A Most General Refinement Operator for Reduced Sentences

Patrick R. J. van der Laag
Department of Computer Science and Tinbergen Institute
Erasmus University of Rotterdam
P.O. Box 1738, 3000 DR Rotterdam, the Netherlands
patrick@cs.eur.nl

Abstract

Many learning systems use the space of logic formulas as the search space of hypotheses. To build efficient systems, the set of first order logic formulas can be reduced in many ways. Most systems restrict themselves to (subsets of) Horn clauses. In this paper we investigate the space of *reduced* first order sentences, which has the same expressive power as an arbitrary first order logic. Shapiro [6] has used the subset of reduced first order sentences to define a most general refinement operator. His operator is claimed to be complete, i.e., all reduced sentences are derivable from the empty sentence. In this article we will show that his operator is not complete and propose a new, complete refinement operator for reduced first order sentences.

1 Introduction.

1.1 Motivation.

In [6] Shapiro developed his notions of refinement and refinement operators. Shapiro uses refinement operators in an incremental inference algorithm to replace refuted, too strong hypotheses by logically weaker refinements of hypotheses. A most general refinement operator, i.e., a refinement operator for any first order language, is defined for so called *reduced* sentences.

Plotkin [4] has described an algorithm that, given a sentence p, computes a reduced sentence q. It appears that in every equivalence class under subsumption all reduced sentences are equal up to renaming variables. We therefore might speak of one unique representative for each equivalence class. Since restricting to reduced sentences does not diminish the expressive power, it is interesting for any algorithm that uses first order logic, whether we can restrict our search to reduced sentences.

Shapiro's Model Inference System is one of the first that inductively learns theories from facts. Other approaches include inverse resolution and different non-monotonic reasoning systems. Most of the more recent systems based on logic can use the presence of background knowledge and are restricted to Horn logic. Shapiro already pointed out that "The existence of a most general refinement operator is interesting mostly for theoretical reasons. In practice some information on the structure of the set of hypotheses to be inferred is usually known. In such a case we always prefer a less general refinement operator [...]." Shapiro's refinement still is a subject of interest, for example as an specialization operator in a default system [3].

In this paper we will first introduce the notations and notions of refinement and reduction. We will show that Shapiro's refinement operator for reduced sentences is not complete. Next, we will show that Laird's refinement operator, which is part of a more general framework of refinement, is complete for general first order sentences. We will investigate the properties of the search space of reduced sentences, and from these new insights we will develop a new, complete most general refinement operator for reduced sentences.

1.2 Notation.

Throughout this paper we will adopt some of the notations and definitions of Shapiro [6].

Given a given first order language $\mathcal{L}$ with finitely many predicate and function symbols, the sentences of $\mathcal{L}$ are of the form

$$\{L_1, L_2, \ldots, L_j\} \leftarrow \{L_{j+1}, L_{j+2}, \ldots, L_{j+k}\}, j, k \geq 0$$

where all L_i's are positive atoms, or

$$\{L_1, L_2, \ldots, L_{j+k}\}, j, k \geq 0$$

where the L_i's are positive $(1 \leq i \leq j)$ or negative $(j+1 \leq i \leq j+k)$ atoms. The interpretation of such a sentence is that the conjunction $L_{j+1}, \ldots, L_{j+k}$ implies the disjunction $L_1, \ldots, L_j$.

Sentences are denoted by p, q, p',... The conjunction of condition literals is denoted by A, A', A_1,... The disjunction of conclusion literals is denoted by B, B', B_i,... $\Box$ denotes the empty sentence, false in any model of $\mathcal{L}$. Predicate symbols are denoted by P, Q,..., literals by L, L_1,..., variables by x, y, x_1,..., function symbols by f, g, f_1,..., and constants by a, b,...

Throughout this paper sentences that are alphabetical variants are considered equal.

2 Reduction and refinement.

In this section we copy some definitions of Shapiro's paper that are necessary for our purposes.

2.1 Refinement.

Following Shapiro, we assume some structural complexity measure *size*, which is a function from sentences of $\mathcal{L}$ to natural numbers, with the property that for every $n > 0$ the set of sentences of size n is finite.

∘ Let $\mathcal{L}$ be a first order language, p and q sentences of $\mathcal{L}$. We say that q is a *refinement* of p if p implies q and $size(p) < size(q)$.

∘ A *refinement operator* ρ is a mapping from sentences of $\mathcal{L}$ to subsets of their refinements, such that for any $p \in \mathcal{L}$ and any $n > 0$ the set $\rho(p)(n)$, that is, the set $\rho(p)$ restricted to sentences of size $\leq n$, is computable.

A refinement operator over $\mathcal{L}$ induces a partial order over $\mathcal{L}$, with the empty sentence $\Box$ as a minimal element: a finite sequence of sentences $p = p_0, p_1, \ldots, p_n = q$ such that $p_{i+1} \in \rho(p_i)$ is called a *finite total ρ-chain.* If such a ρ-chain exists we denote it by $p \leq_\rho q$. For any sentence p, the set $\{q \in \mathcal{L} | p \leq_\rho q\}$ is denoted by $\rho^*(p)$. The set $\rho^*(\Box)$ is denoted by ρ^*.

From these definitions, it follows that a refinement operator is a specialization operator. Starting with the empty sentence $\Box$, by repeatedly specializing the resulting sentences, each sentence should be derivable.

∘ Let $S \subseteq \mathcal{L}$ be a set of sentences that include the empty sentence $\Box$. A refinement operator ρ is said to be *complete for S* if $\rho^* = S$.

∘ Let ρ and ρ' be two refinement operators over $\mathcal{L}$. We say that ρ' is *more general* than ρ if $\rho^* \subset \rho'^*$.

2.2 Reduction.

The notion of reduction is taken from Plotkin [4]. A motivation for looking at reduced sentences is that its search space is (much) smaller than the space of all first order sentences. Still, every sentence in a first order language has a logical equivalent sentence in the reduced subset, so there is no loss of expressiveness.

∘ A sentence p *subsumes* a sentence q via a substitution θ if $p\theta \subseteq q$.

∘ Two sentences p and q are called *equivalent* ($p \sim q$) if p subsumes q and q subsumes p.

∘ A set of literals S is called *reduced* if $S' \subseteq S$ and $S \sim S'$ implies $S = S'$.

∘ Given a sentence p, the following algorithm [4] gives the reduced sentence q such that $p \sim q$.

1. Set q to p.
2. Find a literal L in q and a substitution θ such that $q\theta \subseteq q \setminus \{L\}$. If this is impossible, stop.
3. Change q to $q\theta$ and go to 2.

It appears that if both p and q are reduced sentences and $p \sim q$, then p and q are equal up to renaming variables.

According to Shapiro [6] a sentence $p = A \leftarrow B$ is called *reduced* if both A and B are reduced. It is not clear whether this means that the sentence $\{Q(x, y)\} \leftarrow \{P(x), P(y)\}$ should be regarded as not reduced, since the condition side is not reduced. The set $\{Q(x, y), \neg P(x), \neg P(y)\}$ however, is reduced. We consider this sentence, and all sentences such that the set of all literals is reduced, reduced.

3 Shapiro's refinement operator ρ_0.

In Shapiro's paper [6] refinement operators are just a part of an incremental inductive algorithm. In this paper we do not discuss these algorithms in detail but concentrate on the relation between reduction and refinement.

The refinement operator ρ_0 is defined for reduced sentences, not for general first order sentences. Completeness of this operator therefore means that there must be a ρ_0-chain from $\Box$ to q for every *reduced* sentence q in a first order language $\mathcal{L}$. For some sentences q it is possible to construct an infinite chain of sentences from $\Box$ to q such that every sentence subsumes its successor. Considering only sentences of size$\leq k$ and demanding a refinement operator to give refinements with strictly larger size however, prohibits infinite ρ_0-chains.

The following notions are used by Shapiro's ρ_0.

∘ A substitution θ is said to *decrease* a set of literals S if $|S\theta| < |S|$.

Consequently, a substitution decreases a sentence $p = A \leftarrow B$ if it decreases either A or B.

∘ A *most general* term (literal) is a term of the form $f(y_1, ..., y_n)$, where f is an k-place function (predicate) symbol and every y_i, $1 \leq i \leq n$, is a distinct new variable.

∘ Let L_1 and L_2 be literals of $\mathcal{L}$ and U be a set of literals or a sentence. We say that L_1 is *more general than L_2 with respect to U* if there exists a substitution θ such that $L_1\theta = L_2$ and $U\theta = U$.

Definition 5.13 [6] Let $p = A \leftarrow B$ be a reduced sentence of $\mathcal{L}$. Then $q \in \rho_0(p)$ when exactly one of the following holds:

1. $q = p\theta$, where $\theta = \{x/y\}$ does not decrease p and both variables x and y occur in p.
2. $q = p\theta$, where $\theta = \{x/f(y_1, ..., y_n)\}$ does not decrease p, f is an n-place function symbol, x occurs in p and every y_i, $1 \leq i \leq n$, is a distinct variable that does not occur in p.
3. $q = A \cup \{L\} \leftarrow B$, where L is a most general literal with respect to $A \leftarrow B$ for which $q = A \cup \{L\} \leftarrow B$ is reduced.
4. $q = A \leftarrow B \cup \{L\}$, where L is a most general literal with respect to $A \leftarrow B$ for which $q = A \leftarrow B \cup \{L\}$ is reduced.

In general, the set of literals L that can be added in cases 3 and 4 of the ρ_0 operator is infinite. However, Shapiro [6] has sketched an algorithm that systematically generates all such literals of size$\leq k$. Since there are finitely many such literals, this algorithm is locally finite.

Let $p = A \leftarrow B$ be the sentence to be refined. Choose a most general literals L. Successively choose a variable x that occurs in L and perform one of the following operations:

1. Choose a variable y that occurs in L but not in p, and set L to $L\{x/y\}$.
2. Choose a most general term t, and set L to $L\{x/t\}$.
3. Choose a variable y that occurs in p, and set L to $L\{x/y\}$.

until either $A \cup \{L\} \leftarrow B$ is reduced, or $size(L) > k$. If the second condition holds then fail. Otherwise verify that the sentence $A \cup \{L'\} \leftarrow B$ is not reduced, for any literal L' that is more general than L with repect to p. via a substitution that does not change $A \leftarrow B$. If so, return L.

In the next section we will comment some lemma's and a theorem concerning Shapiro's ρ_0-operator. The main point is that ρ_0 is not a complete refinement operator.

4 ρ_0 is not a complete refinement operator.

All reduced sentences represent an equivalence class under subsumption. Also, every equivalence class under subsumption contains a unique reduced sentence (up to renaming variables). The partial ordering induced by subsumption, $p\theta \subseteq q$ can be split into $p\theta = q$, and $p \subseteq q$. When every q that satisfies one of these two relations with p can be derived from p by ρ_0, then ρ_0 would be complete. Shapiro [6] tries to prove this by lemma 5.15 ($p\theta = q$) and lemma 5.16 ($p \subseteq q$).

4.1 Incorrectness of Shapiro's lemma 5.15.

Lemma 5.15 [6] *Let p and q be two reduced sentences such that $p\theta = q$ for some substitution θ then there is a finite total ρ_0-chain from p to q.*

From the definition of *decrease*, it does not follow that if a sentence p is reduced, and a substitution θ is non-decreasing, then $p\theta$ is also reduced. For example, consider the sentence
$p = \{P(a, w), P(x, b)\} \leftarrow$ and the substitution $\theta = \{w/b\}$.
p is reduced and θ is non-decreasing, but
$p\theta = \{P(a, b), P(x, b)\} \leftarrow$
can be reduced with the substitution $\{x/a\}$. Since we want to restrict our search space to reduced sentences, (ρ_0 is defined only for reduced sentences) such substitutions are not allowed.

Shapiro states that examining the proof of Theorem 4 in Reynolds' paper [5] shows that lemma 5.15 is a generalization of this theorem. Reynolds' theorem 4 states that every atomic sentence can be derived from $\square$ by a ρ_0-like operator. Since Reynolds only considers atomic sentences, his theorem has no relation with sentences being either reduced or not reduced. Therefore, his theorem is not necessarily a special case of lemma 5.15.

In order to prove the incorrectness of lemma 5.15 (and 5.16), we use two lemma's. The first lemma states that no predicate or function symbol will disappear when applying ρ_0.

Lemma 4.1 *Let p and q be reduced sentences such that $q \in \rho_0^n(p)$, $n \geq 1$. Then for every literal L in p there is a literal $L\theta$ in q.*

Proof. Proof by induction on m, the number of applications of ρ_0.

$m = 1$ For any single substitution the lemma is trivial. When a literal is added, the resulting sentence is required to be reduced so no literals can disappear and the empty substitution will satisfy the lemma.

Next we assume that the lemma holds for $m = 0, .., n-1$. By the inductive assumption we know that for $m = n-1$:

$\forall L \in p, q' \in \rho_0^{n-1}(p), q \in \rho_0(q'), \exists \sigma, L\sigma \in q'$.

For any such $L\sigma$ in q' applying ρ_0 such that $q \in \rho_0(q')$ directly results in a literal $L\sigma\tau \in q$ by the same arguments as at $m = 1$. Taking $\theta = \sigma\tau$ leads to $L\theta \in q$, which completes our proof. □

The next lemma states that literals cannot be 'absorbed' by other literals when applying ρ_0.

Lemma 4.2 *Let p and q be reduced sentences such that $q \in \rho_0^n(p)$, $n \geq 0$. Then for all sets of literals $p' \subseteq p$ there is a set of literals $q' \subseteq q$ such that $|p'| = |q'|$ and $p'\theta = q'$ for some substitution θ.*

Proof. Since adding literals does not affect existing literals we need only consider ρ_0-applications that are substitutions.

By lemma 4.1 we know that for every literal L in p', there exists a literal $L\sigma$ in q. Defining θ as the union of all these σ's, we get $q' = p'\theta \subseteq q$,

For arbitrary substitutions θ we have $|p'\theta| \leq |p'|$. Since all substitutions in ρ_0 are required to be non decreasing, we have $|p'\theta| \geq |p'|$, we conclude $|p'| = |p'\theta|$. □

The next lemma shows that there is no proof for lemma 5.15.

Lemma 4.3 *For some reduced sentences p and q, such that $p\theta = q$, there is no finite total ρ_0-chain from p to q.*

Proof. We consider the next two sentences:

$p = \{P(a,w), P(x,b), P(c,y), P(z,d)\} \leftarrow$ (as before), and

$q = \{P(a,b), P(c,b), P(c,d), P(a,d)\} \leftarrow$.

We prove that none of the 4 items of $\rho_0(p)$ is a candidate for a ρ_0-chain from p to q. We start with p:

1. Not applicable, since all variables in p must become distinct constants, and by lemma 4.2 literals cannot dissapear.

2. Every substitution that introduces a function symbol that does not occur at the same place in a literal in q is no candidate. Since, by lemma 4.1 such a symbol cannot be removed by further applications of ρ_0.

   As can be verified, every substitution towards q, e.g. $\{w/b\}$ results in a sentence that is not reduced.

3. Adding a most general atom at the conclusion side cannot be a candidate. Since by lemma 4.2 such an atom cannot disappear.

4. Adding a most general atom at the condition side cannot be a candidate. Since by lemma 4.2 such an atom cannot disappear.

We conclude that no member of $\rho_0(p)$ can be part of a ρ_0-chain from p to q, and, since $p \neq q$, there is no finite total ρ_0-chain from p to q. □

4.2 Incorrectness of Shapiro's lemma 5.16.

Lemma 5.16 [6] *Let p and q be two reduced sentences such that $p \subseteq q$ then there is a finite total ρ_0-chain from p to q.*

Consider the next two sentences
$p = \{P(x)\} \leftarrow \{Q(x,a)\}$, and
$q = \{P(x)\} \leftarrow \{Q(x,a), Q(y,z), Q(z,y)\}$.
Both sentences are reduced and $p \subseteq q$. According to lemma 5.16 there exists a finite total ρ_0 chain from p to q. By the inductive assumption in Shapiro's proof of the lemma there is a finite total ρ_0-chain from p to $p' = \{P(x)\} \leftarrow \{Q(x,a), Q(y,z)\}$. Since this sentence is not reduced the proof of lemma 5.16 does not hold.

The next lemma shows that there is no alternative proof for lemma 5.16.

Lemma 4.4 *For some reduced sentences p and q, such that $p \subseteq q$, there is no finite total ρ_0-chain from p to q.*

Proof. We consider the sentences p and q as above. None of the 4 items of $\rho_0(p)$ is a candidate for a ρ_0-chain from p to q:

1. Not applicable, since there are no two distinct variables in p.
2. Every function substitution will introduce a function symbol that does not occur at the same place in q. By lemma 4.1 such a symbol cannot be removed by further applications of ρ_0.
3. Adding a most general atom at the conclusion side cannot be a candidate. Since by lemma 4.2 such an atom cannot disappear.
4. We distinguish two cases of adding a literal L at the condition side: either there exists a substitution θ such that $L\theta \in q \setminus p$ or such a substitution does not exist.

   If such a θ does not exist application of case 4 cannot be part of a ρ_0-chain from p to q analog to case 3.

   If on the other hand such a θ does exist L must be an alphabetic variant of either $Q(y,z)$ or $Q(z,y)$, say $Q(v,w)$. However, the resulting sentence

   $p' = \{P(x)\} \leftarrow \{Q(x,a), Q(v,w)\}$

   is not reduced (p' can be reduced by $\theta = \{v/x, w/a\}$).

We conclude that no member of $\rho_0(p)$ can be part of a ρ_0-chain from p to q, and, since $p \neq q$, there is no finite total ρ_0-chain from p to q. $\square$

4.3 Incorrectness of Shapiro's theorem 5.14.

Theorem 5.14 [6] *ρ_0 is a complete refinement operator over $\mathcal{L}$.*

The lemma's in the former subsections do not imply that there are reduced sentences in $\mathcal{L}$ that cannot be derived at all. In an incremental inference algorithm it might be sufficient that every reduced sentence q can be derived from $\square$ in at least one way, not necessarily through all reduced sentences p such that $p\theta \subseteq q$. The next lemma shows that some sentences cannot be derived at all.

Lemma 4.5 *For some reduced sentence q there is no finite total ρ_0-chain from $\square$ to q.*

Proof. We consider the reduced sentence

$q = \{P(v,w), P(w,v), P(x,y), P(y,z), P(z,x)\} \leftarrow$

If there exists a ρ_0-chain from $\Box$ to q, then there also exists a reduced sentence p such that for some n, $p \in \rho_0^n(\Box)$ and $q \in \rho_0(p)$.

We now show that for every such p none of the four cases of $\rho(p)$ is a candidate for $q \in \rho_0(p)$.

1. p must contain one variable that is not in q, call it u. All possible unifications u with one of the variables $v, \ldots, z$. Since substitutions are non-decreasing, $|p| = |q|$ and p equals q with one variable changed to t. All these sentences are not reduced sentences, as can be verified.

2. Since no function symbols occur in q, q cannot be obtained from any sentence p by a function substitution.

3. For all $L \in q$ $q \setminus \{L\}$ is not reduced, so q cannot be obtained from any p by adding a most general atom at the conclusion side.

4. Not applicable, since q has no literals at the condition side.

We conclude that there exists no reduced sentence p such that $q \in \rho_0(p)$, therefore there is no ρ_0-chain from $\Box$ to q. $\Box$

Theorem 1 *ρ_0 is not a complete refinement operator.*

Proof Follows directly from lemma 4.5. $\Box$

While writing this paper, the whole set of sentences that cannot be derived is unknown. The reader might oppose that the underivable sentence of lemma 4.5 is not a meaningful sentence, whatever meaningful means. This sentence however, chosen because it has no predecessors, also has non-derivable variants with successors that look quite meaningful. For example consider the next sentence:

$q = \{P(u)\} \leftarrow \{Q(u,v,w), Q(u,w,v), Q(u,x,y), Q(u,y,z), Q(u,z,x)\}$

This sentence has some predecessors, for example $q \setminus \{P(u)\}$. Also, some of the u's can be anti-unified. However, every sentence p, such that $q \in \rho_0(p)$ by unifying one of the variables v-z or by adding one of the literals on the condition side, is not reduced. (The same holds for $q \setminus \{P(u)\}$, and all sentences found by anti-unifying some u's.) So, there is no ρ_0-chain from $\Box$ to q.

Also note that this sentence is a Horn sentence, and ρ_0 is neither a complete refinement for reduced Horn sentences.

5 A most general refinement operator for general sentences.

In the previous section we have shown that Shapiro's refinement operator ρ_0 is not complete. Laird [2] has defined a refinement operator for general first order sentences. As is shown before, the incompleteness of ρ_0 is merely a result of not accepting refinements that are not reduced. A solution for defining a complete most general refinement operator can be dropping the condition of reducedness. The resulting operator then works on a (much) larger search space in which many sentences have the same logical expressiveness. Laird's refinement operator works in this way. He does not give a proof of completeness of his version of ρ_0 which we will call ρ_L, instead he refers to the proof of Shapiro. Moreover, Laird does not mention the difference between his and Shapiro's operator. In this section we will prove that ρ_L is a complete refinement operator for arbitrary first order languages.

5.1 Definition of ρ_L.

Laird uses a different notation to define his refinement operator. Instead of sentences $A \leftarrow B$ where A and B are sets of atoms, Laird considers clauses of a language $\mathcal{L}_L$ where repetition of literals is allowed. Therefore substitutions are never decreasing. Also, (refinements of) clauses are not required to be reduced.

Definition 1[2] Let $p = A \leftarrow B$ be a clause in the language $\mathcal{L}_L$. Then $q \in \rho_L(p)$ when exactly one of the following holds:

1. $q = p\theta$, where $\theta = \{x/y\}$ and both variables x and y occur in p.
2. $q = p\theta$, where $\theta = \{x/t\}$ and t is a most general term.
3. $q = A \vee P \leftarrow B$, where P is a most general atom.
4. $q = A \leftarrow B \wedge P$, where P is a most general atom.

In order to proof the completeness of ρ_L we will first prove some lemma's that are variants of lemma 5.15 and 5.16 in [6].

Lemma 5.1 *Let p and q be two clauses in $\mathcal{L}_L$ such that $p\theta = q$ for some substitution θ then there is a finite total ρ_L-chain from p to q.*

Proof. Because Laird does not restrict to reduced clauses and arbitrary substitutions are allowed, this lemma is a straightforward generalization of theorem 4 in Reynolds' paper [5]. $\square$

Lemma 5.2 *Let p and q be two clauses in $\mathcal{L}_L$ such that all literals in p are in q, then there is a finite total ρ_L-chain from p to q.*

Proof. The proof is by induction on n the number of atoms in q but not in p. If $n = 0$ then $p = q$, and the empty chain satisfies the lemma. Assume that for some $0 \leq m < n$, that there is a ρ_L-chain from p to p_m such that p_m is p with m atoms added and p_m is contained in q. Let L be a literal in q but not in p_m.

If L is a most general literal then by definition of ρ_L, $p_{m+1} \in \rho_L(p_m)$, where p_{m+1} is p_m with L added on the right side. Otherwise there is a most general atom M such that $M\theta = L$, and $p'_{m+1} \in \rho_L(p_m)$, where p'_{m+1} is p_m with M added. By lemma 5.1 there is a finite total ρ_L-chain from p'_{m+1} to p_{m+1}. In both cases, there is a finite total ρ_L-chain from p to p_{m+1}. $\square$

Theorem 2 *ρ_L is a complete refinement operator over $\mathcal{L}_L$.*

Proof. Taking $p = \square$ and q an arbitrary clause in $\mathcal{L}_L$ satisfies the preconditions of lemma 5.2. So for any clause $q \in \mathcal{L}_L$ there is a ρ_L-chain from $\square$ to q and $\rho_L^* = \mathcal{L}_L$. $\square$

In [7], Shapiro has included the Prolog-source of another general refinement operator that is similar to Laird's ρ_L. Like Laird's, this operator does not restrict the search space of hypotheses to reduced sentences.

6 A most general refinement operator for reduced sentences.

The refinement operator ρ_L, as discussed in the previous section, is complete for all clauses in a first order language $\mathcal{L}_{\mathcal{L}}$. Shapiro's original ρ_0 was intended to be complete for the smaller search space of reduced sentences of any first order language $\mathcal{L}$. In this section we suggest some changes to Shapiro's ρ_0 that result in a refinement operator ρ_p which is complete for reduced sentences.

6.1 Components.

In section 4 we have shown that ρ_0 is not a complete refinement operator. It appeared that some reduced sentences are not reachable from any other reduced sentence by one application of ρ_0. In order to be a complete refinement operator ρ, for any sentence q there must be a sentence p such that $q \in \rho(p)$. Such sentences p can be located by 'applying the inverse of ρ'. In lemma 4.5 all such inverse applications resulted in non-reduced sentences. All these sentences p that were not reduced shared the property that p consisted of two substructures such that one substructure of p was a more general form of the other. An attempt to define a complete most general refinement operator for reduced sentences could be disallowing sentences that contain more than one substructure . In this subsection we formalize the concept of substructures called components and show that disallowing more-component sentences does not lead to a complete operator.

Sentences can be partitioned into *components*. A component is a 'linked substructure' of a sentence. Formally we can find all components of a sentence as follows:

- Given a sentence p

  1. Assign $p_1 = p$, $i = 1$
  2. Choose a literal L of p_i, assign $c_i = \{L\}$ and $vars_i = \{x | x \text{ is a variable of } L\}$
  3. Add to c_i all literals of p_i that contain a variable of $vars_i$ and add to $vars_i$ all variables of those literals that are not yet in $vars_i$. Repeat this step until no more such literals can be found.
  4. Assign $p_{i+1} = p_i \setminus \{L | L \in c_i\}$
  5. If $p_{i+1} = \phi$ then assign $n = i$ and stop, otherwise increase i with 1 and go to step 2.

After termination $c_1, \ldots, c_n$ are the components of p. Clearly, every grounded atom is a component by itself (since it has no variables).

Helft [1] uses the notion of Horn clauses being *linked*, this notion is similar to sentences containing only one component. He states that 'non-linked clauses are generally not meaningful'.

The following lemma states that disallowing more-component sentences does not make a refinement operator defined on reduced sentences complete.

Lemma 6.1 *Let ρ, an arbitrary refinement operator consisting of non-decreasing substitutions and additions of literals defined for reduced sentences, be incomplete. Then, ρ defined for reduced one-component sentences is also incomplete.*

Proof. ρ is incomplete for more-component sentences, so there exists a reduced sentence q such that there is no reduced sentence p and $q \in \rho(p)$. For any such q we can construct a reduced one-component sentence q_x by replacing all literals L of q by L_x, where L_x is L with one extra argument x on the first place, and x is a variable not occurring in q.

This q_x (q with x's added) can have predecessors found by anti-unifying x with x_i's, we call them q_j. There is only a finite number of these q_j's and all of them contain all arguments of q.

Every other inverse application of ρ to q_x or to any q_j, results in a sentence p_x. This inverse application has a corresponding inverse application of ρ to q that leads to a non reduced sentence p. For this p there is a literal L in p and a substitution θ such that $p\theta \subseteq p \setminus \{L\}$.

Defining $\theta_x = \theta \cup \bigcup_i \{x_i/x\}$, gives us $p_x\theta_x \subseteq p_x \setminus \{L_x\}$. It appears that every p_x is non-reduced for any inverse application of ρ that is not an anti-unification of x. q_x has no other predecessors than the q_j's, and ρ is also incomplete for reduced one-component sentences. $\square$

Example. As an example we revisit the sentence that could not be derived in lemma 4.5:

$q = \{P(v,w), P(w,v), P(x,y), P(y,z), P(z,x)\} \leftarrow$
becomes
$q_u = \{P_u(u,v,w), P_u(u,w,v), P_u(u,x,y), P_u(u,y,z), P_u(u,z,x)\} \leftarrow$
One possible predecessor of this sentence is
$q_1 = \{P_u(u_1,v,w), P_u(u,w,v), P_u(u_1,x,y), P_u(u,y,z), P_u(u,z,x)\} \leftarrow$
However, when one of the variables of q in q_u is anti-unified, or when a literal is removed, the resulting sentence p is not reduced.

6.2 Inverse reduction.

In section 2.2 we have given Plotkin's reduction algorithm. Incompleteness of a refinement operator arises when all inverse applications of it to a reduced sentence result in non-reduced sentences. When making a refinement operator complete, the most natural place for sentences to lead to a formerly underivable sentence is (one of) the reduced representative(s) of a non-reduced sentence found by such an inverse application. As we will show, ρ_0 can be made complete by first generating all sentences of size$\leq k$ that are in the equivalence class under subsumption of the reduced sentence to be refined. In this section we will present an inverse reduction algorithm that can be used later on to generate all those equivalent sentences.

First we reformulate the reduction algorithm.

∘ Given a sentence p, the following algorithm gives the reduced sentence q such that $p \sim q$.

1. Set q to p.
2. Find a literal L in q and a substitution θ such that $q\theta \subseteq q \setminus \{L\}$. If this is impossible, stop.
3. Change q to $q \setminus \{L\}$ and go to 2.

The only change to the algorithm is made in step 3. In the original version, q was changed to $q\theta$ which could mean deleting more than one literal in one cycle of the algorithm. For such a case it is clear that, with a cycle for each such literal and the same substitution θ, all those literals can be repeatedly deleted. However, this version of the algorithm is much easier to invert. The following inverse reduction algorithm assumes that the literals of a reduced sentence p have a fixed place such that L_i is the i'th literal of p.

∘ Given a reduced sentence p, all sentences q such that $p \sim q$, and $size(q) \leq k$, where k is fixed, are generated by the call InvRed(p,1,ϕ).

Procedure InvRed(q : sentence; n : integer; θ: substitution)

```
if $size(q) \leq k$
  output($q$)
  for$i := n$ to $|p|$
     forall $L$ such that $\exists\sigma$: $L\theta\sigma = L_i$ and $q\sigma = q$
        InvRed($q \cup L$,$i$, $\theta\sigma$)
```

Because every literal of p has only a finite number of inverse substitutions only finitely many branches are generated by each call. In every recursion a literal is added to q, this increases the size of q, so every branch terminates. As an invariant we have that $q\theta = p$.

The parameter n acts like a place counter of the literals to be inverted. The use of it prohibits that if a sentence q can be generated by respectively adding inverse substitutions of L_i and L_j, $i < j$, the same sentence will also be generated by respectively adding the inverse substitutions of L_j and L_i. (Leon van der Torre suggested this more efficient recursive approach.)

6.3 Definition of ρ_r.

Using the inverse reduction algorithm of the former subsection we now define a complete most general refinement operator for reduced sentences. In this definition ρ_0' equals ρ_0 where the sentence to be refined is not required to be reduced. (The refinements still are required to be reduced!)

Definition 2 Let p be a reduced sentence of L. Then $q \in \rho_r(p)$ when $q \in \rho_0'(p_1)$, for some p_1 in the equivalence class of p under subsumption.

Theorem 3 *ρ_r is a complete refinement operator for reduced sentences.*

Proof. Let q be a reduced sentence with $size(q) \leq k$. The following algorithm finds a ρ_r-chain from $\Box$ to q.

1. Assign q to *chain*.
2. If q contains a most general literal L with respect to $q \setminus \{L\}$, then assign $p = q \setminus \{L\}$.

   Otherwise, find a (possibly non-reduced) sentence p such that $q = p\theta$ for some substitution θ such that $q \in \rho_0'(p)$ through case 1 or 2.
3. Let p' be the reduced representative of p, and append p' to *chain*,
4. If $p' = \Box$ then output reverse(*chain*), otherwise assign $q = p'$ and go to 2.

If, in step 2, q does not contain a most general literal L with respect to $q \setminus \{L\}$, then q contains a most general term or q contains a variable twice, and hence there exists a p such that $p\theta = q$.

In every iteration the size of q diminishes, so the algorithm terminates. $\Box$

Example. We revisit the sentence of lemma 4.5.

$q = \{P(v,w), P(w,v), P(x,y), P(y,z), P(z,x)\} \leftarrow$

Inverse application of ρ_0' that changes one of the first two literals results in a non-reduced sentence with reduced equivalent

$p' = \{P(x,y), P(y,z), P(z,x)\} \leftarrow$

One member of the equivalence class of p', obtained by adding the atoms $P(v,w)$ and $P(w,u)$, is the non-reduced sentence

$p_1 = \{P(v,w), P(w,u), P(x,y), P(y,z), P(z,x)\} \leftarrow$

From this sentence p_1, q can be derived by the unification $\{u/w\}$.

Inverse application of any of ρ_0 to one of the last three literals results in a non-reduced sentence with reduced equivalent

$p' = \{P(v,w), P(w,v)\} \leftarrow$

One member of the equivalence class of p' obtained by adding the atoms $P(x,y)$ and $P(y,z)$ is the non-reduced sentence

$p_2 = \{P(v,w), P(w,v), P(x,y), P(y,z)\} \leftarrow$

From this sentence p_2, q can be derived by adding the most general literal $P(z,x)$.

Properties.

ρ_r is a complete version of Shapiro's refinement operator ρ_0.

In a system like Shapiro's Model Inference System, refinement operators are used to replace too general hypotheses by weaker refinements. In such a system, the use of ρ_r would lead to reduced hypotheses only. This is an advantage over the only other complete refinement operator that we know of, Laird's ρ_L, where also (many) non-reduced sentences have to be stored in memory.

However, to find all reduced refinements of a sentence, many non-reduced sentences have to be generated that are all submitted to ρ'_0.

Generating and testing all possible most general literals with repect to all these logically equivalent sentences is the most time-consuming step in computing $\rho_r(p)$. In the next subsection we show that this can be avoided.

6.4 Definition of ρ_p.

In the example of the previous section it is shown that the formerly underivable sentence q now could be derived in more than one way, unifying two variables of p_1 or adding a predicate to p_2. In general, any reduced sentence that can be anti-unified resulting in a (possibly non-reduced) sentence can be derived from the reduced equivalent of that sentence through unification. Also, sentences that contain a most general term are derivable through a most general term substitution, and sentences that contain a most general literal are derivable through adding a most general literal.

Since we are not interested in finding refinements q from p in more than one way, we can save a lot of work by pruning the derivation tree of ρ_r. We will show that every reduced sentence q that can be derived from p by adding a most general literal to a non-reduced sentence p_1 can also be derived without adding most general literals to non-reduced sentences.

Let p be the reduced equivalent of $p_1 = p \cup S$, where $S\theta \subseteq p$ for some θ. Because p_1 is not reduced, and $q = p \cup S \cup \{L\}$ is, L must share at least one variable with S that is not in p. If anti-unifying one such variable results in a reduced sentence, then q can be derived from that sentence through unification. If the resulting sentence is not reduced, q can be derived from its reduced equivalent. Repeatly anti-unifying variables that are in S and L but not in p will result in either the sentence p or $p \cup \{L\}$. The latter is derivable from p, so there is a ρ_r-chain from p to q without adding most general atoms to non-reduced sentences.

We suggest that non-reduced sentences are only used for substitutions. The resulting refinement operator ρ_p is also complete for reduced sentences:

Definition 3 Let p be a reduced sentence of L. Then $q \in \rho_p(p)$ when $q \in \rho_0(p)$, or $q = p_1\theta$ and $q \in \rho'_0(p_1)$ for some p_1 in the equivalence class of p under subsumption.

7 Conclusions.

We have been looking for a most general refinement operator for reduced sentences. First, we have shown that Shapiro's refinement operator ρ_0 is not complete for reduced first order sentences. Another refinement operator for general first order sentences, Lairds' variant of ρ_0, is proven to be complete. We have studied the causes of incompleteness when working with reduced sentences only. As a product, an inverse reduction algorithm has been presented. Finally we have described a new refinement operator ρ_r based on inverse reduction, complete for reduced sentences. The resulting refinement operator requires less storage space then Lairds' operator when applied in a system like Shapiro's MIS, since only reduced sentences are possible hypotheses. Computationally, the ρ_r-operator is not too attractive, since before refining, all sentences in an equivalence class under subsumption are generated. Some remarks on computing less refinements of non-reduced sentences are made, resulting in a more efficient complete refinement operator ρ_p.

Acknowledgements. I would like to thank my thesis supervisor Cor Bioch for the suggestions that led me to the subject of this article, him and Arie de Bruin for the supporting discussions, and Leon van der Torre for reading and commenting all versions of this article.

References

[1] N. Helft. Inductive Generalization: A Logical Framework. In I. Bratko and N. Lavrac, editor, *Progress in Machine Learning: EWSL-87*, pages 149–157. Sigma Press, Wilmslow, England, 1987.

[2] P.D. Laird. *Learning from Good and Bad Data.* Kluwer Academic Publishers, 1988.

[3] W.E. Nijenhuis and C. Witteveen. Constructive identification with Poole's default logic. Technical Report 90-96, Faculty of Technical Mathematics and Informatics, 1990.

[4] G.D. Plotkin. A Note on Inductive Generalization. In *Machine Intelligence 5*, pages 153–163. Edinburgh University Press, Edinburgh, 1970.

[5] J.C. Reynolds. Transformational Systems and the Algebraic Structure of Atomic Formulas. In B. Meltzer and D. Mitchie, editor, *Machine Intelligence 5*, pages 135–153. Edinburgh University Press, Edinburgh, 1970.

[6] E.Y. Shapiro. Inductive Inference of Theories from Facts. Technical Report 624, Department of Computer Science, Yale University, New Haven. CT., 1981.

[7] E.Y. Shapiro. *Algorithmic program debugging.* MIT Press, 1983.