A note on refinement operators

Tim Niblett *(tim@turing.com)*
The Turing Institute
36 North Hanover Street
Glasgow G1 2AD, Scotland

Abstract

The top down induction of logic programs is faced with the problem of ensuring that the search space includes all the desired hypotheses. The conventional way of of organizing the search space is via *refinement* of clauses. Within this context the existence of a well behaved refinement operator complete for Horn clause logic is desirable.

We show that there is no natural way in which a complete refinement operator can be defined which avoids the production of non-reduced clauses. Consideration is given to subsets of full Horn clause logic for which more efficient refinement operators can be constructed.

Category: Short Paper

1 Introduction

The Model Inference System (MIS) was developed by Shapiro ([6], [7]) drawing from theoretical studies by (among others) Gold ([1]), and Plotkin ([3], [5], [4]).

Plotkin studied the basis of generalization mechanisms within first order logic and characterized the notion of generalization as θ-subsumption. It is difficult to ensure that an inductive learning system based on subsumption is complete since subsumption is not well behaved. The contribution of MIS is that it provides a framework for a complete learning system, capable of learning any finitely axiomatizable clausal theory up to a given level of complexity. The search space is organized as a *refinement graph*, which is generated by a *refinement operator*. Shapiro claims in [7] that there is a most general refinement operator, complete for clausal logic. The purpose of this note is to demonstrate that the most general refinement operator described by Shapiro is not complete. In addition it is shown that there there is no "natural" refinement operator which does not introduce non-reduced refinements. This raises doubts about the efficiency of top down methods of inferring logic programs.

1.1 Organization of this note

The central result is straightforward to prove and is found in Section 5. The material before that is a review of supporting concepts. In Section 2 we review the concept of

subsumption and explain why a naive approach to top-down learning does not work. In Section 3 we outline the model inference problem in the framework created by Shapiro, and explain the role of refinement operators. Refinement operators and their properties are described in Section 4. The most general refinement operator is explained and the central result proven in Section 5. In Section 6 a complete refinement operator which introduces redundant clauses is described. Finally, in Section 7 we discuss the implications of the negative result we have uncovered.

Throughout this note we restrict attention to Horn clause logic, rather than the full clausal logic considered by Shapiro. It does not affect the principal result which holds *a fortiori* for full clausal logic, simplifies the presentation, and is consistent with practical uses of the MIS architecture.

2 Subsumption

A Horn clause $B_0 \leftarrow B_1, \ldots, B_n$ is equivalent to the set $\{B_0, \bar{B}_1, \ldots, \bar{B}_n\}$, where the B_i and $\bar{B}_i$ are atoms and negated atoms respectively, in what follows.

A substitution σ of terms for variables, is written $\{t_1/x_1, \ldots, t_n/x_n\}$. If C is a clause, then $C\sigma$ is the result of substituting the t_i in σ for the corresponding x_i.

We say that clause C θ-subsumes clause D if $C\sigma \subseteq D$ for some substitution σ. We write this as $C \leq D$. In what follows θ-subsumption is simply called subsumption.

The *size* of a clause C is the number of symbol occurrences in C (excluding punctuation) minus the number of distinct variables occurring in C. If every model of a set of sentences Σ is also a model of clause C we say that $\Sigma \models C$ We assume a complete derivation procedure for Horn clause logic and write $\Sigma \vdash C$ to show that C is derivable from Σ.

Given two clauses C and D we write $C \cong D$ when C subsumes D and D subsumes C. This is an equivalence relation. Two properties of subsumption are of interest to us. If $C \leq D$ then $C \vdash D$, although the reverse is not true. If $C \leq D$ then it is *not* true that $size(C) \leq size(D)$, hence subsumption cannot itself function as a refinement operator. Plotkin ([3]) gives examples of infinite strictly descending chains under subsumption. Plotkin also defines the notion of reduction. A clause C is *reduced* if there is no clause D such that $D \subset C$ and $D \cong C$. If two clauses are reduced then they are equivalent up to renaming of variables. We say that a substitution θ *decreases* clause C if $|C\theta| < |C|$, where $|C|$ is the number of elements in C interpreted as a set.

3 The model inference problem

The language $\mathcal{L}$ is the set of Horn clause sentences constructed from a finite number of function and predicate symbols. We consider two clauses C and D to be equivalent if they are alphabetic variants, that is if there exists a substitution $\tau = \{y_1/x_1, \ldots, y_n/x_n\}$ where the y_i are distinct and $C\tau = D$.

The sentences of $\mathcal{L}$ are of the form

$$b_0 \leftarrow b_1 \wedge \ldots \wedge b_n (n \geq 0)$$

```
T = {}
repeat
    read the next fact
    repeat
        while T is too strong do
            apply the contradiction backtracing algorithm,
            and remove from T the refuted clause.
        while T is too weak do
            add to T refinements of previously refuted hypotheses.
    until T is neither too strong nor too weak with respect
        to the facts read so far
forever
```

Figure 1: The MIS framework

where the b_i are atoms.

Two subsets of $\mathcal{L}$ are distinguished. $\mathcal{L}_o$ the observational language, and $\mathcal{L}_h$ the hypothesis language. We assume $\mathcal{L}_o \subset \mathcal{L}_h \subseteq L$. Both sublanguages are decidable.

The domain of enquiry is a model M of $\mathcal{L}$ and we assume an oracle for M which given $\alpha \in \mathcal{L}_o$ returns true iff α is true in M. The set of observational sentences true in M is $\mathcal{L}_o^M$.

A set of sentences $\Sigma \subset \mathcal{L}_h$ is an $\mathcal{L}_o$-complete axiomatization for M iff Σ is true in M and $\Sigma \vdash \mathcal{L}_o^M$ The Model Inference problem is to find a finite $\mathcal{L}_o$-complete axiomatization for M.

3.1 Admissibility requirements

$\mathcal{L}_o$ should contain enough information to refute any false theory. $\langle \mathcal{L}_o, \mathcal{L}_h \rangle$ is admissible if for every model M of $\mathcal{L}$ and every $\Sigma \subset \mathcal{L}_h$ the set $\{\alpha \in \mathcal{L}_o \mid \Sigma \vdash \alpha\} = \mathcal{L}_o^M$ implies that Σ is true in M.

Shapiro ([6]) shows that the observational language of ground literals of $\mathcal{L}$ with hypothesis language $\mathcal{L}$ itself is complete.

3.2 The incremental algorithm

The MIS framework is shown in Figure 1. It is intended to identify finitely axiomatizable theories in the limit. To ensure termination, it is important to note that all proof attempts are done modulo a total recursive function h which provides a limit on the complexity of the proof for any literal. In general the value of h should depend on the syntactic complexity of the goal to be proved.

The efficiency of this algorithm depends on two things, the contradiction backtracing algorithm (see [8]), and the structure of refinements. For completeness it is essential that the refinement generation process be complete. We now turn our attention to this.

4 Refinement operators

We first consider refinement operators. From a technical point of view a refinement operator is a specialization of subsumption, introduced to ensure that there are no infinite descending chains. In general we assume that there is a measure (*size*) of the structural complexity of clauses which maps to the natural numbers. We insist that for any natural number $n(\geq 0)$, the set of clauses of size n (modulo alphabetic variants) is finite.

Definition 1 *Clause D is a refinement of clause C if $C \vdash D$ and* size$(C) <$ size(D).

Definition 2 *A* refinement operator ρ *is a mapping from clauses to subsets of their refinements, such that for any $C \in \mathcal{L}$ and any $n > 0$ the set $\rho(C)(n)$ (the set $\rho(C)$ restricted to clauses of size $\leq n$) is recursively enumerable.*

A refinement operator ρ induces a partial order $\leq_\rho$ with the empty clause $\square$ as (unique) minimal element. We say that $C \leq_\rho D$ if there is a chain $C = C_0, \ldots C_n = D$ such that $C_{i+1} \in \rho(C_i), 0 \leq i \leq n$. We write $C <_\rho D$ if $C \leq_\rho D$ and $\neg(D \leq_\rho C)$.

For any clause C the set $\{D \in \mathcal{L} \mid C \leq_\rho D\}$ is written as $\rho^\star(C)$.

Definition 3 *A refinement operator ρ is complete for $\mathcal{L}$ if $\rho^\star(\square) = \mathcal{L}$.*

We now turn to a consideration of Shapiro's most general refinement operator ρ_o which was claimed in [6] to be complete for $\mathcal{L}$. A most general refinement operator is simply a refinement operator which is complete for $\mathcal{L}$.

5 Shapiro's most general operator

If X and Y are atoms and C a clause we say that X *is more general than* Y *with respect to* C if there is a substitution θ such that $X\theta = Y$ and $C\theta = C$. If $H \leftarrow B$ is a reduced clause then X *is a most general atom such that* $H \leftarrow B \wedge X$ *is reduced* if for any atom Y such that Y is more general than X with respect to $H \leftarrow B$, the clause $H \leftarrow B \wedge Y$ is not reduced.

Definition 4 *Let $C = H \leftarrow B$ be a reduced Horn clause. Then $D \in \rho_o(C)$ if exactly one of the following holds:*

1. *$D = C\theta$, where $\theta = \{V/W\}$ does not decrease C and both V and W occur in C.*
2. *$D = C\theta$, where $\theta = \{f(X_1, \ldots, X_n)/W\}$ does not decrease C, f is an n-ary $(n \geq 0)$ function symbol, W occurs in C and each $X_i, 1 \leq i \leq n$, is a distinct variable not occurring in C.*
3. *$D = H \leftarrow B \wedge P$, where P is a most general atom with respect to C for which $H \leftarrow B \wedge P$ is reduced.*

Note that for any clause C there are at most a finite number of applications of (1) and (2), but an infinite number of applications of (3) since in general there are infinitely many atoms most general with respect to C. Shapiro [7] shows that these refinements can be enumerated by *size*.

Theorem 1 *ρ_o is not complete for $\mathcal{L}$.*

Proof We exhibit a counterexample. Consider the reduced clause $C = a \leftarrow p(A,B,C) \wedge p(D,E,C) \wedge p(F,G,E) \wedge p(F,B,H)$. Assume that $C \in \rho_o^\star(\square)$. There must exist a reduced clause D such that $C \in \rho_o(D)$, obtained by an application of (1), (2) or (3) in the definition of ρ_0. Assume that C was obtained by (3). Removing any of the goals in the body produces a clause that is not reduced, counter to assumption. Clause C cannot have been obtained by (2) since it contains no function symbols. It cannot have been obtained by (1), since the replacement of a single occurrence of B, C, E or F leads to a reduced clause, counter to assumption. Since C cannot have been obtained by an application of (1), (2) or (3) it has no predecessor in ρ_0. ■

6 A most general refinement operator

The above proof shows that if a refinement operator adds at most one literal at a time to a reduced clause, then it is impossible to construct a most general refinement operator, none of whose refinements will be reducible. This is a potentially serious problem for top down inductive systems since the introduction of non-reduced clauses into a refinement operator will lead to greatly reduced efficiency.

For the sake of completeness we produce a most general refinement operator (ρ_1), based on Shapiro's, which is complete for $\mathcal{L}$.

Definition 5 *Let $C = H \leftarrow B$ be a Horn clause. Then $D \in \rho_1(C)$ if exactly one of the following holds:*

1. *$D = C\theta$, where $\theta = \{V/W\}$ does not decrease C and both V and W occur in C.*
2. *$D = C\theta$, where $\theta = \{f(X_1,\ldots,X_n)/W\}$ does not decrease C, f is an n-ary ($n \geq 0$) function symbol, W occurs in C and each $X_i, 1 \leq i \leq n$, is a distinct variable not occurring in C.*
3. *$D = H \leftarrow B \wedge P(X_1,\ldots,X_n)$, where P is a predicate symbol of arity n, and where the X_i are new variables not occurring in H or B.*

The proof that this refinement operator is complete is straightforward. Given a non-null clause $C = H \leftarrow B$ with B possibly empty we can show that C has a predecessor C' such that $C' \in \rho_1(C)$. If any literal $t \in B$ is of the form $p(X_1,\ldots,X_n)$ where the X_i are distinct variables not occurring elsewhere in C then $C' = C - \{t\}$ is a predecessor. Similarly if there is a term $t = f(X_1,\ldots,X_n)(n \geq 0)$ occurring one or more times in any place in any of the literals of C, and the X_i are distinct variables not occurring

elsewhere in C then $C' = C\theta$ where $\theta = \{f(X_1, \ldots, X_n)/W\}$ and W does not occur in C is a predecessor. Otherwise if X is any variable occurring more than once in C, then replacing a single occurrence of X by W, where W is a variable not occurring in C provides a predecessor C'. Finally, if $C = p(X_1, \ldots, X_n) \leftarrow$ where the X_i are distinct variables, then the predecessor is $\square$. As each application of of ρ_1 increases *size* by 1 and since there a finite number of clauses of any given size n it follows that ρ_1 is a most general refinement operator for $\mathcal{L}$.

7 Conclusion

It seems that any top-down structuring of the Hypothesis space for Horn clause programs will suffer from the problem of redundant hypotheses. This leaves us in the position of having to make a tradeoff between efficiency and completeness, or of having to focus on restrictions of full Horn clause logic if we want to learn top down.

It is an open questions as to which restrictions on full first order logic are compatible with complete non-redundant refinement operators. Shapiro ([7]) illustrates a non-redundant refinement operator for the class of context-free transformations. We mention one additional subset here.

An interesting semantic restriction is to ij-determinate clauses [2]. With the restriction to ij-determinate clauses it is possible to show given a non-reduced clause correct with respect to the intended interpretation that any extension of this clause is equivalent with respect to the intended interpretation to a non-reduced clause. This means that redundant (ie. non-reduced) clauses need never be generated, and hence there is a most general refinement operator for ij-determinate clauses which does not generate redundant clauses.

The result suggests that it may be possible to learn ij-determinate programs efficiently within the MIS framework.

Acknowledgements

My thanks to Wray Buntine who pointed out a flaw in the proof that ρ_0 is a most general refinement operator. This work was partially supported by the European Community Esprit Programme under contract P2154, MLT.

References

[1] E.M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[2] S. Muggleton and C. Feng. Efficient induction of logic programs. In S. Muggleton, editor, *Inductive Logic Programming*, pages 281–298. Academic Press, London, 1992.

[3] G.D. Plotkin. A note on inductive generalisation. In B. Meltzer and D. Michie, editors, *Machine Intelligence 5*, pages 153–163. Elsevier North-Holland, New York, 1970.

[4] G.D. Plotkin. *Automatic Methods of Inductive Inference*. PhD thesis, Edinburgh University, August 1971.

[5] G.D. Plotkin. A further note on inductive generalisation. In B. Meltzer and D. Michie, editors, *Machine Intelligence 6*, pages 101–124. Elsevier North-Holland, New York, 1971.

[6] E. Y. Shapiro. An algorithm that infers theories from facts. In *Proceedings of IJCAI-81*, pages 446–451. Kaufmann, Los Altos, CA, 1981.

[7] E.Y. Shapiro. Inductive inference of theories from facts. TR 192, Dept. Comp. Sc., Yale University, Connecticut, 1981.

[8] E.Y. Shapiro. *Algorithmic Program Debugging*. MIT Press, 1983.