by similarity axiom 3 we have

$$S_{\mathcal{T}}(X, Y \cup Y' \cup Y'') < S_{\mathcal{T}}(X \cup X', Y \cup Y' \cup Y'')$$

from which it follows that

$$S_{\mathcal{T}}(C, Y \cup Y' \cup Y'') \leq S_{\mathcal{T}}(X, Y \cup Y' \cup Y'') < S_{\mathcal{T}}(X \cup X', Y \cup Y' \cup Y'')$$

which is to say that

$$S_{\mathcal{T}}(C, Q) < S_{\mathcal{T}}(D, Q)$$

proving the indexing theorem. ■

$$\begin{tikzcd} C \arrow[d, "h"] \arrow[dr, "f = h;g"] & \\ \underset{(X \cup X')}{D} \arrow[r, "g"] & \underset{(Y \cup Y' \cup Y'')}{Q} \end{tikzcd}$$

Appendix D

Amino acid abbreviations

Abbreviation	Description	Alternative
A	Alanine	ALA
C	Cysteine	CYS
D	Aspartic acid	ASP
E	Glutamic acid	GLU
F	Phenylalanine	PHE
G	Glycine	GLY
H	Histidine	HIS
I	Isoleucine	ILE
K	Lysine	LYS
L	Leucine	LEU
M	Methionine	MET
N	Asparagine	ASN
P	Proline	PRO
Q	Glutamine	GLN
R	Arginine	ARG
S	Serine	SER
T	Threonine	THR
V	Valine	VAL
W	Tryptophan	TRP
Y	Tyrosine	TYR
		Amino acids with property
X	any amino acid	all
t	tiny	ACS
h	hydrophobic	AFGHIKLMTVWY
y	hydrophylic	SNDEQR
s	small	ACDGNPSTV
c	charged	DEHKR
v	positive	HKR
n	negative	DE
r	aromatic	FHWY
l	aliphatic	ILV
d	hydrogen bond donor	WYHTKCSNQR
a	hydrogen bond acceptor	YTCSDENQ
p	polar	CDEHKNQRSTWY
Z	GLX	QE
B	ASX	DN

Bibliography

Agrawal, R.; Borgida, A.; and Jagadish, H. 1989. Efficient management of transitive relationships in large data and knowledge bases. In *ACM SIGMOD '89*. 253–262.

Agrawal, R.; Imielinksi, T.; and Swami, A. 1993. Database mining: A performance perspective. *IEEE Transactions on Knowledge and Data Engineering* 5(6):985–987. Special Issue on Learning and Discovery in Knowledge-Based Databases.

Ait-Kaci, H.; Boyer, R.; Lincoln, P.; and Nasr, R. 1989. Efficient implementation of lattice operations. *ACM TOPLAS* 11(1):115–146.

Aldenderfer, M. S. and Blashfield, R. K. 1984. *Cluster Analysis.* Sage Publications.

Allen, F. H.; Davies, J.; Galloy, J.; Johnson, O.; Kennard, O.; Macrae, C.; Mitchell, E.; Mitchell, G.; Smith, J.; and Watson, D. 1991a. The development of Versions 3 and 4 of the Cambridge Structural Database System. *J. Chem. Inf. Comp. Sci.* 31:187–204.

Allen, F. H.; Doyle, M. J.; and Taylor, R. 1991b. Automated conformational analysis from crystallographic data. 1. A symmetry–modified single–linkage clustering algorithm for three–dimensional pattern recognition. *Acta Crystallographica* B47:29–40.

Allen, F. H.; Doyle, M. J.; and Taylor, R. 1991c. Automated conformational analysis from crystallographic data. 2. Symmetry-modified Jarvis-Patrick and complete-linkage clustering algorithms for three-dimensional pattern recognition. *Acta Crystallographica* B47:41–49.

Allen, F. H.; Doyle, M. J.; and Taylor, R. 1991d. Automated conformational analysis from crystallographic data. 3. Three-dimensional pattern recognition within the Cambridge Structural Database System: Implementation and practical examples. *Acta Crystallographica* B47:50–61.

Allen, F. H.; Rowland, R. S.; Fortier, S.; and Glasgow, J. I. 1992. Knowledge acquisition from crystallographic databases: towards a knowledge-based approach to molecular scene analysis. *Tetrahedron Computer Methodology* 3(6C):757–774.

Allen, F. H.; Howard, J. A. K.; and Pitchford, N. A. 1993a. Symmetry-modified conformational mapping and classification of the medium rings from crystallographic data. 1. Cycloheptane. *Acta Crystallographica* B49:910–928.

Allen, F. H.; Kennard, O.; Galloy, J.; Johnson, O.; and Watson, D. G. 1993b. 3-D searching and numerical analyses applied to files of crystallographic data: methodologies, examples, and integration with 1-D and 2-D techniques. In Warr, W., editor, *Chemical Structures 2*. Springer-Verlag.

Allen, F. H.; Howard, J. A. K.; Pitchford, N. A.; and Vinter, J. G. 1994. Symmetry-modified conformational mapping and classification of the medium rings from crystallographic data. 2. Exo-unsaturated and heterocyclic seven-membered rings. *Acta Crystallographica* B50:328.

Allen, F. H.; Bath, P.; and Willett, P. 1995. Angular spectroscopy: Rapid visualization of three-dimensional substructure dissimilarity using valence angle or torsional descriptors. *J. Chem. Inf. Comp. Sci.* To appear.

Allen, F. H. and Fortier, S. 1993. Stereochemical and conformational classification of the hexopyranose sugars using numerical clustering methods. *Acta Crystallographica* B49:1021.

Allen, F. H. and Taylor, R. 1991. Automated conformational analysis from crystallographic data. 5. Recognition of special positions in conformational space in symmetry-modified clustering algorithms. *Acta Crystallographica* B47:404–412.

Anderson, J. R. 1983. *The Architecture of Cognition.* Harvard University Press.

Baader, F.; Hollunder, B.; Nebel, B.; Profitlich, H.; and Franconi, E. 1992. An empirical analysis of optimization techniques for terminological representation systems. In Nebel, B.; Rich, C.; and Swartout, W., editors, *KR92 : Principles of Knowledge Representation and Reasoning.* 270–281.

Baader, F. and Hollunder, B. 1991. $\mathcal{KRIS}$: Knowledge Representation and Inference System. *SIGART Bulletin* 2(3):8–14.

Barrow, H. G.; Ambler, A. P.; and Burstall, R. M. 1972. Some techniques for recognising structure in pictures. In Watanabe, S., editor, *Frontiers of Pattern Recognition.* Academic Press. 1–29.

Barrow, H. G. and Burstall, R. M. 1976. Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters* 4(4):83–84.

Barwise, J. and Etchemendy, J. 1992. Hyperproof: Logical reasoning with diagrams. In Narayanan, N. H., editor, *Proceedings of the AAAI Spring Symposium on Reasoning with Diagrammatic Representations.* 80–84.

Beck, H. W.; Gala, S. K.; and Navathe, S. B. 1989. Classification as a query processing technique in the CANDIDE semantic data model. In *Proc. IEEE International Data Engineering Conference.* 572–581.

Bernstein, F. C.; Koetzle, T. F.; Williams, J. B.; Meyer Jr., E. F.; Brice, M. D.; Rodgers, J. R.; Kennard, O.; Shimanouchi, T.; and Tasumi, M. 1977. The Protein Data Bank: A computer–based archival file for macromolecular structures. *Journal of Molecular Biology* 112:535–542.

Bezdek, J. C. and Pal, S. K. 1992. *Fuzzy models for pattern recognition: methods that search for structures in data.* IEEE Press.

Bisson, G. 1992a. Conceptual clustering in a first order logic representation. In Neumann, B., editor, *Proc. ECAI–92: Tenth European Conference on Artificial Intelligence.* John Wiley and Sons. 458–462.

Bisson, G. 1992b. Learning in FOL with a similarity measure. In *Proc. AAAI-92.* The MIT Press. 82–87.

Blundell, T. L.; Sibanda, B. L.; Sternberg, M. J. E.; and Thornton, J. M. 1987. Knowledge–based prediction of protein structures and the design of novel molecules. *Nature* 326:347–352.

Borgida, A.; Brachman, R.; McGuinness, D.; and Resnick, L. 1989. CLASSIC: A structural data model for objects. In *ACM SIGMOD '89.* 58–67.

Brachman, R. J. and Schmolze, J. G. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive Science* 9:171–216.

Brachman, R. J. 1983. What ISA is and isn't: An analysis of taxonomic links in semantic networks. *IEEE Computer* 16(10):30–36.

Bucher, P. and Bairoch, A. 1994. A generalized profile syntax for biomolecular sequence motifs and its function in automatic sequence interpretation. In Altman, R.; Brutlag, D.; Karp, P.; Lathrop, R.; and Searls, D., editors, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology.* AAAI Press. 53–61.

Buntine, W. 1988. Generalized subsumption and its application to induction and redundancy. *Artificial Intelligence* 36:149–176.

Burkert, U. and Allinger, N. L. 1982. *Molecular Mechanics.* American Chemical Society.

CCDC, 1994. *Cambridge Structural Database System: Getting Started.* Cambridge, England. User's manual, Cambridge Crystallographic Data Centre.

Chang, S. K.; Shi, Q. Y.; and Yan, C. W. 1987. Iconic indexing by 2-D strings. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9(3):413–428.

Chang, S. K. and Liu, S. Y. 1984. Picture indexing and abstraction techniques for pictorial databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6(4):475–484.

Charniak, E. and McDermott, D. 1985. *Introduction to Artificial Intelligence.* Addison-Wesley.

Cheeseman, P.; Kelly, J.; Self, M.; Stutz, J.; Taylor, W.; and Freeman, D. 1988. Autoclass: A Bayesian classification system. In *Proceedings of the Fifth International Workshop on Machine Learning.*

Chothia, C. 1992. One thousand families for the molecular biologist. *Nature* 357:544–545.

Claessens, M.; Van Cutsem, E.; Lasters, I.; and Wodak, S. 1989. Modelling the polypeptide backbone with spare parts from known protein structures. *Protein Engineering* 2(5):335–345.

Cohen, W.; Borgida, A.; and Hirsh, H. 1992. Computing least common subsumers in description logics. In *Proc. AAAI-92.* The MIT Press. 754–760.

Cohen, P. R. and Feigenbaum, E. A. 1982. *The Handbook of Artificial Intelligence*, volume 3. Kaufmann.

Cohen, F. E. and Sternberg, M. J. E. 1980. On the prediction of protein structure: the significance of the root-mean-square deviation. *Journal of Molecular Biology* 138:321–333.

Colloc'h, N.; Etchebest, C.; Thoreau, E.; Henrissat, B.; and Mornon, J. P. 1993. Comparison of three algorithms for the assignment of secondary structure in proteins: the advantages of a consensus assignment. *Protein Engineering* 6(4):377–382.

Conklin, D.; Fortier, S.; Glasgow, J.; and Allen, F. 1992. Discovery of spatial concepts in crystallographic databases. In Zytkow, J. M., editor, *Proceedings of the ML92 Workshop on Machine Discovery*, Aberdeen, Scotland. 111–116.

Conklin, D.; Fortier, S.; and Glasgow, J. 1993a. Knowledge discovery in molecular databases. *IEEE Transactions on Knowledge and Data Engineering* 5(6):985–987. Special Issue on Learning and Discovery in Knowledge-Based Databases.

Conklin, D.; Fortier, S.; and Glasgow, J. 1993b. Representation for discovery of protein motifs. In Hunter, L.; Searls, D.; and Shavlik, J., editors, *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology.* AAAI Press. 101–108.

Conklin, D.; Fortier, S.; and Glasgow, J. 1994. Knowledge discovery of multilevel protein motifs. In Altman, R.; Brutlag, D.; Karp, P.; Lathrop, R.; and Searls, D., editors, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology.* AAAI Press. 96–102.

Conklin, D. and Glasgow, J. 1992. Spatial analogy and subsumption. In Sleeman, D. and Edwards, P., editors, *Machine Learning: Proceedings of the Ninth International Conference (ML92).* Morgan Kaufmann. 111–116.

Conklin, D. and Jenkins, M. A. 1993. Compilation of description logics. Unpublished research report.

Conklin, D. and Witten, I. H. 1994. Complexity–based induction. *Machine Learning* 16(3):203–225.

Conklin, D. 1993. Transformation-invariant indexing and machine discovery for computer vision. In Bowyer, K. and Hall, L., editors, *Machine Learning in Computer Vision: What, Why, and How?*, number FSS-93-04 in AAAI 1993 Fall Symposium Series. 10–14.

Connell, J. and Brady, M. 1987. Generating and generalising models of visual objects. *Artificial Intelligence* 31:159–183.

Cook, D. J. and Holder, L. B. 1994. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research* 1:231–255.

Crandell, C. W. and Smith, D. H. 1983. Computer-assisted examination of compounds for common three-dimensional substructures. *J. Chem. Inf. Comp. Sci.* 23:186–197.

Crippen, G. M. and Havel, T. F. 1988. *Distance Geometry and Molecular Conformation.* John Wiley & Sons.

Crossley, J. N.; Ash, C. J.; Brickhill, C. J.; Stillwell, J. C.; and Williams, N. H. 1972. *What is mathematical logic?* Dover.

Davies, T. R. and Russell, S. J. 1987. A logical approach to reasoning by analogy. In *Proceedings of the International Joint Conference on AI.*

Dey, L.; Das, P.; and Chaudhury, S. 1993. Recognition and learning of unknown objects in a hierarchical knowledge base. In Bowyer, K. and Hall, L., editors, *Machine Learning in Computer Vision: What, Why, and How?*, number FSS-93-04 in AAAI 1993 Fall Symposium Series. 15–19.

Dietterich, T. G. and Michalski, R. S. 1981. Inductive learning of structural descriptions. *Artificial Intelligence* 16:257–294.

Dolata, D. P.; Leach, A. R.; and Prout, C. K. 1987. Wizard: artificial intelligence in conformational analysis. *Journal of Computer–Aided Molecular Design* 1:73–85.

Dolata, D. P. 1986. Molecular modelling by symbolic logic. In *3rd European Seminar on Computer-Aided Molecular Design.* IBC Technical Services.

Doyle, J. and Patil, R. S. 1991. Two theses of knowledge representation: Language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence* 48(3):261–297.

Duax, W. L.; Griffin, J. F.; Rohrer, D. C.; and Weeks, C. M. 1980. Conformational analysis of sterols: comparison of X-ray crystallographic observations with data from other sources. *Lipids* 15(9):782–792.

Ellis, G. 1991. Compiling conceptual graphs. In *Processing Declarative Knowledge '91 : Lecture Notes in Artificial Intelligence 567.* Springer-Verlag. 41–55.

Eshera, M. A. and Fu, K. S. 1984. A graph distance measure for image analysis. *IEEE Trans. Systems, Man, and Cybernetics* 14(3):398–408.

Falkenhainer, B.; Forbus, K. D.; and Gentner, D. 1989. The structure–mapping engine: algorithm and examples. *Artificial Intelligence* 41:1–63.

Finzel, B. C.; Kimatian, S.; Ohlendorf, D. H.; Wendoloski, J. J.; Levitt, M.; and Salemme, F. R. 1990. Molecular modeling with substructure libraries derived from known protein structures. In Bugg, C. E. and Ealick, S. E., editors, *Crystallographic and Modelling Methods in Molecular Design.* Springer-Verlag. 175–188.

Fisher, D. H. and Langley, P. 1986. Conceptual clustering and its relation to numerical taxonomy. In Gale, W. A., editor, *Artificial Intelligence and Statistics.* Addison-Wesley. chapter 4, 77–116.

Fisher, D. H. and Pazzani, M. 1991. Theory-guided concept formation. In Fisher, D. H.; Pazzani, M.; and Langley, P., editors, *Concept Formation: Knowledge and Experience in Unsupervised Learning.* Morgan Kaufmann. 165–177.

Fisher, D. H. 1987. Knowledge acquisition via incremental conceptual clustering. *Machine Learning* 2:139–172.

Fortier, S.; Castleden, I.; Glasgow, J.; Conklin, D.; Walmsley, C.; Leherte, L.; and Allen, F. 1993. Molecular scene analysis: The integration of direct–methods and artificial–intelligence strategies for solving protein crystal structures. *Acta Crystallographica* D49:168–178.

Frege, G. 1884. *The Foundations of Arithmetic.* English translation by J. L. Austin, Northwestern University Press, 1968.

Funt, B. V. 1980. Problem–solving with diagrammatic representations. *Artificial Intelligence* 13(3):201–230.

Garey, M. R. and Johnson, D. S. 1979. *Computers and intractability: A guide to the theory of NP-completeness.* W. H. Freeman.

Genesereth, M. R. and Nilsson, N. J. 1988. *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann.

Gennari, J. H.; Langley, P.; and Fisher, D. 1989. Models of incremental concept formation. *Artificial Intelligence* 40:11–61.

Gentner, D. 1989. The mechanisms of analogical learning. In Vosniadou, S. and Ortony, A., editors, *Similarity and Analogical Reasoning.* Cambridge University Press. 199–237.

Glasgow, J. 1993. The imagery debate revisited: A computational perspective. *Computational Intelligence* 9(4):309–333.

Glasgow, J. 1994. Spatial reasoning in indeterminate worlds. In *Proc. AAAI-94.* AAAI Press.

Goebel, R. 1993. Spatial and visual reasoning: humans versus logic. *Computational Intelligence* 9(4):406–412.

Goguen, J.; Thatcher, J.; and Wagner, E. 1978. An initial algebra approach to the specification, correctness, and implementation of abstract data types. In *Current trends in Programming Methodology*, volume 4. Prentice Hall. 80–149.

Golender, V. E. and Rozenblit, A. B. 1983. *Logical and Combinatorial Algorithms for Drug Design.* Research Studies Press.

Goodman, N. 1951. *The Structure of Appearance.* Bobbs-Merrill.

Gottlob, G. 1987. Subsumption and implication. *Information Processing Letters* 24:109–111.

Gray, N. A. B. 1986. *Computer-Assisted Structure Elucidation.* John Wiley & Sons.

Guttman, L. 1979. A definition of dimensionality and distance for graphs. In Lingoes, J.; Ruskam, E.; and Borg, I., editors, *Geometric Representations of Relational Data.* Mathesis Press. 713–723.

Hall, R. P. 1989. Computational approaches to analogical reasoning: A comparative analysis. *Artificial Intelligence* 39:39–120.

Handa, K.; Nishikimi, M.; and Matsubara, H. 1994. Utilizing structure information in concept formation. In Furukawa, K.; Michie, D.; and Muggleton, S., editors, *Machine Intelligence 13.* Oxford University Press. 123–142.

Haralick, R. M. and Shapiro, L. G. 1993. *Computer and Robot Vision*, volume 2. Addison-Wesley.

Harel, D. 1988. On visual formalisms. *Communications of the ACM* 31(5):514–530.

Harris, R. J. 1985. *A Primer of Multivariate Statistics.* Academic Press.

Haussler, D. 1989. Learning conjunctive concepts in structural domains. *Machine Learning* 4:7–40.

Hayes, P. J. 1985a. The logic of frames. In Brachman, R. J. and Levesque, H. J., editors, *Readings in knowledge representation.* Morgan Kaufmann.

Hayes, P. J. 1985b. Some problems and non–problems in representation theory. In Brachman, R. J. and Levesque, H. J., editors, *Readings in knowledge representation.* Morgan Kaufmann.

Hodgman, T. C. 1989. The elucidation of protein function by sequence motif analysis. *Computer Applications in the Biosciences* 5(1):1–13.

Hunter, L. and States, D. J. 1991. Bayesian classification of protein structural elements. In Hunter, L., editor, *Proc. Seventh IEEE Conf. on AI Applications: The Biotechnology Computing Minitrack*.

Indurkhya, B. 1989. Modes of analogy. In Jantke, K. P., editor, *Analogical and inductive inference*. Springer–Verlag. 217–230.

Jarvis, R. A. and Patrick, E. A. 1973. Clustering using a similarity measure based on shared nearest neighbors. *IEEE Trans. Comput.* 22:1025–1034.

Jones, D. T.; Taylor, W. R.; and Thornton, J. M. 1992. A new approach to protein fold recognition. *Nature* 358:86–89.

Jones, T. A. and Thirup, S. 1986. Using known substructures in protein model building and crystallography. *EMBO* 5(4):819–822.

Kabsch, W. and Sander, C. 1983. Dictionary of protein secondary structure. *Biopolymers* 22:2577–2637.

Kabsch, W. 1976. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica* B32:922–923.

King, R. D. and Sternberg, M. J. E. 1990. Machine learning approach for the prediction of protein secondary structure. *Journal of Molecular Biology* 216:441–457.

Klinger, A. and Pizano, A. 1989. Visual structure and data bases. In Kunii, T. L., editor, *Visual Database Systems*. Elsevier Science Publishers. 3–25.

Klingler, T. M. and Brutlag, D. L. 1994. Discovering side-chain correlations in α-helices. In Altman, R.; Brutlag, D.; Karp, P.; Lathrop, R.; and Searls, D., editors, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*. AAAI Press. 236–243.

Klyne, W. and Prelog, V. 1960. Description of steric relationships across single bonds. *Experientia* 16:521–523.

Langley, P. and Carbonell, J. 1984. Approaches to machine learning. *J. American Society for Information Science* 35(5):306–316.

Larkin, J. and Simon, H. 1987. Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science* 11:65–99.

Lathrop, R. H.; Webster, T. A.; and Smith, T. F. 1987. ARIADNE: Pattern–directed inference and hierarchical abstraction in protein structure recognition. *Communications of the ACM* 30(11):909–921.

Lathrop, R. H.; Webster, T. A.; Smith, R.; Winston, P.; and Smith, T. 1993. Integrating AI with sequence analysis. In Hunter, L., editor, *Artificial Intelligence and Molecular Biology*. AAAI/MIT Press. chapter 6.

Leach, A. R.; Prout, K.; and Dolata, D. P. 1988. An investigation into the construction of molecular models by the template joining method. *Journal of Computer–Aided Molecular Design* 2:107–123.

Lebowitz, M. 1986a. Concept learning in a rich input domain: generalization-based memory. In Michalski, R.; Carbonell, J.; and Mitchell, T., editors, *Machine Learning: An Artificial Intelligence Approach*, volume II. Morgan Kaufmann. 193–214.

Lebowitz, M. 1986b. Integrated learning: Controlling explanation. *Cognitive Science* 10:219–240.

Lebowitz, M. 1987. Experiments with incremental concept formation: UNIMEM. *Machine Learning* 2:103–138.

Levesque, H. J. and Brachman, R. J. 1987. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence* 3:78–92.

Levesque, H. 1986. Making believers out of computers. *Artificial Intelligence* 30:81–108.

Levinson, R. A. 1985. *A Self–Organizing Retrieval System for Graphs.* Ph.D. Dissertation, University of Texas at Austin.

Levinson, R. 1992. Pattern associativity and the retrieval of semantic networks. *Computers Math. Applic.* 23(6):573–600.

Levinson, R. A. 1994. UDS: A universal data structure. Technical Report UCSC-CRL-94-15, University of California, Santa Cruz.

Levitt, M. and Greer, J. 1977. Automatic identification of secondary structure in globular proteins. *Journal of Molecular Biology* 114:181–239.

Levitt, M. 1992. Accurate modeling of protein conformation by automatic segment matching. *Journal of Molecular Biology* 226:507–533.

Leytus, S. P.; Loeb, K. R.; Hagen, F. S.; Kurachi, K.; and Davie, E. W. 1988. A novel trypsin-like serine protease (hepsin) with a putative transmembrane domain expressed by human liver and hepatoma cells. *Biochemistry* 27:1067–1074.

Lloyd, J. W. 1987. *Foundations of logic programming.* Springer–Verlag.

MacGregor, R. and Brill, D. 1992. Recognition algorithms for the LOOM classifier. In *Proc. AAAI-92.* The MIT Press. 774–779.

MacGregor, R. M. 1991. Inside the LOOM description classifier. *SIGART Bulletin* 2(3).

Maehara, H. 1984. Space graphs and sphericity. *Discrete Applied Mathematics* 7:55–64.

Maggiora, G.M. and Johnson, M.A. 1990. Concepts and applications of molecular similarity. In Johnson, M. A. and Maggiora, G. M., editors, *Introduction to similarity in chemistry.* John Wiley & Sons. 1–14.

Mallgren, W. R. 1982. Formal specification of graphical data types. *ACM Transactions on Programming Languages and Systems* 4(4):687–710.

Markovitch, S. and Scott, P. D. 1993. Information filtering: selection mechanisms in learning systems. *Machine Learning* 10:113–151.

Marr, D. 1982. *Vision.* Freeman.

Matheus, C. J.; Chan, P. K.; and Piatetsky-Shapiro, G. 1993. Systems for knowledge discovery in databases. *IEEE Transactions on Knowledge and Data Engineering* 5(6):903–913. Special Issue on Learning and Discovery in Knowledge-Based Databases.

Matsuo, Y. and Kanehisa, M. 1993. An approach to systematic detection of protein structural motifs. *Computer Applications in the Biosciences* 9(2):153–159.

Mays, E.; Dionne, R.; and Weida, R. 1991. K-Rep system overview. *SIGART Bulletin* 2(3):93–97.

McGregor, J. J. 1982. Backtrack search algorithms and the maximal common subgraph problem. *Software—Practice and Experience* 12:23–34.

Meyers, K. and Konolige, K. 1992. Reasoning with analogical representations. In Nebel, B.; Rich, C.; and Swartout, W., editors, *KR92 : Principles of Knowledge Representation and Reasoning.* Morgan Kaufmann. 189–200.

Meyers, K. L. 1994. Hybrid reasoning using universal attachment. *Artificial Intelligence* 67(2):329–375.

Michalski, R. S. and Stepp, R. E. 1983a. Automated construction of classifications: Conceptual clustering versus numerical taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5(4):219–243.

Michalski, R. S. and Stepp, R. E. 1983b. Learning from observation: conceptual clustering. In Michalski, R.; Carbonell, J.; and Mitchell, T., editors, *Machine Learning*. Tioga. 331–363.

Mineau, G. W. and Godin, R. 1994. Automatic structuring of knowledge bases by conceptual clustering. *IEEE Transactions on Knowledge and Data Engineering*. To appear.

Minsky, M. 1975. A framework for representing knowledge. In Winston, P. H., editor, *The Psychology of Computer Vision*. McGraw-Hill.

Muggleton, S.; Srinivasan, A.; and Bain, M. 1992. Compression, significance and accuracy. In Sleeman, D. and Edwards, P., editors, *Machine Learning: Proceedings of the Ninth International Conference (ML92)*. Morgan Kaufmann. 338–347.

Muggleton, S. and Feng, C. 1992. Efficient induction of logic programs. In Muggleton, S., editor, *Inductive Logic Programming*. Academic Press. 281–298.

Muggleton, S. 1992. Inductive logic programming. In Muggleton, S., editor, *Inductive Logic Programming*. Academic Press. 3–27.

Muggleton, S. 1993. Inductive logic programming: derivations, successes and shortcomings. In Bradzil, P. B., editor, *ECML-93 : Lecture Notes in Artificial Intelligence 667*. Springer Verlag. 21–38.

Nagy, M.; Kozics, S.; Veszpremi, T.; and Bruck, P. 1988. Substructure search on very large files using tree-structured databases. In Warr, W. A., editor, *Chemical Structures: The International Language of Chemistry*. Springer–Verlag. 127–130.

Napoli, A 1992. Subsumption and classification-based reasoning in object-based representations. In Neumann, B., editor, *Proc. ECAI–92: Tenth European Conference on Artificial Intelligence*. John Wiley and Sons. 425–429.

Nebel, B. 1988. Computational complexity of terminological reasoning in BACK. *Artificial Intelligence* 34(3):371–383.

Nebel, B. 1990. *Reasoning and Revision in Hybrid Representation Systems*. Springer–Verlag.

Nebel, B. 1991. Terminological cycles: Semantics and computational properties. In Sowa, J. F., editor, *Principles of Semantic Networks*. Morgan–Kaufmann. 331–361.

Nordhausen, B. and Langley, P. 1990. An integrated approach to empirical discovery. In Shrager, J. and Langley, P., editors, *Computational Models of Scientific Discovery and Theory Formation*. Morgan Kaufmann. chapter 4, 97–128.

Nussinov, R. and Wolfson, H. J. 1991. Efficient detection of three–dimensional structural motifs in biological macromolecules by computer vision techniques. *Proc. Natl. Acad. Sci. USA* 88:10495–10499.

Okada, T. and Wipke, W. T. 1989. CLUSMOL: A system for the conceptual clustering of molecules. *Tetrahedron Computer Methodology* 2(4):249–264.

Okada, T. 1993. Similarity and analogy based on discrimination net. In Warr, W., editor, *Chemical Structures 2*. Springer-Verlag. 389–398.

Onizuka, K.; Ishikawa, M.; Wong, S. T. C.; and Asai, K. 1993. A multi–level description scheme of protein conformation. In Hunter, L.; Searls, D.; and Shavlik, J., editors, *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology*. AAAI Press. 301–309.

Padgham, L. and Lambrix, P. 1994. A framework of part-of hierarchies in terminological logics. In *KR94 : Principles of Knowledge Representation and Reasoning*.

Parsons, T. J. 1989. Conceptual clustering in relational structures: An application in the domain of vision. In *EWSL89, Proceedings of the Fourth European Working Session on Learning.* 163–177.

Patel-Schneider, P. F.; Brachman, R.; and Levesque, H. 1984. ARGON: Knowledge representation meets information retrieval. In *Proc. IEEE AIA-84.* 280–286.

Pearlman, R. S. 1993. Three-dimensional structures: How do we generate them and what can we do with them? *Chemical Design Automation News* 3–15.

Pentland, A. 1986. Perceptual organization and the representation of natural form. *Artificial Intelligence* 28:293–331.

Piatetsky-Shapiro, G. 1991. Discovery, analysis, and presentation of strong rules. In Piatetsky-Shapiro, G. and Frawley, W., editors, *Knowledge Discovery in Databases.* AAAI Press. chapter 13, 229–248.

Piatetsky-Shapiro, G. 1992. Probabilistic data dependencies. In Zytkow, J. M., editor, *Proceedings of the ML92 Workshop on Machine Discovery*, Aberdeen, Scotland. 11–17.

Pitt, L. and Reinke, R. E. 1988. Criteria for polynomial-time (conceptual) clustering. *Machine Learning* 2:371–396.

Plotkin, G. D. 1971. A note on inductive generalisation. *Machine Intelligence* 6:101–124.

Ponder, J. W. and Richards, F. M. 1987. Tertiary templates for proteins: Use of packing criteria in the enumeration of allowed sequences for different structural classes. *Journal of Molecular Biology* 193:775–791.

Pongor, S.; Skerl, V.; Cserzo, M.; Hatsagi, Z.; Simon, G.; and Bevilacqua, V. 1993. The SBASE domain library: a collection of annotated protein segments. *Protein Engineering* 6(4):391–395.

Prestrelski, S. J.; Williams Jr., A. L.; and Liebman, M. N. 1992. Generation of a substructure library for the description and classification of protein secondary structure. i. overview of the methods and results. *PROTEINS: Structure, Function, and Genetics* 14:430–439.

Quinlan, J. R. 1990. Learning logical definitions from relations. *Machine Learning* 5(3):239–266.

Randic, M.; Brissey, G. M.; and Wilkins, C. L. 1981. Computer perception of topological symmetry via canonical numbering of atoms. *J. Chem. Inf. Comp. Sci.* 21:52–59.

Reiter, R. 1985. On reasoning by default. In Brachman, R. J. and Levesque, H. J., editors, *Readings in knowledge representation.* Morgan Kaufmann.

Richards, F. M. and Kundrot, C. E. 1988. Identification of structural motifs from protein coordinate data: Secondary structure and first-level supersecondary structure. *PROTEINS: Structure, Function, and Genetics* 3:71–84.

Riddell, F. G. 1980. *The Conformational Analysis of Heterocyclic Compounds.* Academic Press.

Ring, C. S.; Kneller, D. G.; Langridge, R.; and Cohen, F. E. 1992. Taxonomy and conformational analysis of loops in proteins. *Journal of Molecular Biology* 224:685–699.

Rooman, M. J.; Wodak, S. J.; and Thornton, J. M. 1989. Amino acid sequence templates derived from recurrent turn motifs in proteins: critical evaluation of their predictive power. *Protein Engineering* 3(1):23–27.

Rooman, M. J.; Rodriguez, J.; and Wodak, S. J. 1990a. Automatic definition of recurrent local structure motifs in proteins. *Journal of Molecular Biology* 213:327–336.

Rooman, M. J.; Rodriguez, J.; and Wodak, S. J. 1990b. Relations between protein sequence and structure and their significance. *Journal of Molecular Biology* 213:337–350.

Rooman, M. J. and Wodak, S. J. 1988. Identification of predictive sequence motifs limited by protein database size. *Nature* 335:45–49.

Rooman, M. J. and Wodak, S. J. 1991. Weak correlation between predictive power of individual sequence patterns and overall prediction accuracy in proteins. *PROTEINS: Structure, Function, and Genetics* 9:69–78.

Rouveirol, C. 1994. Flattening and saturation: two representation changes for generalization. *Machine Learning* 14(2):219–232.

Royer, V. and Quantz, J. 1993. Axiomatization of description logics. In *Proceedings of the IJCAI Workshop on Object-Based Representation Systems.* 330–332.

Sali, A. and Blundell, T. 1990. Definition of general topological equivalence in protein structures. *Journal of Molecular Biology* 212:403–428.

Salton, G. and McGill, M. J. 1983. *Introduction to modern information retrieval.* McGraw-Hill.

Salton, G. and Wong, A. 1978. Generation and search of clustered files. *ACM Trans. Database Systems* 3(4):321–346.

Sammut, C. and Banerji, R. B. 1986. Learning concepts by asking questions. In Michalski, R.; Carbonell, J.; and Mitchell, T., editors, *Machine Learning: An Artificial Intelligence Approach*, volume II. Morgan Kaufmann. 167–191.

Schmidt-Schauß, M. and Smolka, G. 1991. Attributive concept descriptions with complements. *Artificial Intelligence* 48:1–26.

Schmolze, J. G. and Lipkis, T. A. 1983. Classification in the KL-ONE knowledge representation system. In *IJCAI-83.* 330–332.

Schulz, G. E. and Schirmer, R. H. 1979. *Principles of Protein Structure.* Springer–Verlag.

Segen, J. 1990. Graph clustering and model learning by data compression. In *Proceedings of the Seventh International Conference on Machine Learning.* 93–101.

Selman, B. 1988. Analogues. Technical Report CSRI-47, University of Toronto.

Sengupta, K. and Boyer, K. 1993. Information theoretic clustering of large structural modelbases. In *Int. Conf. Computer Vision and Pattern Recognition.* 174–179.

Shah, A. V. and Dolata, D. P. 1993. Conformations of large cycloalkanes: cyclooctadecane, cyclononadecane and cycloicosane. *Journal of Computer-Aided Molecular Design* 7:103–124.

Shapiro, L. G. and Haralick, R. M. 1982. Organization of relational models for scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 4(6):595–602.

Shapiro, L. G. 1980. A structural model of shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2(2):111–126.

Shapiro, E. Y. 1983. *Algorithmic program debugging.* The MIT Press.

Sloman, A. 1971. Interactions between philosophy and artificial intelligence: the role of intuition and non–logical reasoning in intelligence. *Artificial Intelligence* 2:209–225.

Smith, E. and Medin, D. 1981. *Categories and Concepts.* Harvard University Press.

Smith, R. F. and Smith, T. F. 1990. Automatic generation of primary sequence patterns from sets of related protein sequences. *Proc. Natl. Acad. Sci.* 87:118–122.

Solomons, T. W. Graham 1988. *Organic Chemistry.* John Wiley & Sons.

Sowa, J. F. 1984. *Conceptual Structures: Information Processing in Mind and Machine.* Addison–Wesley.

Sowa, J. F., editor. *Principles of Semantic Networks: Explorations in the Representation of Knowledge.* Morgan–Kaufmann.

Stepp, R. E. and Michalski, R. S. 1986. Conceptual clustering: inventing goal-oriented descriptions of structured objects. In Michalski, R.; Carbonell, J.; and Mitchell, T., editors, *Machine Learning: An Artificial Intelligence Approach*, volume II. Morgan Kaufmann. 471–498.

Stepp, R. E. 1987. Machine learning from structured objects. In *Proceedings of the Fourth International Workshop on Machine Learning.* Morgan Kaufmann. 353–363.

Sternberg, M. J. E. and Islam, S. A. 1990. Local protein sequence similarity does not imply a structural relationship. *Protein Engineering* 4(2):125–131.

Tanimoto, S. 1976. An iconic/symbolic data structuring scheme. In Chen, C. H., editor, *Pattern Recognition and Artificial Intelligence.* Academic Press. 452–471.

Tanimoto, S. 1980. Image data structures. In Tanimoto, S. and Klinger, A., editors, *Structured Computer Vision.* Academic Press. chapter 2, 31–55.

Taylor, W. R. 1986. Identification of protein sequence homology by consensus template alignment. *Journal of Molecular Biology* 188:233–258.

Tennent, R. D. 1991. *Semantics of Programming Languages.* Prentice–Hall.

Thompson, K.; Langley, P.; and Iba, W. 1991. Using background knowledge in concept formation. In *Proceedings of the Eighth International Workshop on Machine Learning.* Morgan Kaufmann.

Thompson, K. and Langley, P. 1989. Organization and retrieval of composite concepts. In *Proceedings of the Case–Based Reasoning Workshop.* Morgan Kaufmann. 329–333.

Thompson, K. and Langley, P. 1991. Concept formation in structured domains. In Fisher, D. H.; Pazzani, M.; and Langley, P., editors, *Concept Formation: Knowledge and Experience in Unsupervised Learning.* Morgan Kaufmann. 127–161.

Thornton, J. M. and Gardner, S. P. 1989. Protein motifs and data-base searching. *Trends in Biochemical Science* 14:300–304.

Thornton, J. M. and Swindells, M. B. 1993. Modelling of related protein structures. In Diamond, R., editor, *Molecular Structures in Biology.* Oxford University Press. chapter 4.

Tversky, A. 1977. Features of similarity. *Psychological Review* 84:327–352.

Unger, R.; Harel, D.; Wherland, S.; and Sussman, J. 1989. A 3D building blocks approach to analyzing and predicting structure of proteins. *Proteins* 5:355–373.

van Benthem, J. 1991. *The Logic of Time.* Kluwer.

Vosselman, G. 1992. *Relational matching: Lecture Notes in Computer Science 628.* Springer-Verlag.

Wasserman, K. and Lebowitz, M. 1983. Representing complex physical objects. *Cognition and Brain Theory* 6(3):333–352.

Wasserman, K. 1985. *Unifying representation and generalization: Understanding hierarchically structured objects.* Ph.D. Dissertation, Department of Computer Science, Columbia University.

Weida, R. and Litman, D. 1992. Terminological reasoning with constraint networks and an application to plan recognition. In Nebel, B.; Rich, C.; and Swartout, W., editors, *KR92 : Principles of Knowledge Representation and Reasoning.* Morgan Kaufmann. 282–293.

Wilcox, C. S. and Levinson, R. A. 1986. A self–organized knowledge base for recall, design, and discovery in organic chemistry. In Pierce, T. H. and Hohne, B. A., editors, *Artificial Intelligence Applications in Chemistry.* American Chemical Society. 209–230.

Willett, P. 1990. Algorithms for the calculation of similarity in chemical structure databases. In Johnson, M. A. and Maggiora, G. M., editors, *Concepts and applications of molecular similarity.* John Wiley & Sons. chapter 3.

Willett, P. 1991. *Three-Dimensional Chemical Structure Handling.* John Wiley & Sons.

Winston, M.; Chaffin, R.; and Herrmann, D. 1987. A taxonomy of part-whole relations. *Cognitive Science* 11(4):417–444.

Winston, P. H. 1975. Learning structural descriptions from examples. In Winston, P. H., editor, *The Psychology of Computer Vision.* McGraw-Hill.

Winston, P. H. 1980. Learning and reasoning by analogy. *Communications of the ACM* 23(12):689–703.

Winston, P. H. 1984. *Artificial Intelligence.* Addison–Wesley.

Wipke, W. T. and Hahn, M. A. 1986. Analogy and intelligence in model building. In Pierce, T. H. and Hohne, B. A., editors, *Artificial Intelligence Applications in Chemistry.* American Chemical Society. 136–146.

Wirth, K. 1986. Coding of relational descriptions of molecular structures. *J. Chem. Inf. Comp. Sci.* 26:242–249.

Wogulis, J. and Langley, P. 1989. Improving efficiency by learning intermediate concepts. In *Proc. IJCAI–89.* Morgan–Kaufmann. 657–662.

Wolfson, H. J. and Lamdan, Y. 1992. Transformation invariant indexing. In Mundy, J. L. and Zisserman, A., editors, *Geometric Invariance in Computer Vision.* The MIT Press. chapter 17.

Woods, W. and Schmolze, J. 1992. The KL-ONE family. *Computers Math. Applic.* 23(6).

Woods, W. 1985. What's in a link: Foundations for semantic networks. In Brachman, R. J. and Levesque, H. J., editors, *Readings in knowledge representation.* Morgan Kaufmann.

Woods, W. 1991. Understanding subsumption and taxonomy: A framework for progress. In Sowa, J. F., editor, *Principles of Semantic Networks.* Morgan–Kaufmann. 45–94.

Yoshida, K.; Motoda, H.; and Indurkya, N. 1993. Unifying learning methods by colored digraphs. In Tecuci, G.; Kedar, S.; and Kodratoff, Y., editors, *Proceedings of the IJCAI-93 Workshop on Machine Learning and Knowledge Acquisition.* 253–269.

Zembowicz, R. and Zytkow, J. M. 1992. Discovery of regularities in databases. In Zytkow, J. M., editor, *Proceedings of the ML92 Workshop on Machine Discovery*, Aberdeen, Scotland. 18–27.

Zhang, X.; Fetrow, J. S.; Rennie, W. A.; Waltz, D. L.; and Berg, G. 1993. Automated derivation of substructures yields novel structural building blocks in globular proteins. In Hunter, L.; Searls, D.; and Shavlik, J., editors, *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology.* AAAI Press. 438–446.

Zytkow, J. 1993. Cognitive autonomy in machine discovery. *Machine Learning* 12:7–16. Introduction to Special Issue on Machine Discovery.