Representing and Learning Temporal Relationships among Experimental Variables

Vanathi Gopalakrishnan
Department of Computer Science
University of Pittsburgh
vanathi@cs.pitt.edu

Bruce G. Buchanan
Intelligent Systems Laboratory
University of Pittsburgh
buchanan@cs.pitt.edu

Abstract

We describe the necessity to capture temporal information in scientific experiment design for analysis by machine learning algorithms that can learn useful temporal patterns among experimental variables. We have identified three types of temporal information, namely, duration, rate of change, and sequence of application of laboratory operators that are useful to learn from experimental data. Our motivation stems from study of experiment design in the domain of macromolecular crystallography[18]. In this paper, we identify the challenges posed both by the domain as well as the temporal information on machine learning learning programs, and describe work in progress. We outline our method of temporal specialization for inducing temporal relations between experimental variables, and illustrate with an example from the domain.

1 Introduction

In experimental science, changing experimentally controllable parameters determines the outcomes of experiments. Temporal relations are essential for describing experiments when the order and duration of steps matter to the outcome, but they complicate learning common features of successful experiments. Temporal patterns are difficult to infer in such domains, as the interactions among the many input parameters may be inherently nonlinear. Although machine learning algorithms have learned generalizations of a concept from experimental data, learning temporal relationships has posed a major challenge to machine learning algorithms, due to computational and complexity issues involved with the representation and reasoning of continuously changing variables.

We have been studying experiment design in macromolecular crystallography [14, 11, 10], and recognize the need for capturing temporal information that can be analyzed by machine learning algorithms capable of learning useful temporal relationships from such experimental data. We have identified three types of temporal information, namely, duration, rate of change, and sequence of application of laboratory operators that are useful to learn from experimental data.

This paper describes work in progress. We describe the motivation for learning these temporal relationships among experimental variables, particularly in the domain of macromolecular crystallography. We will then specify how we represent each type of temporal information, so that we can employ machine learning techniques with appropriate modifications. Then, we describe our method of temporal specialization for learning these relationships among experimental variables. Finally, we will present examples to illustrate our learning method, and comment on the current status and future directions of this work.

2 Motivation and Background

Time plays an important role in the outcome of most multi-step scientific experiments. From a computational point of view, time has been a challenge for representation and reasoning purposes. The motivation for research described in this paper stems from three very different factors that we will describe in the following subsections. The first subsection describes the domain of macromolecular crystallography experiment design, where subtle changes in parameters such as temperature, can cause major changes in experimental outcome. The challenge this domain provides for machine learning arises from the non-existence of a complete theoretical model underlying the process of crystallization. Thus, experimentation is guided largely by trial-and−error with a few heuristics from several partial models provided by expert crystallographers. We hope that machine learning can help augment our understanding of this domain by learning from past examples of failed and successful experiments concerning different classes of macromolecules (protein, DNA, protein-DNA or DNA−DNA complexes).

The second subsection describes the need to learn operator application sequence and how this problem has been studied in a more general context by past researchers.

The third subsection describes the necessity for machine learning programs to be able to learn from structured attributes. In particular, we describe the symbolic inductive learning program, RL[5, 20], and why we need to augment it with capabilities to learn from examples containing typed attributes.

2.1 Experiment Design in Macromolecular Crystallography

The goal of an experiment is to grow an X-ray diffractible crystal of a macromolecule, such as protein or DNA. The key to growing a good crystal is to start with an initial solution of protein and precipitants, and then vary the experimental parameters such as temperature, pH, and concentration to first reach a state of supersaturation, and then to *rapidly* lower the level of supersaturation until the saturation of chemicals is just right for a crystal to grow. A good quality crystal is needed for subsequent analyses to determine the three-dimensional structure of a macromolecule. The quality of crystal grown is indicated mostly by the resolution limit of the diffraction pattern, which is largely a function of size, shape and regularity.

Early experimentation in crystallography demonstrated the impact of varying temperature on the outcome of crystallizations. We quote an example of successful experimentation to obtain crystals of E.Coli Phosphatase in the 1970's[12].

> Single crystals of E. Coli Alkaline Phosphatase .... Crystallization was carried out by slowly warming a solution of protein, 20 to 25 mg per ml from 4° to 25°. The solution was 56% saturated in $(NH_4)_2SO_4$, 0.05 M Tris, 0.01 M $MgCl_2$, pH 8.0.

The rate at which temperature was varied, the actual values of temperature, and how it was varied (.e.g., by increasing or decreasing) clearly impacted experimental outcome. Note that the type and concentration of protein, precipitant ($(NH_4)_2SO_4$), buffer (Tris) and salt ($MgCl_2$) together with the pH play an important role in determining the outcome of experimentation. Thus, apart from being able to decide which values of these experimental variables to use, crystallographers need to decide at what temperature to set-up the initial solution, and whether and how to vary the temperature of the set-up thereafter. More examples and relevant explanations can be found in [3, 24, 16, 18].

An experiment in the domain of macromolecular crystallography therefore, consists of putting together an initial solution of several chemical and bio-chemical compounds and varying physical and chemical controllable parameters such as the temperature and pH, until the level of supersaturation is sufficient for the macromolecule to nucleate and grow. Some interesting temporal features of this process include:

1. If it takes too long to cross the boundary between saturation and supersaturation, the protein might form an amorphous precipitate, but not a crystal.

2. If you cross the boundary too rapidly, a crystal might nucleate, but with an undesirable value of resolution limit of diffraction(i.e. bad crystal).

3. The rate at which the physical chemistry takes place is determined to a large extent on the values chosen for the controllable parameters of an experiment. Controllables, such as temperature, that can vary with time, can cause subtle changes in the values of other parameters, thereby affecting the outcome.

This domain is particularly challenging in that no complete model of the crystallization process exists. We have chosen to work with a few important controllable parameters and a partial model of the process, as identified by domain experts[23].

2.2 Learning Laboratory Operator Sequence

Learning a sequence of laboratory operators can easily be viewed more generally as learning the sequence of operators that perform actions in the real world (that is, steps in a plan). Since STRIPS[8] and ABSTRIPS[25], considerably more work in AI has focussed on representing plans then on learning them, and most of the temporal relations have been simple before and after relations. Some researchers have studied the problem of learning plan schemata that are structured knowledge chunks similar to scipts, frames and macroops[26, 6, 4]. We focus on the representation of time in experimental science with respect to efficiency of learning generalizations of temporal relationships between any two laboratory operators.

Let us consider what it takes to make ice from water. First, we need to lower the temperature of water to below $0°C$ (sub-zero conditions) for ice crystals to nucleate, and then we hold the temperature at around zero degrees for ice to form and remain in solid phase. Let us now describe these two operations as *decrease-temp* (< 0) and *hold−temp* (0). It is clear that the sequence in which these two operations are performed will directly impact the outcome, i.e. whether ice forms or not. If we simply hold the temperature at zero and never go below it, nucleation will not occur, and hence, ice will not form. The domain of macromolecular crystallography is even more complicated due to the number of parameters (> 25) that can possibly influence nucleation and growth of crystal. Also, macromolecules are much more sensitive to variations in the solution conditions, as they can change structure fairly easily.

2.2.1 Learning from Structured Attributes

Structural learning, also known as relational learning, has been a topic of interest to machine learning researchers for a long time. Structural concept learners use a more expressive first-order predicate calculus representation for their hypotheses, as opposed to simple attribute-value pairs of features. The complexity in developing such learning algorithms stems from the expressive power of first-order representations to include existentially quantified variables in the hypotheses. It has been shown that even matching a hypothesis to an example is NP-complete[13].

Most symbolic induction systems, such as decision-tree learners and rule learners, learn generalizations of a concept from training examples represented as attribute-value vectors in a propositional language. Let us consider a simple example to show why relations are hard to express and learn in purely propositional learners. Suppose we have two operations concerning the same operand that are performed at different start times:

Throw(Ball, Time_1) and Catch(Ball, Time_2) $\Rightarrow$ Game

We are essentially dealing with existential variables, and the learning program needs to be able to learn relationships between the two start times, such as NOT-EQUAL(Time_1, Time_2), LESS$-$THAN(Time_1, Time_2) or GREATER-THAN(Time_1, Time_2).

Purely propositional rule learners can represent the above example in the form of (attribute value) pairs, such as first attribute name = ThrowBallAt, value = Time_1, second attribute name = CatchBallAt, value = Time_2. The learning is performed on the values of the individual attributes in conjunction with appropriate values for other attributes, but relationships between values are not expressible.

We could augment these learners to learn from any set of attributes expressed as n-ary predicates. The computational complexity involved in learning such relational terms needs to be handled by structuring terms and values hierarchically. For example, we could have introductory predicates such as ANY-RELATION or ANY-RATE at the root of the specialization hierarchies. This will not only enable us to learn most general concept descriptions, but will also immensely reduce the search space. We hope to study the effectiveness of augmenting a symbolic inductive rule-learner to accept structured attribute representations during generalization, by introducing a structured attribute such as a laboratory operator that is dependent on time.

FOIL[22], KATE[17], STRUCT[27] are some early successful efforts in inducing concepts from structural data. FOIL and STRUCT represent relations as Horn clauses. FOIL constructs Horn clause programs from numerous examples. Manago's KATE is a decision-tree algorithm that uses an object-oriented frame language equivalent to first-order predicate calculus. STRUCT borrows ideas from FOIL, KATE and other programs such as INDUCE[7] and FRINGE[19], for representation of concept, examples and feature construction. Pazzani's FOCL is another algorithm that learns structural descriptions.

FOIL uses logic representation schemes and uses Inductive Logic Programming (ILP) techniques; while KATE uses structured representation schemes such as frames and objects. KATE preserves the qualities of an ID3[21] symbolic decision-tree inductive learning algorithm: viz. efficiency (preference for smaller tree), model driven search and hill-climbing. KATE also uses the value hierarchies introduced first in RL[9] (described below), for specializing structured slot values. The types of algorithms that perform structural learning can also be classified according to learning method: *separate-and-conquer* also known as covering algorithms, (INDUCE[7] was an early algorithm that used this method to generate disjuncts, FOIL is a more recent example); *divide-and-conquer* or decision tree inducers (e.g., KATE); and *adaptive feature construction* (e.g., FRINGE).

A recent topic of interest in relational learning includes the use of regression to learn quantitative expressions that predict numeric variable (i.e. continuous class), such as learning from time series data. Systems such as FORS (Karalic 1995) and SRT (Kramer 1996) integrate statistical methods of regression into the ILP framework. Other systems such as CART (Breiman et al 1984), RETIS (Karalic 1992), and M5 (Quinlan 1992) use regression tree methods in a decision-tree framework. Systems such as FOCL, FORS, RL, SRT, KATE use background knowledge (initial domain theory) to drive hypotheses generation and search.

2.3 Inducing rules from data with RL

The RL program[5, 20] views inductive learning as a knowledge-based problem solving activity that could be implemented in the heuristic search paradigm. It was first used to learn rules for predicting mass spectra of complex organic molecules, and has been generalized and extended since then in several ways, which are mentioned below. Its method is primarily to search a space of possible rules by successive specialization, guided by data in the training set and by prior knowledge. It learns a disjunctive set of weighted conjunctive rules. One of the distinguishing features of a knowledge-based approach to learning is that RL can use a partial theory of the domain as prior knowledge with which to construct its rules. Prior knowledge may include the legal semantics and syntax for the rules plus additional bias about plausible or implausible relationships that are well agreed upon by the domain experts.

RL learns rules of the form P_1,...,$P_k \Rightarrow C$, where the left-hand side is a conjunction of propositions P_1,...,P_k and C is the concept predicate. The rules can be described also as IF condition THEN concept-class. The condition part of the IF-THEN rule consists of a conjunction of values for one

or more attributes that comprise the input parameters of the training examples presented to RL. RL starts with the most general rule with no conjuncts on the left-hand side, and performs a heuristic search of the space of specializations by adding conjuncts to candidate rules. The set of rules learned constitute a disjunction of conjunctive conditions that describe the concept.

One feature of RL that makes it a flexible learner is its ability to use background knowledge to constrain the search for rules. Background knowledge is incorporated as RL's Partial Domain Model, and includes such information as constraints on numeric valued attributes, such as range of value and step size, desirable properties of rules being learned, as well as available ISA hierarchies of domain attributes. This latter property of being able to constrain the search space of hypotheses rules by using ISA hierarchies is very useful for our purposes of representing and dealing with temporal information.

3 Temporal Relationships and Representation

We have identified three basic types of temporal information that can be useful for experimentally controlling parameters:

1. Duration of laboratory operators that are time-dependent. Example: temperature effects, such as increasing, decreasing or holding constant temperature over a particular time range.

2. Rate of change of laboratory operators. Example: Rate of change of temperature over some time interval could be either rapid or slow, depending on whether the temperature variation takes place over a period of minutes or days. Rate of change of temperature could also be described as linear, exponential or sudden spikes that go up or down.

3. Temporal relationships between two or more laboratory operators that indicate sequence of application of operators. Example, temperature was increased *before* changing the pH of the chemical solution.

3.1 Representation of temporal data

Laboratory operators represent the changes made to experimental controllable parameters over time. Time is represented as time intervals due to the flexibility and ease with which we can reason with this representation[1]. We will define a special attribute of type *time interval* as consisting of two numeric values start time and duration. Each of these attributes will have values that are organized hierarchically. Thus,

TimeInterval(I) $\models$ Start(I, Value), Duration(I, Value)
Laboratory-Operator(I) $\models$
Change (controllable-parameter, rate, I) ||
Hold (controllable-parameter, value, I)

Duration of laboratory operators is simply treated as a numeric variable in the context of rule learning. In this domain, we do not feel the need to reason about durations, other than to learn a range of possible values for application of a particular laboratory operator in any situation.

Rate of change of laboratory operators: We are particularly interested in characterizing rate as slope at the mid-pt over some time interval. We incorporate a more qualitative description of rate of change of a laboratory operator, as part of the description of the operator itself. For example, laboratory operator UP-LINEAR(Temperature, Slowly, I) describes a linear increase in the experimental variable temperature slowly over time interval I.

Temporal relationships between any two laboratory operators are described based on Allen's Temporal Interval hierarchy[1] $--$ example of the hierarchy is shown in Figure 1. Also shown, are samples of the specialization hierarchies of laboratory-operators, rate of change, and duration as a numeric value.

3.2 Data

For the purposes of testing our approach to learning temporal relationships, we have built a simulator based on the simple model of crystallization mentioned in a subsequent section. The simulator will act as both a data generator and the evaluation function. We input the simulator values for experimental parameters, both givens such as protein name and molecular weight and controllables such as pH, salt concentration, protein concentration, and the simulator will give us the observables $--$ classified as good crystal, clear (failure) or amorphous precipitate. We are in the process of introducing the temporal effects into the simulator. The simulator will be given inputs which are time-dependent such as temperature in the following format:

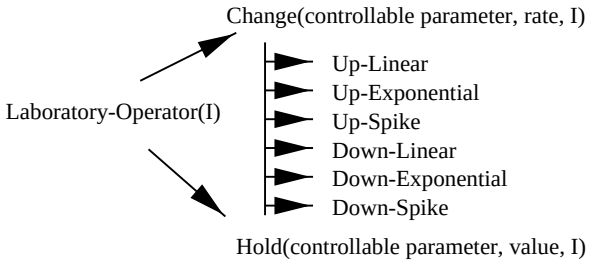
Start Temp, End Temp, Rate, Start time, Duration.

For the purposes of simplifying the description of laboratory operators in our initial computational experiments, we will assume that the controllable parameter is mentioned as part of operator (e.g. change-temperature).

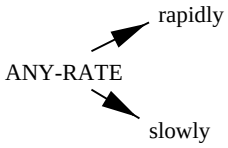
4 Temporal Specialization

Our hypothesis is that we can learn interesting temporal relationships between experimental variables that can help in the design of scientific experiments. In order to test this hypothesis, we are currently working on an implementation of a prototype system that will be able to process the following types of input:

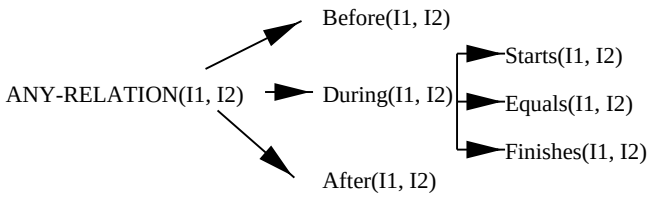
Temporal Events as Laboratory Operators



Rate of Change of Controllable Parameter



Temporal Interval Relationships



Duration of a Temporal Interval
(same as a numeric-valued attribute)

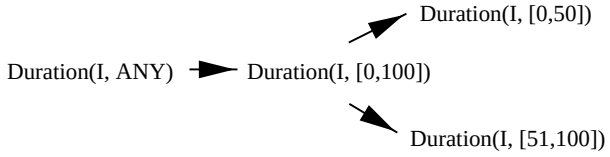


Figure 1. Representative Samples of Temporal Specialization Hierarchies. Arrows show direction in which specialization proceeds.

Example of cases:
Protein = *Alpha Globulase* (Hypothetical Protein),
Molecular Weight = *10000* (kilo Daltons(kD)),
Precipitating-Agent = *PEG* (PolyEthylene Glycol),
Laboratory-Operator(rate, (start−time, duration)) = *up−linear−temperature(slowly, (1, 2000))*,
Laboratory-Operator (rate, (start−time, duration)) = *down−spike−temperature(rapidly, (2001, 20))*,
Laboratory-Operator (value, (start−time, duration)) = *hold−temperature (35, (2022, 3000))*,
Outcome-Class = *good−crystal*.

This example states that protein Alpha Globulase, molecular weight 10000 kD was successfully crystallized by first raising the temperature of the set-up slowly, then suddenly lowering the temperature for a short time and raising it and holding it at $35°C$.

Initially, we will try to learn generalizations of patterns that involve rates and durations. Later, we will include actual values of the temperature to the input data, so that we can learn patterns of values. For the sake of readability, we have decided to break down the problem into rates and values. Temperature values in this particular domain are very few, and hence can be treated as discrete values for learning purposes (as opposed to continuous numeric values, which add to computational complexity).

We will use an augmented version of the RL algorithm to learn temporal relationships from structured attributes. We will use the same generate-and−test paradigm to generate candidate hypotheses (rules) and test them against the training examples to retain statistically significant rules for further specialization. The most general values of each type of attribute will be used in the initial hypotheses rules. In addition, we will use temporal specialization hierarchies for determining how to integrate or refine a temporal clause within a candidate rule.

We will start with the most general rule, say NIL → good-crystal. Temporal specialization consists of one-step specializations that can be one of the following operations:

1. **Temporal Clause Integration**

   1. Adding a temporal clause containing a laboratory-operator with DURATION(ANY), AND
   2. If there is more than one temporal clause with laboratory-operators, adding a the most general temporal relation i.e. ANY−RELATION between the two time intervals.

2. **Temporal Clause Refinement**

   3. Specializing an existing temporal relation using the specialization hierarchy, such as in Figure 1.

3. **Temporal Interval Adjustment**

   4. Specializing the duration of an interval of a temporal relation, using a specialization hierarchy that is formed dynamically from the training examples, using certain heuristics to handle overlapping intervals for positive and negative training examples[15].

4.1 Partial Domain Model

4.1.1 Representation of an Experiment

An experiment in this domain is modeled as shown below:

$$Experiment(x) \models Solution(x), Events(x)$$
$$Solution(x) \models Substance(s, x), Solvent(x)$$
$$Substance(s, x) \models Protein(s) || DNA(s) || Complex(s)$$
$$Solvent(x) \models Chemical(c, x), Solvent(x)$$
$$Chemical(c, x) \models PptAgent(c) || pH(c) || buffer(c) || salt(c)$$
$$Events(x) \models Laboperator(i, x), Events(x)$$

An experiment is modeled as ($\models$) a solution containing protein and other chemicals in initial conditions followed by one or more temporal events. These events are described by laboratory-operators applied over a particular time interval.

4.1.2 Heuristics Used

The combinatorics involved in computing all possible temporal relationships between any two laboratory operators is huge. Several heuristics will be incorporated as part of the method of temporal specialization to effectively minimize the number of hypotheses generated. All of them will be used to decide how to specialize a temporal clause. Some of them are stated below:

1. Specialize the value of duration of a laboratory-operator only after we know that it applies in a particular context. For example, temporal specialization step 3 (interval adjustment) will be performed only if we find that a rule of the form: *protein AND laboratory-operator(I) AND duration(I, ANY)* $\Rightarrow$ *good-crystal* holds for a large number of training examples.

2. Interval relationships between any two laboratory-operators A & B are introduced into a hypothesis rule only after A & B occur in a rule. This heuristic simply states that rules containing single laboratory-operators will need to be specialized further before temporal relationships are added as part of the hypothesis rule.

3. Rule schemas that describe commonly applied sequences of laboratory operators will be used first to determine if they apply with respect to the training data. They will be transformed, if necessary. Examples of rule schemas include operator sequences that describe tricks that are well-known to domain experts, such as, *sharp spikes in temperature followed by changes in the pH can cause a crystal to nucleate*.

4. Quantities and constraints used for numeric values of variables in the simulator model will be used to constrain hypotheses (rule) generation during learning.

The above are only a few of the heuristics that will be used during learning. The Partial Domain Model used by RL will contain several other definitions and constraints that form the background domain knowledge.

5 An Example

We will now illustrate our proposed method for learning temporal relations with an example from the domain of macromolecular crystallography. Let us suppose that we have done some pre-processing and we present the following three cases to the learning program (Note that typically, we need in the order of hundreds or thousands of cases for training purposes).

Alpha Globulase, PEG, up-linear−temperature(slowly, (1, 2000)), down-spike−temperature(rapidly, (2001, 20))
$\Rightarrow$ good crystal.
Alpha Globulase, PEG, up-linear−temperature(rapidly, (1, 20)), down-spike−temperature(rapidly, (2001, 20))
$\Rightarrow$ good crystal.
Alpha Globulase, PEG, hold-temperature(4, (1, 2000))
$\Rightarrow$ bad crystal.

We will show below how a candidate hypothesis is formed and specialized. It is to be noted that at each step, each rule can be specialized in several different ways, and each new rule becomes a candidate hypothesis for the next round of matching against the training examples. For the purpose of this example, we will assume that the controllable parameter being manipulated experimentally is described as part of the laboratory operator description (e.g. change-temp for changing the temperature). We will assume that when a laboratory operator is described in a rule for the first time, another clause is added that says that the duration of the time interval over which the operator applies could be anything (e.g. Duration(I_1, ANY)), and will be specialized later. For the sake of brevity of illustration, we will ignore the clauses that contain duration (one instance is shown).

NIL $\Rightarrow$ *good-crystal*
Experiment(x) $\Rightarrow$ *good-crystal*
Solution(x), Events(x) $\Rightarrow$ *good-crystal*
Substance(s,x), Solvent(x), Events(x) $\Rightarrow$ *good-crystal*
Protein(s), Solvent(x), Events(x) $\Rightarrow$ *good-crystal*

Alpha Globulase, Solvent(x), Events(x) ⇒ good-crystal
Alpha Globulase, Chemical(c,x), Events(x) ⇒ good-crystal
Alpha Globulase, PptAgent(c), Events(x) ⇒ good-crystal
Alpha Globulase, PEG, Events(x) ⇒ good-crystal
Alpha Globulase, PEG, Laboperator(i_1, x), Events(x) ⇒ good-crystal
Alpha Globulase, PEG, change-temp(ANY-RATE, i_1) Events(x), duration(i_1, ANY) ⇒ good-crystal

⋮ ⋮

Alpha Globulase, PEG, up-linear-temp(ANY-RATE, i_1), change-temp(ANY-RATE, i_2), ANY-RELATION(i_1, i_2) ⇒ good-crystal

⋮ ⋮

Alpha Globulase, PEG, up-linear-temp(ANY-RATE, i_1), down-spike-temp(rapidly, i_2), Meets(i_1, i_2) ⇒ good-crystal

Each proposition or predicate added is specialized down its type or value hierarchy: Substance is specialized to protein, and then to Alpha Globulase, and solvent to PEG. The temporal relationships are specialized according to their hierarchies as shown in Figure 1, and in the manner stated in the method of temporal specialization. We have omitted the two clauses containing Duration(i_1, ANY) and Duration(i_2, ANY) for sake of brevity. ANY-RELATION is an introductory predicate that specializes down the temporal interval hierarchy to describe that time interval i_1 is just before i_2.

6 Current Status and Future Work

Our representation for each of the temporal relationships identified is descriptive and can be learned. Efficient general representation for modeling rate of change exists as storing position and velocity. Our representation of a laboratory operation performed over a time interval can easily be mapped into this more general representation. We have simply chosen to separate out certain descriptive elements such as linear or exponential rates of change and incorporate them into the description of a laboratory operation that can be changed or held constant. It is useful to view the operators this way, since we can build neat specialization hierarchies as we have shown.

Currently, the prototype under development is being implemented in the C programming language and generates candidate hypotheses for testing just as outlined in the algorithm. The use of types in deciding how to specialize various aspects of a structured attribute works well in biasing the learning program to choose only "productive" one-step specializations to be performed on a rule. Rule schemas are still being implemented and will certainly aid in reducing the number of "unproductive" hypotheses being generated. The simulator model needs to be modified to take into account temperature dependencies on solubility of a macromolecule. The existing version of the simulator yields several time series data that describe the rate of change of various experimental parameters. Even though we have described rate qualitatively for machine learning purposes, it would be worthwhile in the future to augment our machine learning algorithm with some statistical techniques such as linear regression, in order to be able to learn to quantify concepts such as rate of change. Previously cited systems such as FORS and SRT are some recent work in learning continuous class based on regression techniques. Most of the temperature dependencies introduced into the simulator will be linear. Augmenting our learning algorithm with regression techniques will enable us to validate the learned expressions and their predictive power against the model used by the simulator.

The matching technique is still primitive and is expensive. In order to eliminate/reduce matching, we are looking to use breadth-first marker passing technique as discussed in Aronis[2]. Some other ideas for efficient matching include the formation of queries in SQL or database language.

We will compare our ideas and implementations to those used in inductive logic programming systems such as FOIL and FORS; as well as frame and object-oriented systems such as KATE. Preliminary comparsions based on type of learning model, use of background knowledge and type of representation schemes used for data and domain knowledge have been stated in Section 2.

In a very global sense, we are trying to deal with the classic frame problem in AI. Our assumption is that we can express scientific experiment design in the form of explicit time dependent laboratory operators, which in turn hold explicit information regarding “things that do not change". Events involving application of laboratory operators indicate that there is a whole change taking place in the physical chemistry of the experimental set-up and obviously, there is insufficient domain knowledge to infer the consequences. The partial domain theory used for developing the simulator actually captures some of the simultaneous changes taking place in the form of the driving equations. The actual interactions yielding the state of an experiment at any point in time, would constitute several differential equations, which need to be solved simultaneously. So, even if we did have a perfect theory, it would not be possible to computationally determine the state due to the dimensionality. In our approach, we assume that explicitly captured events have consequences as stated by the observations, and we try to learn generalizations of types of events that are likely to result in (un)favorable consequences.

We have not focussed on different scales of time such as hours, days, weeks, years, . . . explicitly using specialization hierarchies. We are currently assuming that a uniform representation of time units will suffice. In this paper, we do have not referenced the extensive work done in qualitative

reasoning research. This is mainly because the model used for the simulator is quantitative, even though some of the features used for learning are expressed qualitatively in our initial prototype.

We have focussed our initial efforts on the identification and representation of temporal relationships that are useful in guiding scientific experimentation. We are in the process of completing the implementation of our initial prototype TIPS (Temporal Induction of Parameter Sequence) that will enable us to critically evaluate the computational aspects of our approach.

7 Acknowledgments

This work has been supported in part by the NIH (NCRR) (RR10447-02] , NSF (IRI-9412549) and the W.M. Keck Center for Advanced Training in Computational Biology (921277). We would like to thank Dr. John Aronis at the Intelligent Systems Laboratory for his discussions and useful insights regarding this problem. We are also grateful to Prof. John M. Rosenberg, Department of Biological Sciences, for his enthusiasm and support.

References

[1] J. Allen. Maintaining knowledge about temporal intervals. *Communications. ACM*, 26:832–843, 1983.

[2] J. M. Aronis and F. Provost. Increasing the efficiency of data mining algorithms with breadth-first marker propagation. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, pages 119–122. AAAI Press, 1997.

[3] T. Blundell and B. Johnson. Protein crystallography (molecular biology series). pages 59–82, June 1976.

[4] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32(3), 1987.

[5] S. Clearwater and F. Provost. RL4: A tool for knowledge-based induction. In *Proceedings of the Second International IEEE Conference on Tools for Artificial Intelligence*, pages 24–30. IEEE CS. Press, 1990.

[6] G. F. DeJong and R. J. Mooney. Induction of decision trees. *Machine Learning*, 1(2):145–176, 1986.

[7] T. G. Dietterich and R. S. Michalski. Inductive learning of structural descriptions. *Artificial Intelligence*, 16(3):257–294, 1981.

[8] R. Fikes and N. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[9] L. M. Fu and B. G. Buchanan. Learning intermediate concepts in constructing a hierarchical knowledge base. In *Proceedings of the Ninth IJCAI*, pages 659–666. Morgan Kaufmann, 1985.

[10] V. Gopalakrishnan. Learning plan sequence from temporal data. Technical report, University of Pittsburgh, 1995. PhD. Thesis Proposal.

[11] V. Gopalakrishnan, D. Hennessy, B. G. Buchanan, D. Subramanian, P. A. Wilcosz, K. Chandrasekhar, and J. M. Rosenberg. Preliminary tests of machine learning tools for the analysis of biological macromolecular crystallization data. Technical report, University of Pittsburgh, 1994. ISL-94-17.

[12] A. W. Hanson, M. L. Applebury, J. Coleman, and H. Wykoff. X-ray studies on single crystals of escherichia coli alkaline phosphatase. *J. Biol. Chem.*, 245:4975–4977, 1970.

[13] D. Haussler. Learning conjunctive concepts in structural domains. *Machine Learning*, 4:7–40, 1989.

[14] D. Hennessy, V. Gopalakrishnan, B. G. Buchanan, and D. Subramanian. Induction of rules for biological macromolecule crystallization. In *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, pages 179–187, Aug 1994.

[15] Y. Lee. *Learning a robust set of rules*. PhD thesis, Computer Science Department, University of Pittsburgh, 1995.

[16] W. Littke. Experimental set-ups for protein crystal growth. In R. Geigé, A. Ducruix, J. C. Fontecilla-Camps, R. S. Feigelson, R. Kern, and A. McPherson, editors, *Crystal Growth of Biological Macromolecules*. North Holland, 1987. Proceedings of the Second International Conference on Protein Crystal Growth, Bischenberg, Strasbourg, France. A FEBS Advanced Lecture Course.

[17] M. Manago. Knowledge intensive induction. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 151–155. Morgan Kaufmann, Jun 1989.

[18] A. McPherson. Current approaches to macromolecular crystallization. *European Journal of Biochemistry*, 189:1–23, 1990.

[19] G. Pagallo and D. Haussler. Feature discovery in empirical learning. Technical report, University of California at Santa Cruz, Santa Cruz, CA, 1990. UCSC-CRL-90-27.

[20] F. Provost, B. Buchanan, S. Clearwater, Y. Lee, and B. Leng. Machine learning in the service of exploratory science and engineering: a case study of the RL induction program. Technical report, University of Pittsburgh, 1993. ISL-93-6.

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

[22] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[23] J. M. Rosenberg. Personal communication.

[24] F. Rosenberger and E. J. Meehan. Control of nucleation and growth in protein crystal growth. In R. Geigé, A. Ducruix, J. C. Fontecilla-Camps, R. S. Feigelson, R. Kern, and A. McPherson, editors, *Crystal Growth of Biological Macromolecules*. North Holland, 1987. Proceedings of the Second International Conference on Protein Crystal Growth, Bischenberg, Strasbourg, France. A FEBS Advanced Lecture Course.

[25] E. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5:115–135, 1974.

[26] J. W. Shavlik. An empirical analysis of ebl approaches for learning plan schemata. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 183–187. Morgan Kaufmann, Jun 1989.

[27] L. Watanabe and L. Rendell. Learning structural decision trees from examples. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 770–776. Morgan Kaufmann, 1991.