Inductive Concept Learning in an Equational Logic System

PH.D RESEARCH PROPOSAL
DEPARTMENT OF COMPUTER SCIENCE AND STATISTICS
UNIVERSITY OF RHODE ISLAND

CANDIDATE:
Arthur A. McDonald

SUPERVISORY COMMITTEE:
Lutz Hamel, Supervisor
Committee Member 1
Committee Member 2
Committee Member 3

September 24, 2012

Contents

1 Title of the Study

Inductive Concept Learning in an Equational Logic System

2 Statement of the Problem

For this dissertation, we propose to research and develop inductive concept learning algorithms using equational logic as the representation language. A concept learning algorithm essentially has three parts: representation, search and evaluation [1]. Since the representation language of equational logic has been thoroughly established and formalized, we intend to spend the majority of this research on the search and evaluation of our algorithms.

3 Justification for and Significance of the Study

In this section we present some preliminary topics that must be covered in order to understand the significance of this research study. We provide brief overviews of concept learning, previous research on Inductive Logic Programming (ILP) and concept learning in first-order predicate logic, and an introduction to equational logic and how it has been used as a programming language.

3.1 Concept Learning

Concept learning can be defined as the discovery of a set of rules that describe a concept or phenomenon based on a set of examples. In addition, these rules should also be able to predict the results of unseen examples. As a subfield of machine learning, a concept learner is a computational system or computer program that automatically learns a concept or set of concepts when given training examples and background knowledge as input.

A classic concept learning task in the machine learning literature is the task of learning whether to play or not play tennis based on certain environmental conditions. For this task, a concept learner would be provided input training examples similar to those shown in Table 1, and a set of decision rules would be developed so that all of the positive input examples can be derived from the rules, and none of the negative examples can be derived.

Outlook	Temp	Humidity	Wind	Play-Tennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes

Table 1: Play Tennis training examples.

Figure 1 shows the decision rules that describe the concept based on the examples previously given. Here, $\vee$ and $\wedge$ are the standard symbols for logical OR and AND.

if Outlook=Overcast then PlayTennis=Yes $\vee$
if Outlook=Rain $\wedge$ Wind=Weak then PlayTennis=Yes

Figure 1: Play Tennis decision rules.

The process of inducing these decision rules from examples, or inductive concept learning, can be viewed as a search problem. The learning system must search the domain of all possible rules, called the Hypothesis Space or H, to find a set of rules that describe the concept.

3.2 Inductive Logic Programming

The term Inductive Logic Programming (ILP) was first introduced by Muggleton [2]. This early work led to significant developments in concept learning algorithms that use first-order logic as the representation language. Systems such as Quinlan's FOIL [3] [4], Muggleton's Progol [5], Flach's Escher [6] and Shapiro's MIS [7] and others have been successfully applied to problems in carcinogenesis predictions [8], protein structure prediction [9], medical diagnoses [10], natural language processing [11], recovering software specifications [12].

3.3 Equational Logic

Equational logic is a subset of first-order logic. It deals with logic sentences where the only logical operator is the binary predicate for equality, typically written as $=$ or the standard equals sign [13]. It is the logic of substituting equals for equals using algebras as models and term rewriting as the operational semantics [14]. Typed equational logic is both sound and complete.

Written in standard equational logic, the training examples from Table 1 would be represented as an equational theory. Figure 2 contains the Play Tennis training examples as represented as an equational theory in standard equational logic syntax.[1]

As a brief introduction to equational logic syntax, we now provide a line-by-line description of this theory. A more formal definition of equational theories follows.

- Line 1: Define a new equational theory (or object) TENNIS-EXAMPLES.
- Line 2: Define the sorts (another term for data types), that will be used in the theory.
- Lines 3-6: Define the attributes that will be used in the equations, and associates them with a sort. These ops of arity 0 are constants.
- Line 7: Define the playtennis operator, which takes four parameters of the sorts Outlook, Temp, Humidity and Wind and produces a boolean result of true or false. This is an operator of arity 4.
- Lines 9-15: These are the training example equations, with the first four being positive examples and the next three being negative examples.[2]

[1] Standard equational logic as defined by systems based on Goguen's OBJ family of languages [15].

[2] It is interesting to note that, in equational logic, negative examples can be rewritten as positive examples by writing them as (X =/= true) = true, rather than X = false.

```
1  obj TENNIS-EXAMPLES is
2    sorts Outlook Temp Humidity Wind .
3    ops overcast rain sunny : -> Outlook .
4    ops hot mild cool : -> Temp .
5    ops high normal : -> Humidity .
6    ops weak strong : -> Wind .
7    op playtennis : Outlook Temp Humidity Wind -> Bool .
8
9    eq playtennis(overcast, hot, high, weak)  = true .
10   eq playtennis(rain, mild, high, weak) = true .
11   eq playtennis(rain, cool, normal, weak) = true .
12   eq playtennis(overcast, cool, normal, strong) = true .
13   eq (playtennis(sunny, hot, high, weak) =/= true) = true .
14   eq (playtennis(sunny, hot, high, strong) =/= true) = true .
15   eq (playtennis(rain, cool, normal, strong) =/= true) = true .
16 endo
```

Figure 2: Play Tennis concept training examples.

- Line 16: End of theory definition.

Definition 1 *An equational signature is a pair (S, Σ), where S is a set of sorts and Σ is a $(S^* \times S)$-sorted set of operation names. We usually abbreviate (S, Σ) as Σ.*

Relating Definition 1 back to the TENNIS-EXAMPLES theory, the equational signature is the sorts and ops, or lines 2-7. The equational theory, or Σ-theory, is the equational theory plus the set of equations (lines 9-15). Formally,

Definition 2 *A Σ-theory is a pair (Σ, E) where Σ is an equational signature and E is a set of Σ-equations. Each equation $e \in E$ has the form $(\forall X) l = r$, where X is a set of variables and $l, r \in T_\Sigma(X)$ are terms over the set Σ and X. If l and r contain no variables, i.e. $X = \emptyset$, then we say the equation is **ground**.*

Training examples are always provided as ground equations. A valid inductive learning system in equational logic would take these examples and produce a set of rules that covers all of the positive examples and none of the negative examples. As stated before, these rules would also be good at predicting unseen positive and negative examples. Figure 3 is a valid equational hypothesis that correctly describes the TENNIS concept based on the training examples. Here we can see that, based on the training examples provided, we were able to produce a target concept that includes just two rules (equations) that accurately cover all of the positive examples and none of the negative ones. One other point to mention is the use of variables in this theory. On lines 8-10 we define several variables with their sorts that will be used in the equations. The variables can be substituted for any op that is of the same sort as the variable.

```
1  obj TENNIS is
2    sorts Outlook Temp Humidity Wind .
3    ops overcast rain sunny : -> Outlook .
4    ops hot mild cool : -> Temp .
5    ops high normal : -> Humidity .
6    ops weak strong : -> Wind .
7    op playtennis : Outlook Temp Humidity Wind -> Bool .
8    var TempVar : Temp .
9    var HumidityVar : Humidity .
10   var WindVar : Wind .
11
12   eq playtennis(overcast, TempVar, HumidityVar, WindVar) = true .
13   eq playtennis(rain, TempVar, HumidityVar, weak) = true .
14 endo
```

Figure 3: Play Tennis concept definition.

The main goal of this thesis research is to develop learning algorithms in equational logic that can automatically produce the description of a concept, as in the tennis concept in Figure 3, given an equational theory, such as Figure 2.

3.4 Justification

While the problem of concept learning in systems using first-order predicate logic and the traditional attribute-value representation languages has been well researched, the use of equational logic as the representation language is still an open problem. To our knowledge, Hamel [16] [14] and Shen [17] is the only work that has successfully developed inductive concept learning algorithms in equational logic systems. Both of these algorithms use genetic programming techniques to search the hypothesis space for candidate theories. The limitations of these systems include:

- Implementation of only a subset of equational logic. Conditional equations are not supported.
- Some solutions produced are technically correct, yet presented in a way that is algebraically incorrect. This was a result of the way the underlying rewrite engine considers equations in order.
- These systems used significant memory resources and computational time.

Because of these limitations, we propose that algorithms for inductive concept learning in equational logic is an area where new research could be beneficial. Many of the successful ILP applications that have been previously implemented can be represented in equational logic and the concepts can be learned with our new system. We also propose that research that leads to new algorithms for inductive learning in equational logic may solve some of the open problems in Hamel and Shen's work. Namely, the handling of conditional equations (see Figure 4 for an example of a conditional equation in equational logic) as well as the memory and speed limitations.

```
ceq boolean(M) = true if M =/= 0 .
```

Figure 4: A conditional equation in equational logic

Equational logic is an excellent representation language for inductive concept learning for several reasons. First, it has a strong type system, which applies a constraint on the hypothesis search space as sentences with invalid types can be eliminated from consideration. Also, since equational logic uses rewriting to evaluate terms, it makes it a faster and more efficient logic system than first-order logic that uses more complicated unification algorithms [17].

4 Methodology or Procedures

This thesis will consist of a research study, design of an inductive concept learning algorithm or possibly several algorithms, program implementation of the algorithm, and testing and analysis of the system.

4.1 Research Study

The research study will be directed towards understanding the mathematical theory of equational logic and the computational semantics of equational logic programming. Another objective will be an attempt to fully understand the current implementations of inductive concept learning algorithms in first-order logic.

4.2 Algorithm Design and Implementation

Using the knowledge gained in the research study, we will then design an algorithm or algorithms for inductive concept learning in equational logic. The algorithm design will then be implemented as part of a current equational logic programming system, such as OBJ3, Maude or BOBJ. We have already conducted some research into the design of these systems, including source code walk-throughs to learn how the systems are implemented.

4.2.1 Approaches

There are two approaches that we will consider when designing these inductive concept learning algorithms. The first approach is to adapt an already existing algorithm for inductive learning in first-order predicate logic for use in equational logic. Broadly speaking, these systems use one of two methods for inducing predicate clauses from examples: top-down or bottom-up induction.

MIS [7] and FOIL [3] [4] use a top-down algorithm for searching the hypothesis space. With top-down induction, an initial general rule is successively specialized. This is basically a generate-then-test approach. Other top-down algorithms that we will explore include ILS [18], AQ15 [19], CILS [20], and CN2 [21].

Progol [5], Golem [22], and a hybrid of the two systems called ProGolem [23] use a bottom-up algorithm for rule creation. Bottom-up induction uses a specific training example as a starting rule and successively generalizes the rule to cover more training examples. Bottom-up induction is an example-driven approach.

The second approach is to develop completely new algorithms for equational logic based on current research on algorithms for searching hypothesis spaces. One promising method is to use stochastic search as explored in [24] and [25]. The limitation of performing a locally optimal search in standard top-down/bottom-up inductive learning algorithms may be overcome by using a stochastic, or random, hypothesis space search strategy. In [24], the authors use a method for solving large scale (possibly NP complete) optimization problems called simulated annealing.

A second stochastic search method uses rapid random restarts (RRR) as described in [26] and [27]. These algorithms perform short searches starting from an initial random clause and proceeding through the clause lattice via the neighbors. These short searches are repeated a certain maximum number of times and for a maximum duration. This approach has been shown to be superior to exhaustive searches in many domains.

Another method for stochastic search in [28] and [29], called stochastic clause selection, uses subsampling. This restricts the search space by randomly selecting a set of training examples to generate clauses. The hypothesis that the authors propose is that a concept learning system that uses a sampling method constructs, in lesser computational time, a concept description that is no worse in predictive accuracy than the same algorithm that does not use sampling. A stochastic sampling technique is also developed in the STILL system [30]. The results of this work seem promising, and we will examine possible stochastic sampling methods in equtional logic concept learning.

4.3 Testing and Evaluation

The final part of this research will involve testing the algorithm against some classic learning tasks in the machine learning literature, as well as some complex, real-world examples. The results of these tests will be analyzed for their effectiveness and efficiency. Some classic concept learning tasks include the summation of natural numbers, even/odd numbers, stack data structure and operators, train direction, and the play tennis task described earlier in this proposal. Real-world data sets that we will use for testing include Haberman's survival data[3], car buying evaluation data[4], income-census data[5], and credit approval data[6]. We will also evaluate the algorithms in their ability to classify unseen examples.

[3] 5-year survival rate post breast-cancer surgery.
[4] Buy or not buy a car based on certain attributes.
[5] Income $>=$ \$50k based on census demographics.
[6] Approve or deny credit based on economic indicators.

5 Resources Required

There are several resources required to complete this project: research articles and texts, data sets of test concepts to be learned, and a personal computer running the necessary software. To date, 64 research articles and 3 texts have been obtained from online sources and the URI Library. Additional journal and conference proceedings can be retreived from online sources such as CiteSeer^x scientific literature digital library, the ACM digital library and the IEEE Xplore digital library. Additional print material can be retreived from the URI Library.

We will use the datasets from Shen and Hamel's original work on concept learning in equational logic for initial testing and evaluation. Other datasets will be obtained from the University of California, Irvine's Machine Learning Repository [31].

The software required to implement the algorithms for this study are the Linux operating system, Oracle's Java 1.6 development kit, the GNU C++ compiler (g++), and the OBJ3, BOBJ and Maude equational logic programming systems. This is all open source software and currently installed on the personal computer that will be used for this study.

6 Literature Cited

[1] Pedro Morais Domingos Delgado, *A unified approach to concept learning*, Ph.D. thesis, University of California at Irvine, Irvine, CA, USA, 1997, UMI Order No. GAX97-28076.

[2] Stephen Muggleton, "Inductive logic programming," *New Generation Comput.*, vol. 8, no. 4, pp. 295–318, 1991.

[3] J. R. Quinlan, "Learning logical definitions from relations," *MACHINE LEARNING*, vol. 5, pp. 239266, 1990.

[4] J. Ross Quinlan and R. Mike Cameron-Jones, "FOIL: a midterm report," in *ECML*, 1993, pp. 3–20.

[5] S.H. Muggleton, "Inverse entailment and progol," *New Generation Computing*, vol. 13, pp. 245286, 1995.

[6] P. A. Flach, Flach Giraud-Carrier, and J. W. Lloyd, "Strongly typed inductive concept learning," in *Proceedings of the 8th International Conference on Inductive Logic Programming, volume 1446 of Lecture Notes in Artificial Intelligence*. 1998, p. 185194, Springer-Verlag.

[7] Ehud Y. Shapiro, "Inductive inference of theories from facts," in *Computational Logic - Essays in Honor of Alan Robinson*, 1991, pp. 199–254.

[8] A. Srinivasan, R.D. King, S. H. Muggleton, and M.J.E. Sternberg, "Carcinogenesis predictions using ilp," 1997.

[9] S.H. Muggleton, R.D. King, and M.J.E. Sternberg, "Protein secondary structure prediction using logic-based machine learning," *Protein Engineering*, vol. 5, no. 7, pp. 647–657, 1992.

[10] Hayato Ohwada, Makiko Daidoji, Shiroaki Shirato, and Fumio Mizoguchi, "Learning first-order rules from image applied to glaucoma diagnosis," in *PRICAI98: Topics in Artificial Intelligence*, Hing-Yan Lee and Hiroshi Motoda, Eds., vol. 1531 of *Lecture Notes in Computer Science*, pp. 494–505. Springer Berlin / Heidelberg, 1998, 10.1007/BFb0095295.

[11] Raymond Mooney and Mary Elaine Califf, "Induction of first-order decision lists: Results on learning the past tense of english verbs," *Journal of Artificial Intelligence Research*, vol. 3, pp. 1–24, 1995.

[12] William W. Cohen, "Recovering software specifications with inductive logic programming," in *Proc. Twelfth National Confeernce on Artificial Intelligence, The*. 1994, pp. 142–148, Press.

[13] Don Pigozzi, "Equational logic and equational theories of algebras," Technical Report 85, Purdue University, Mar. 1975.

[14] Lutz Hamel, "Evolutionary search in inductive equational logic programming," in *Proceedings of the Congress on Evolutionary Computation*, Canberra, Australia, 2003, pp. 2426–2434, IEEE.

[15] Joseph A. Goguen and Grant Malcolm, *Algebraic Semantics of Imperative Programs*, MIT Press, Cambridge, MA, USA, 1996.

[16] Lutz H. Hamel, "Breeding algebraic structures - an evolutionary approach to inductive equational logic programming," in *GECCO*, 2002, pp. 748–755.

[17] Chi Shen, "Evolutionary concept learning in equational logic," M.S. thesis, University of Rhode Island, 2006.

[18] Mehmet R. Tolun, Hayri Sever, Mahmut Uludag, and Saleh M. Abu-Soud, "Ila-2: an inductive learning algorithm for knowledge discovery," *Cybernetics and Systems*, vol. 30, no. 7, pp. 609–628, 1999.

[19] R. Michalski, I. Mozetic, J. Hong, and N. Lavrac, "The AQ15 inductive learning system: an overview and experiments," in *Proceedings of IMAL 1986*, Orsay, 1986, Universit de Paris-Sud.

[20] Simon Anthony and Alan M. Frisch, "Cautious induction in inductive logic programming," in *ILP*, 1997, pp. 45–60.

[21] Peter Clark and Tim Niblett, "The CN2 induction algorithm," in *MACHINE LEARNING*, 1989, p. 261283.

[22] Stephen Muggleton and Cao Feng, "Efficient induction of logic programs," in *New Generation Computing*. 1990, Academic Press.

[23] Stephen Muggleton, Jos Carlos Almeida Santos, and Alireza Tamaddoni-Nezhad, "ProGolem: a system based on relative minimal generalisation," in *ILP*, 2009, pp. 131–148.

[24] Matevz Kovacic, Nada Lavrac, Marko Grobelnik, Darko Zupanic, and Dunja Mladenic, "Stochastic search in inductive logic programming," in *ECAI*, 1992, pp. 444–445.

[25] Uros Pompe, Matevz Kovacic, and Igor Kononenko, "Sfoil: Stochastic approach to inductive logic programming," in *In Proceedings of the Fourth International Workshop on Inductive Logic Programming (ILP-94), GMD-Studien Nr. 237*. 1993, pp. 189–192, GMD Press.

[26] Filip Zelezny, Ashwin Srinivasan, and David Page, "Lattice-search runtime distributions may be heavy-tailed," in *In Proceedings of the 12th International Conference on Inductive Logic Programming*. 2002, pp. 333–345, Springer Verlag.

[27] Filip elezn, Ashwin Srinivasan, and David Page, "A monte carlo study of randomised restarted search in ilp," in *Inductive Logic Programming*, Rui Camacho, Ross King, and Ashwin Srinivasan, Eds., vol. 3194 of *Lecture Notes in Computer Science*, pp. 251–266. Springer Berlin / Heidelberg, 2004.

[28] Ashwin Srinivasan, "A study of two sampling methods for analyzing large datasets with ilp," *Data Min. Knowl. Discov.*, vol. 3, no. 1, pp. 95–123, Mar. 1999.

[29] Ashwin Srinivasan, "A study of two probabilistic methods for searching large spaces with ilp," 1999.

[30] Michèle Sebag and Céline Rouveirol, "Resource-bounded relational reasoning: Induction and deduction through stochastic matching," *Mach. Learn.*, vol. 38, no. 1-2, pp. 41–62, Jan. 2000.

[31] Irvine University of California, "Machine learning repository," `http://archive.ics.uci.edu/ml/`.