Subsumption and Refinement in Model Inference

Patrick R. J. van der Laag[1,2] and Shan-Hwei Nienhuys-Cheng[1]

[1] Department of Computer Science, Erasmus University of Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, the Netherlands
[2] Tinbergen Institute

Abstract. In his famous Model Inference System, Shapiro [10] uses so-called refinement operators to replace too general hypotheses by logically weaker ones. One of these refinement operators works in the search space of reduced first order sentences. In this article we show that this operator is not complete for reduced sentences, as he claims. We investigate the relations between subsumption and refinement as well as the role of a complexity measure. We present an inverse reduction algorithm which is used in a new refinement operator. This operator is complete for reduced sentences. Finally, we will relate our new refinement operator with its dual, a generalization operator, and its possible application in model inference using inverse resolution.

1 Introduction

In 1981, Shapiro [10] has introduced the notion of model inference. It has since then drawn a lot of attention in the world of inductive learning using logic. Even now the new operation of inverse resolution [5] is in fashion, people still discuss and use his ideas of inference and learning problems [1, 3, 7]. Given a sequence of positive and negative examples of an unknown concept, his incremental Model Inference System tries to find a theory (finite set of hypotheses) that can infer all given positive examples and none of the negative examples. The system essentially uses two techniques: if the theory is too strong (a negative example can be inferred from it) the backtracing algorithm locates a guilty hypothesis and refutes it; if the theory is too weak (a positive example can not be inferred) then refinements (specializations), found by a refinement operator, of the thrown away hypotheses are added. In the limit, a theory will be found that is neither too strong nor too weak.

In this article we will discuss Shapiro's *refinement operator* ρ_0. This operator is defined for *reduced* sentences in a first order language where sentences and refinements of them are restricted by some complexity measure size. The notion of reduced sentences, related to *subsumption*, is introduced by Plotkin [8].

The strength of Shapiro's model inference algorithm is its theoretical approach, the formal description of the operators used in it and their properties. One of these properties is the completeness of a refinement operator for a subset of a first order language, i.e., every sentence in the subset can be derived from the empty sentence by repeatedly applying the refinement operator. We will show that, exactly because of theoretical reasons, the refinement operator ρ_0 is not complete for reduced sentences, as Shapiro claims. To understand the problems, we should know more about the concepts of subsumption and size. Also, we will show that the definition of size which Shapiro has adopted from Reynolds [9] is inadequate for the theories he tries to build. We will introduce another kind of complexity measure.

Due to special properties of subsumption, we will show that the technique of *inverse reduction* is needed for certain problems. An algorithm for inverse reduction finds non-reduced clauses that are equivalent to a given reduced sentence. With this new technique and the new complexity measure we can change the ρ_0-operator into a refinement operator that is complete for reduced sentences.

Our refinement operator can derive exactly one representative of every equivalence class under subsumption, namely the reduced sentence in it (this sentence can be reached by different sentences since the subsumption ordering is a lattice, not a tree). The space of all reduced sentences is much smaller than the space of all sentences in a first order logic. Therefore, considering reduced sentences only is more efficient than considering all sentences.

Some of the results of a working report [12] are included in this article.

The article is divided in the following sections. In Sect. 2 we concentrate on some properties of subsumption which are important in proving the (in-)correctness of refinement operators. In Sect. 3 we introduce some terminology adopted from Shapiro and we show the incompleteness of ρ_0 with examples. Also, we discuss the complexity measure rsize and its shortcomings when related to subsumption. In Sect. 4 we define a new refinement operator and a new complexity measure. In Sect. 5 we compare our new refinement operator with another refinement operator [2] that is complete for first order sentences. In Sect. 6 we look at refinements in a wider framework. We relate refinement operators to their duals, generalization operators, and these generalization operators to inverse resolution and model inference. These relations will also be a subject for future research.

2 Subsumption, Reduction and Inverse Reduction

Let $\mathcal{L}$ be a language of first order logic. In this language we use P, Q, R,... for predicate symbols, f, g, h,... for function symbols, a, b, c,... for constants, and x, y, z,... for variables. Throughout this article, constants are treated as functions with arity zero. Atoms are denoted by A, B,.... A literal is an atom or the negation of an atom, and is denoted by L, M,.... Every sentence in $\mathcal{L}$ is a set of literals:

$$\{A_1, \dots, A_m, \neg B_1, \dots, \neg B_n\}$$

where $\neg B_j$ is the negation of the atom B_j. A sentence represents the disjunction of its literals, where all variables in it are universally quantified over the whole sentence. Sentences can also be written in the following form:

$$\{A_1, \ldots, A_m\} \leftarrow \{B_1, \ldots, B_n\}$$

Sentences are denoted by p, q,. . ., and substitutions by θ, σ, τ. All these symbols may have subscripts. A Horn sentence requires $m \leq 1$. The results in this article are described for the set of first order sentences but hold for the set of Horn sentences as well.

2.1 Subsumption and Reduction

The notions of subsumption and reduction originate from Plotkin [8], and are defined as follows:

Definition 1. A sentence p *subsumes* a sentence q ($p \succeq q$) iff there exists a substitution θ such that $p\theta \subseteq q$.

A sentence p *properly subsumes* a sentence q ($p \succ q$) iff p subsumes q but q does not subsume p.

Two sentences p and q are called (*subsume*) *equivalent* ($p \sim q$) iff p subsumes q and q subsumes p. This is an equivalence relation which defines a partition on sentences. Every set of equivalent sentences is called a (*subsume*) *equivalence class.*

A sentence p is called *reduced* if $p' \subseteq p$ together with $p \sim p'$ implies $p = p'$.

Example 1. The reduced sentence $\{P(x,y)\}$ is equivalent to $\{P(x,y), P(x,z)\}$ which is (therefore) not reduced.

If q is not reduced, we can reduce q to a sentence $p \subset q$ ($p \subseteq q$ and $p \neq q$) such that p is reduced and $p \sim q$. An algorithm to reduce sentences to their smallest equivalent subset is presented by Plotkin [8].

Subsumption is weaker than logical implication [6]. If a sentence p subsumes a sentence q then p logically implies q but not the other way around. $p = \{P(f(x))\} \leftarrow \{P(x)\}$ logically implies $q = \{P(f(f(y)))\} \leftarrow \{P(y)\}$, but p does not subsume q. In the rest of this article, we investigate the ordering induced by subsumption. We will define an operator that, given a reduced representative of an equivalence class under subsumption, yields reduced repesentatives that are subsumed by it. However, sentences that are logically implied but not subsumed will not be found.

It is proved by Plotkin [8] that if two equivalent sentences both are reduced then they are equal up to renaming variables, i.e., they are alphabetical variants.

Throughout this paper alphabetical variants are considered identical. We can therefore speak of one reduced representative of every equivalence class.

Subsumption has some unexpected properties. For example, substitutions do not always preserve equivalence, and subsets of reduced sentences need not be reduced.

Let $p = \{P(x,y)\}$, $q = \{P(x,y), P(x,z)\}$, $r = \{P(x,y), P(x,z), P(y,z)\}$, and $\theta = \{x/z\}$. Then $p \subset q \subset r$, $p \sim q$, $p\theta = \{P(z,y)\}$ and $q\theta = \{P(z,y), P(z,z)\}$. $\{P(z,z)\}$ is the reduced equivalent of $q\theta$, and $p\theta$ is not equivalent to $q\theta$. Also, $q \subseteq r$, where r is reduced and q is not.

The ordering induced by subsumption is a quasi ordering because $x \succeq x$ (reflexivity), and if $x \succeq y$ and $y \succeq z$ then $x \succeq z$ (transitivity). The empty sentence ($\Box$) is the maximal element in this ordering.

In a (learning) system that uses a search space of logic formulae, it often is a waste of time and memory to examine more than one sentence of an equivalence class. Since for any two sentences p and q if $p \sim q$ then $p \vdash r$ iff $q \vdash r$, using reduced sentences as a representative of an equivalence class might lead to more efficient (learning) systems.

However, operations such as substitutions on equivalent sentences can lead to non-equivalent sentences. To overcome this problem we want to build a simple algorithm to reverse the process of reduction. We need the following lemma and theorem for this algorithm.

Lemma 2. *Let p be a sentence. If $p\theta = p$, then for some natural number k, $L\theta^k = L$, for all literals L in p.*

Proof. θ must be injective: if $L_1\theta = L_2\theta$ for different $L_1, L_2 \in p$ then θ would decrease the number of literals in p, i.e., $|p\theta| < |p|$. For every literal L in p consider the following sequence

$$L, L\theta, L\theta^2, L\theta^3, \ldots$$

Since $p = p\theta = p\theta^2 = \cdots$ and since p is finite, not all $L\theta^i$ can be different. Then for some i, j, $i < j$, we have $L\theta^i = L\theta^j$. Because θ is injective, this implies $L\theta^{j-i} = L$.
For every L, let $n(L)$ be the smallest number such that $L\theta^{n(L)} = L$. Then $L\theta^i = L$ if i is a multiple of $n(L)$. Let k be the least common multiple of all $n(L)$. Then $L\theta^k = L$ for all $L \in p$. $\Box$

Lemma 3. *Let p be a reduced sentence, and $p \subseteq q$ such that $p \sim q$. Then there exists a substitution θ such that $q\theta = p$ and $L\theta = L$ for all literals $L \in p$.*

Proof. Since q subsumes p, for some σ, $q\sigma \subseteq p$, this implies $p\sigma \subseteq p$. If $q\sigma \subset p$, then also $p\sigma \subset p$ and p would not be reduced, and we conclude that $p\sigma = p$. By Lemma 2, we know that for some k, $L\sigma^k = L$ for all $L \in p$. Define $\theta = \sigma^k$. $\Box$

2.2 Inverse Reduction

Given a reduced sentence, an inverse reduction algorithm finds equivalent sentences. An inverse reduction algorithm is needed in Sect. 4 to define a complete refinement operator, ρ_r. Also, it shows the kind of sentences that are in the same equivalence class under subsumption.

Let p be a reduced sentence and m a given positive integer. We want to find all sentences that are equivalent to p and contain less than or exactly m literals. For every sentence q' such that $p \sim q'$, an alphabetic variant q of q' exists such that $q = p \cup r$. For example $p = \{P(x,x)\}$ is equivalent to $q' = \{P(u,u), P(u,v)\}$ which is an alphabetic variant of $q = \{P(x,x), P(x,y)\}$ that contains p. Therefore we only have to find equivalent sentences that contain p. By Lemma 3 we know that for every such sentence q a substitution θ exists, such that $q\theta = p$ and $L\theta = L$ for all literals $L \in p$. This implies that q can be reduced to p via θ, where θ is defined only on variables not occuring in p. The following example gives an idea which sentences are to be found.

Example 2. Let $p = \{P(x,x)\}$, and let $m = 3$. All possible q's such that $q \sim p$ and $|q| = 2$ are of the form $\{P(x,x), M_1\}$ where M_1 could be $P(y,z)$, $P(x,y)$, $P(y,x)$ or $P(y,y)$.
For $|q| = 3$, we get $q = \{P(x,x), M_1, M_2\}$ where some of the possible M_1, M_2 and corresponding θ are

M_1	M_2	θ
$P(y,z)$	$P(x,y)$	$\{y/x, z/x\}$
$P(y,z)$	$P(y,x)$	$\{y/x, z/x\}$
$P(y,z)$	$P(y,w)$	$\{y/x, z/x, w/x\}$
$P(x,y)$	$P(y,x)$	$\{y/x\}$
$P(y,y)$	$P(z,z)$	$\{y/x, z/x\}$

To find all sentences that are equivalent to a sentence p, literals have to be added to p. Since for a certain substitution θ, all added literals have to be mapped onto a literal in p, only literals that are more general than literals in p have to be added.

Algorithm 1 *(Inverse Reduction) Let p be a reduced sentence and let $m \geq 0$ be given. The following algorithm finds all sentences equivalent to p with $\leq m$ literals.*

```
Let $l = 0$, if $|p| \leq m$ then output $p$
While $l < (m - |p|)$ do
    $l := l + 1$
    For every sequence $\{L_1, \ldots, L_l\}$,
    where every $L_i \in p$, but the $L_i$'s are not necessarily distinct.
        Find all sets $r = \{M_1, \ldots, M_l\}$ such that $M_i\theta = L_i$ for all $i$,
        where every $M_i$ contains at least one variable not occuring $p$
        and $\theta = \{x_1/t_1, \ldots, x_m/t_m\}$, $x_j \notin \mathrm{var}(p)$ for all $j$;
        For every such $r$ output $p \cup r$
```

3 Incorrectness of the Refinement Operator ρ_o

3.1 Refinement

The ideas we present in this article are based on the refinement operators used in Shapiro's Model Inference System [10], for which he has defined three concrete refinement operators ρ_0, ρ_1, and ρ_2. We therefore first give a brief description of model inference. Let a first order language $\mathcal{L}$, an observational language $\mathcal{L}_o$, and a hypotheses language $\mathcal{L}_h$, such that $\mathcal{L}_o \subseteq \mathcal{L}_h \subseteq \mathcal{L}$, be given. Let M be an unknown model defined on $\mathcal{L}$ and let $\mathcal{L}_o^M = \{\alpha \in \mathcal{L}_o | \alpha$ is true in $M\}$. Suppose that we can enumerate all the sentences in $\mathcal{L}_o$ by α_1, $\alpha_2, \ldots$, and that we can tell the truth value V_i of every α_i. From these facts $\langle \alpha_i, V_i \rangle$, we want to find a finite set of sentences T expressed in $\mathcal{L}_h$ such that $\mathcal{L}_o^M \doteq \{\alpha \in \mathcal{L}_o | T \vdash \alpha\}$. Such a theory T is called a *$\mathcal{L}_o$-complete axiomatization* and it can be used to represent the model M.

Example 3. As an example, given a first order language $\mathcal{L}$, Shapiro's refinement operator ρ_2 uses $\mathcal{L}_o =$ the set of ground atoms in $\mathcal{L}$ and $\mathcal{L}_h =$ the set of atoms and context free transformations in $\mathcal{L}$, where context free transformations are sentences in the form $\{p(t_1, \ldots, t_n)\} \leftarrow \{p(x_1, \ldots, x_n)\}$ where all x_i's are distinct and occur in t_i.

If the language $\mathcal{L}$ contains the constant 0 (zero) and a function s (which can be interpreted as the successor function), then given some positive and negative examples like

$\langle plus(s(s(0)), s(0), s(s(s(0)))), true \rangle$ and
$\langle plus(s(0), 0, 0), false \rangle$,

the Model Inference System will find the hypotheses

$\{plus(x, 0, x)\} \leftarrow$, and
$\{plus(x, s(y), s(z))\} \leftarrow \{plus(x, y, z)\}$.

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. The following definitions are also his.

Definition 4. [10] A sentence q is a *refinement* of p if p (logically) 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 of $\rho(p)(n)$, that is, the set $\rho(p)$ restricted to sentences of size$\leq n$, is computable.

If there is a chain $p = p_0, p_1, \ldots, p_n = q$ such that $p_i \in \rho(p_{i-1})$, then we call it a *finite total ρ-chain.* We use $\rho^*(p)$ to denote the set of all sentences that can be reached by a finite total ρ-chain from p.

We change Shapiro's definition of completeness to weakly completeness because we want to use the notion of completeness for a stronger concept.

Definition 5. Let S be a subset of $\mathcal{L}$ which includes the empty sentence $\Box$. A refinement operator ρ over $\mathcal{L}$ is called *weakly complete* for S if $\rho^*(\Box) = S$.

A refinement operator ρ is called (*strongly*) *complete* for S if for any two sentences $p, q \in S$ such that q is a refinement of p, $q \in \rho^*(p)$.

3.2 Definition of ρ_0

In the section where Shapiro presents the refinement operator ρ_0, he claims that ρ_0 is weakly complete for any first order language $\mathcal{L}$. On the other hand, he redefines 'a sentence' to mean 'a reduced representative of the equivalence class of this sentence' [10, p27]. We assume that his intention was to define a refinement operator which is weakly complete for the set of reduced sentences. Even in this situation, however, it is not.

Definition 6. [10] If $p\theta = q$ and $|p| = |q|$ then θ *does not decrease* p, i.e., by choosing the right indices, we have $p = \{L_1, \ldots, L_m\}$, $q = \{M_1, \ldots, M_m\}$ and $L_i\theta = M_i$ for all i.
A literal L is *more general than* M *with respect to* p if there is a substitution θ such that $L\theta = M$ and $p\theta = p$.

If L is any literal that contains as arguments only distinct variables that do not occur in p, then we call L *most general with respect to* p.

Let M be a literal that is not most general w.r.t. a sentence p. Then we can always find a literal L that is more general than M w.r.t. p and that is most general w.r.t. p. In fact, we just need to replace all arguments of M by distinct variables that are not in p.

Definition 7. [10] A literal L is *most general with respect to* p *such that* $p \cup \{L\}$ *is reduced* if for any M such that M is properly more general than L w.r.t. p, $p \cup \{M\}$ is not reduced.

Example 4. Let $p = \{P(x, y)\}$, and let $L_1 = Q(u)$, $L_2 = \neg P(u, v)$, $L_3 = P(u, v)$ and $L_4 = P(u, x)$.

Then L_1, L_2 and L_3 are most general literals with respect to p, since they contain only distinct variables as arguments that do not occur in p. As can be verified, L_4 is a most general literal with respect to p such that $p \cup \{L_4\}$ is reduced.

Definition 8. [10] Let p be a reduced sentence of $\mathcal{L}$. Then $q \in \rho_0(p)$ when exactly one of the following holds:

ρ_0^1: $q = p\theta$, where $\theta = \{x/y\}$ does not decrease p and both variables x and y occur in p.

ρ_0^2: $q = p\theta$, where $\theta = \{x/f(y_1, \ldots, y_n)\}$ does not decrease p, f is an n-place function symbol, x occurs in p and all y_i's are distinct variables not in p.

ρ_0^3: $q = p \cup \{L\}$, where L is a most general literal with respect to p for which $p \cup \{L\}$ is reduced.

Although the definition of the refinement operator uses the concept of size, these ρ_0^1, ρ_0^2 and ρ_0^3 are concretely defined without it. Hence, we can begin our discussion of the completeness of ρ_0 without worrying about size. In Sect. 3.4 we will relate refinement operators with size.

3.3 Incompleteness of ρ_0

In this subsection we will show that Shapiro's Lemma 5.15 and 5.16 [10] are not correct, and hence that ρ_0 is not (strongly) complete. Shapiro never uses the terminology of strong completeness of ρ_0, Lemma 5.15 and 5.16 together, however, would imply it. We will also show that his Theorem 5.14 is not correct, and hence that ρ_0 is not even weakly complete.

Lemma 5.15 of Shapiro [10]. *Let p and q be two reduced sentences such that $p\theta = q$ for some substitution θ that does not decrease p. Then there is a finite total ρ_0-chain from p to q.*

Proof. [10] This lemma is a generalization of Theorem 4 in Reynolds' paper [9]. Examination of the proof of this Theorem shows that it can be applied to p and $p\theta$ to obtain a finite total chain. □

Reynolds has proved this theorem for atoms. Indeed if we do not restrict ourselves to reduced sentences and consider only non-decreasing substitutions, then we can consider a sentence as a generalized atom.

Example 5. Consider the following chains of sentences $p = p_0, p_1, p_2, p_3, p_4 = q$ and $p = p_0, p'_1, p'_2, p'_3, p_4 = q$. Since $p\theta = q$, and p and q are reduced, Shapiro claims that there is a finite total ρ_0-chain from p to q.

$$\begin{aligned}
&p = p_0 = \{P(a,w), P(x,b), P(c,y), P(z,d)\}, p_0 \text{ is reduced}\\
&p_1 = \{P(a,b), P(x,b), P(c,y), P(z,d)\} \sim p'_1 = \{P(a,b), P(c,y), P(z,d)\}\\
&p_2 = \{P(a,b), P(c,b), P(c,y), P(z,d)\} \sim p'_2 = \{P(a,b), P(c,b), P(z,d)\}\\
&p_3 = \{P(a,b), P(c,b), P(c,d), P(z,d)\} \sim p'_3 = \{P(a,b), P(c,b), P(c,d)\}\\
&q = p_4 = \{P(a,b), P(c,b), P(c,d), P(a,d)\}, p_4 \text{ is reduced}
\end{aligned}$$

Here we are facing a dilemma, either we allow non-reduced sentences (p_1, p_2, p_3) or we reduce after substitution and hence allow decreasing substitutions (e.g., $p'_1 \in \rho_0(p_0)$).

Lemma 9. *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. Let $p = \{P(a,w), P(x,b), P(c,y), P(z,d)\}$, and let $q = \{P(a,b), P(c,b), P(c,d), P(a,d)\}$ as above. We prove that none of ρ_0^1, ρ_0^2 and ρ_0^3 is a candidate to generate the successor of p in a ρ_0-chain from p to q.

ρ_0^1: All variables of p have to be substituted by constants to achieve q and all constants a, b, c, d are to be substituted only once. Unification of variables causes that a variable occurs at least twice. Therefore unification of variables is not applicable to get the next element in the chain.

ρ_0^2: All substitutions by constants in the intended place lead to non-reduced sentences. Every other introduction of a function-symbol can not be removed and does not lead to q.

ρ_0^3: Increasing the number of literals is not applicable, since only non-decreasing substitutions are allowed. Once increased, this number can never be reduced again.

□

Lemma 5.16 of Shapiro [10]. *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.*

Proof (Outline of [10]). The proof is by induction on the number of literals in the difference set $q - p$. If some of the literals in this difference set are removed from q, it is assumed that the resulting sentence still is reduced. □

This assumption, if p and q are reduced sentences such that $p \subseteq q$ then every sentence r such that $p \subseteq r \subseteq q$ is also reduced, is falsified by the following example:

Example 6. Consider the reduced sentences

$$p = \{P(x), \neg Q(x,a)\},$$
$$q = \{P(x), \neg Q(x,a), \neg Q(y,z), \neg Q(z,y)\}.$$

Then

$$r = \{P(x), \neg Q(x,a), \neg Q(y,z)\}$$

fulfills $p \subseteq r \subseteq q$ and r is not reduced.

Lemma 10. *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 (Outline). Consider the sentences p and q from the last example. The problem is to find a successor of p in a ρ_0-chain from p to q. Clearly, literals have to be added by ρ_0^3. Both literals in $q - p$ are most general with respect to p, but adding to p a literal of $q - p$ results in a non-reduced sentence. Formally, it can be proved that there is no candidate for a successor of p in a ρ_0-chain from p to q by the same technique as at Lemma 9. □

The problems of incompleteness are illustrated in Fig. 1. Non-reduced sentences are represented by filled circles, reduced sentences by open circles. Ovals represent equivalence classes under subsumption, and arrows represent a single ρ_0-application.

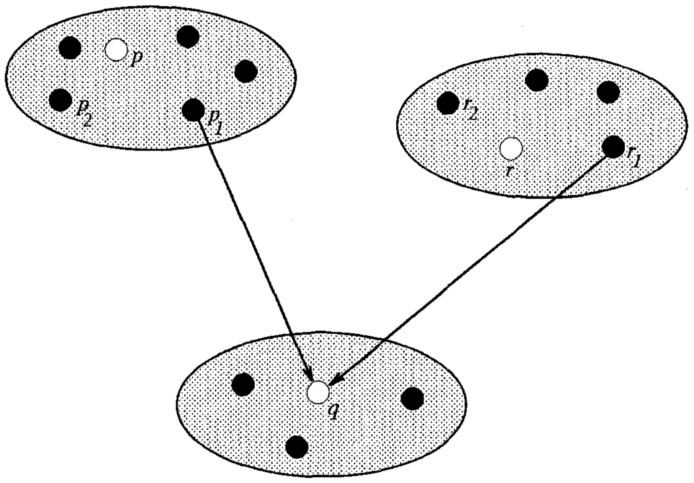

Fig. 1. Equivalence classes and refinements, a reduced sentence that can only be reached from non-reduced sentences

Shapiro's intention was that for every pair of reduced sentences p and q, satisfying $p \subseteq q$ or $p\theta = q$ where θ is non-decreasing, there is a chain of arrows from p to q where only reduced sentences are visited. Problems with ρ_0 arise when a reduced sentence (open circle) like q is only reachable from non-reduced sentences (filled circles) like p_1 and r_1.

Consider the following sentences:

$$\begin{aligned} p\ &= \{P(x,y), P(y,z), P(z,x)\} \\ p_1 &= \{P(u,w), P(w,v), P(x,y), P(y,z), P(z,x)\} \\ r\ &= \{P(v,w), P(w,v)\} \\ r_1 &= \{P(v,w), P(w,v), P(x,y), P(y,z)\} \\ q\ &= \{P(v,w), P(w,v), P(x,y), P(y,z), P(z,x)\} \end{aligned}$$

Both p and r properly subsume q, $p_1 \sim p$ and $r_1 \sim r$. If we apply the substitution $\{u/v\}$ to p_1 we get q. q can also be obtained from r_1 by adding $P(z,x)$, a most general literal w.r.t. r_1 such that $r_1 \cup \{P(z,x)\}$ is reduced.

It is proved in the working report [12] that there is no reduced sentence (open circle) such that q can be derived from it by ρ_0. Therefore, this sentence q is a counterexample of Shapiro's Theorem 5.14 which states that there is a ρ_0-chain from the empty sentence $\Box$ to every reduced sentence.

3.4 The Incorrectness of Using rsize

Shapiro has defined a refinement as follows: 'We say that q is a refinement of p if p implies q and size(q) > size(p)'. Although any complexity measure size that satisfies the requirement that for any fixed value k there are finitely many sentences of size $\leq k$ is allowed, in his concrete refinement operators Reynolds' [9] complexity measure (that we denote by rsize) for atoms is used. This complexity measure can be generalized to sentences by defining

rsize(p) = the number of symbol occurrences in p minus the number of distinct variables in p.

A nice property of Reynolds' rsize is that if an atomic sentence p subsumes an atomic sentence q ($q = p\theta$ for some substitution θ), then p properly subsumes q iff rsize(q) $>$ rsize(p). The idea behind Shapiro's refinement operators is also to find properly subsumed sentences. If a sentence p properly subsumes a sentence q, he intends q to be reachable from p by a chain of refinements found by a refinement operator. Since size can only return integer values and refinements are required to be of strictly larger size, if size(q) $-$ size(p) $= l$ then the chain of refinements from p to q is guaranteed not to exceed length l. As we know, for any sentence p, for example $\{P(x)\}$, we can find an infinite chain of refinements, $\{P(f(x))\}, \{P(f(f(x)))\}, \ldots$. If, however, only sentences of size $\leq k$ are allowed, no chain can exceed length k.

Shapiro's refinement operator ρ_1 is defined for atomic sentences by ρ_0^1 and ρ_0^2. In the atomic case, increasing rsize coincides with proper specialization and vice versa. Also, there are only finitely many atoms of rsize $\leq k$. Therefore, restricting the search of refinements to atomic sentences of rsize $\leq k$ and restricting to refinements of increasing rsize, all atomic sentences of rsize $\leq k$ can be found by ρ_1.

When we study Shapiro's refinement operator ρ_0, some problems arise with the use of Reynolds' rsize and subsumption as ordering.

Consider the following sentences:

$$\begin{array}{lll} p = \{P(x,y), P(y,x)\}, & \text{rsize}(p) & = 4 \\ q_1 = \{P(a,y), P(y,a)\}, & \text{rsize}(q_1) & = 5 \\ q_2 = \{P(x,x)\}, & \text{rsize}(q_2) & = 2 \end{array}$$

Then

$$\text{rsize}(p) < \text{rsize}(q_1) \text{ and } p \text{ subsumes } q_1 \ (p\theta = q_1 \text{ for } \theta = \{x/a\})$$
$$\text{rsize}(p) > \text{rsize}(q_2) \text{ and } p \text{ subsumes } q_2 \ (p\theta = q_2 \text{ for } \theta = \{y/x\})$$

Clearly, there is no direct relationship between subsumption and rsize.

To solve this problem, Shapiro restricts to so-called non-decreasing substitutions (the number of literals is not allowed to decrease). By this constraint, every sentence q found by applying the refinement operator ρ_0 to p satisfies rsize(p) $<$ rsize(q). However, although p properly subsumes q_2, in this approach it is no longer possible to construct a chain of refinements from p to q_2 because rsize(p) $>$ rsize(q). This also leads to incompleteness.

In Sect. 4.2 we will introduce a new complexity measure called newsize that avoids these problems.

4 A New Refinement Operator

When we define an equivalence relation on a set, it is usually required that the equivalence relation is compatible with the important operations on this set, i.e., operations on different members of the same equivalence class yield equivalent results. Unfortunately, this is not true for the equivalence relation defined by subsumption. Two subsumption equivalent sentences may not be equivalent after substitution. Therefore, if substitution is one of the operations, it is not enough to consider only the reduced representative of an equivalence class. This is an important reason why ρ_0 is not complete. Also, subsumption is not compatible with rsize, equivalent sentences may have a different rsize. Allowing non-decreasing substitutions only, guarantees that resulting sentences have a strictly larger rsize. However, if such a sentence is not reduced it will not be accepted. Even if reduction after a non-decresing substitution is allowed, its reduced equivalent can have a strictly smaller rsize and will not be regarded as a refinement.

In this section we will define a new refinement operator ρ_{r} which is complete for the set of all reduced sentences in $\mathcal{L}$. Although we will define a new complexity measure in Sect. 4.2, we are not going to use it explicitly in Sect. 4.1 when we define ρ_{r}.

4.1 A New Refinement Operator for Reduced Sentences

Before defining ρ_{r}, we revisit the problems and difficulties of Shapiro's ρ_0. For every two reduced sentences p and q such that p properly subsumes q, there should be ρ_{r}-chain from p to q.

o Consider the following two reduced sentences:

$p = \{P(x), \neg Q(x,a)\}$
$q = \{P(x), \neg Q(x,a), \neg Q(y,z), \neg Q(z,y)\}$

Here, p properly subsumes q since $p \subset q$. In Lemma 10 we have shown that there is no sentence r such that $r \in \rho_0(p)$ and $r \succeq q$. Adding to p one of the literals in $q - p$ results in a non-reduced sentence equivalent to p. Since q should be derivable from p, this suggests that sometimes more than one literal has to be added in one refinement step.

o Next, consider the following two reduced sentences:

$p = \{P(x,y), P(y,x)\}$
$q = \{P(x,x)\}$

Now, p properly subsumes q since $p\theta = q$ for $\theta = \{y/x\}$. Since $|p| > |q|$, we must allow decreasing substitutions.

In Sect. 2.2 we have presented an algorithm that generates all sentences p' with less than or equal to m literals that are equivalent to a given (reduced) sentence p. In the definition of ρ_{r} (ρ_{r}^1 and ρ_{r}^2) we will use $eq(p)$ to denote the set of all such p''s. Note that every sentence $p' \neq p$ in this set satisfies $p' = p \cup r$ for some set of literals r. Since $|r|$ can be larger than 1, we can use p' to solve the first problem presented above.

In one of the refinement steps of our new refinement operator (ρ_{r}^3), only one literal is added. For example, if $p = \{P(x,y)\}$, then $\neg P(u,v)$, $Q(u)$ and $\neg Q(u)$ can be added to p. We first give a lemma to illustrate its use.

Lemma 11. *Let p be a sentence, and let L be a most general literal with respect to p, i.e., L has only distinct variables as arguments that do not occur in p. Then the following two conditions are equivalent:*

1. *p properly subsumes $q = p \cup \{L\}$*
2. *For any literal M in p, L differs from M in either predicate name or sign.*

Proof. $1 \Rightarrow 2$: Assume that 2 does not hold. Then there is an M in p such that M and L have the same predicate name and sign. Let θ be defined on variables of L only, such that $L\theta = M$. Then $q\theta = (p \cup \{L\})\theta = p$. This means that q also subsumes p. Therefore, $p \sim q$.

$2 \Rightarrow 1$: $p \subset q$, so clearly p subsumes q. Assume that also q subsumes p, then for some θ, $q\theta \subseteq p$. But then also $L\theta \in p$, and $L\theta$ and L must have the same predicate name and sign. □

It is easy to verify that, if p is reduced and L satisfies the conditions of Lemma 11, then $q = p \cup \{L\}$ is also reduced.

Definition 12. Let p be a reduced sentence. Then $q \in \rho_{\mathrm{r}}(p)$ if q is reduced and one of the following conditions holds:

ρ_{r}^1: $p \succ q$ and there are $p' \in eq(p)$ and $q' \in eq(q)$ such that $q' = p'\theta$, where $\theta = \{x/y\}$ and both x and y occur in p'.

ρ_{r}^2: $p \succ q$ and there are $p' \in eq(p)$ and $q' \in eq(q)$ such that $q' = p'\theta$, where $\theta = \{x/f(y_1, \ldots, y_n)\}$, f is an n-place function symbol, x occurs in p', and all y_i's are distinct variables not occurring in p'.

ρ_{r}^3: $q = p \cup \{L\}$, where L is a most general literal such that for every literal $M \in p$, L differs from M in either predicate name or sign.

Theorem 13. *Let p and q be reduced sentences. If $p \succ q$, then there is a ρ_{r}-chain from p to q.*

Lemma 14. *Let $p, q \in \mathcal{L}$ be two reduced sentences such that p properly subsumes q and let $p' \in eq(p)$, $q' \in eq(q)$ satisfy $p'\theta = q'$. Then there is a $r \in \rho_{\mathrm{r}}(p)$ such that r subsumes q.*

Proof. Let $p' = p_0, p_1, \ldots, p_n = q'$ be a chain of sentences such that $p_i = p_{i-1}\theta_{i-1}$, $0 < i \leq n$, where every θ_i is a substitution as defined in ρ_{r}^1 or ρ_{r}^2. Reynolds [9, proof of Theorem 4] has shown how such a chain of substitutions can be constructed. Let p_k be the first p_i that is not equivalent to p. Since $p \succ q$ such a p_k exists. If we let r be the reduced equivalent of p_k, then $p_{k-1} \in eq(p)$, $p_k = p_{k-1}\theta_{k-1}$ and $r \in \rho_{\mathrm{r}}(p)$. Also, since r subsumes p_k, $p_k\theta_k \cdots \theta_{n-1} = p_n = q'$, and $q' \sim q$, r subsumes q. □

The following example illustrates the proof of Lemma 14.

Example 7. Let $p = \{P(a,w), P(x,b), P(c,y), P(z,d)\}$ and $q = \{P(a,b), P(c,b), P(c,d), P(a,d)\}$ as in the first example of Sect. 3.3. If $p' = p$ and $q' = q$ then $p'\theta = q'$ for $\theta = \{w/b, x/c, y/d, z/a\}$. θ can be split in $\theta_0 = \{w/b\}$,$\theta_1 = \{x/c)\}$,$\theta_2 = \{y/d\}$ and $\theta_3 = \{z/a\}$.

$p_0 = p' = p$

$p_1 = p_0\theta_0 = \{P(a,b), P(x,b), P(c,y), P(z,d)\}$

p_1 is not equivalent to p. The reduced equivalent r of p_1 is

$r = \{P(a,b), P(c,y), P(z,d)\}$,

and r is a member of $\rho_{\mathrm{r}}(p)$ that subsumes q.

Lemma 15. *Let $p, q \in \mathcal{L}$ be two reduced sentences such that p properly subsumes q and $p \subset q$. Then there is a refinement $r \in \rho_{\mathrm{r}}(p)$ such that r subsumes q.*

Proof. Let s be a maximal subset of $q - p$ such that $p \cup s \sim p$. That means, for every literal M in $(q-p)-s$, p properly subsumes $p \cup s \cup \{M\}$. Let L be a most general literal with respect to $p \cup s$ such that $L\theta = M$ for one of those literals.

If $p \cup s \cup \{L\}$ is not equivalent to $p \cup s$, then, by Lemma 11, L differs from all literals in $p \cup s$ in either predicate name or sign. L also has this property with respect to p. Let $r = p \cup \{L\}$. Then $r \in \rho_{\mathrm{r}}^3(p)$, and clearly r subsumes q.

Otherwise, $p' = p \cup s \cup \{L\}$ and $q' = p \cup s \cup \{M\}$ satisfies $p'\theta = q'$. Using Lemma 14 a sentence r can be found such that $r \in \rho_{\mathrm{r}}(p)$ and r subsumes q'. Since $q' \subseteq q$, r also subsumes q. □

The following examples illustrate the proof of Lemma 15.

Example 8. Let $p = \{P(x)\}$ and $q = \{P(x), \neg Q(a,x)\}$.
The only subset s of $q-p$ such that $p \cup s \sim p$ is $\{\}$, the empty set. $M = \neg Q(a,x)$ is the only literal in $(q-p)-s$. $L = \neg Q(y,z)$ is most general w.r.t. p and $L\theta = M$ for $\theta = \{y/a, z/x\}$. $p \cup \{L\}$ is reduced and

$r = \{P(x), \neg Q(y,z)\}$

satisfies $r \in \rho_{\mathrm{r}}(p)$ and r subsumes q.

Example 9. Let $p = \{P(x), \neg Q(x,a)\}$ and $q = \{P(x), \neg Q(x,a), \neg Q(y,z), \neg Q(z,y)\}$.
$s = \{\neg Q(y,z)\}$ is a maximal subset of $q - p$ such that $p \cup s \sim p$. Taking $M = \neg Q(z,y)$ we get $L = \neg Q(u,v)$ as a most general literal with respect to $p \cup s$. $p' = p \cup s \cup \{L\}$ is equivalent to p and p' properly subsumes $q' = p \cup s \cup \{M\}$. By Lemma 14 we can find a refinement r of p that subsumes q.

In Lemma 14 we have $p' = \{P(x), \neg Q(x,a), \neg Q(y,z), \neg Q(u,v)\}$, $q' = \{P(x), \neg Q(x,a), \neg Q(y,z), \neg Q(z,y)\}$ and $\theta = \{u/z, v/y\}$. θ can be split in $\theta_0 = \{u/z\}$ and $\theta_1 = \{v/y\}$. $p'\theta_0$ is not equivalent to p' so

$r = \{P(x), \neg Q(x,a), \neg Q(y,z), \neg Q(z,v)\}$

satisfies $r \in \rho_{\mathrm{r}}^1(p)$ and r subsumes q.

Proof of Theorem 13. For every pair of reduced sentences p and q such that p properly subsumes q, $p\theta \subseteq q$ for some θ, let s be the reduced equivalent of $p\theta$.

If p properly subsumes s then p and s satisfy the conditions of Lemma 14. Otherwise, $s \in eq(p)$, and s and q satisfy the conditions of Lemma 15.

In both cases the first element r of a ρ_{r}-chain from p to q can be found. We can complete a ρ_{r}-chain from p to q by repeatedly finding the first element in a chain from r to q. In Lemma 18 in the next subsection we prove that this chain is of finite length. $\square$

4.2 A New Complexity Measure

Shapiro has defined that q is a refinement of p if p implies q and size(p) < size(q). If we consider atoms and rsize, then q is a refinement of p is equivalent to p properly subsumes q. Suppose that rsize can indeed be generalized to sentences or that there is another kind of size which satisfies this property, then the use of size has the following advantages: it can restrict the search space of sentences by discarding sentences of size> k; it prohibits infinite chains of refinements, by demanding size to increase no cycles will occur; also, it can be used to ease proofs of completeness and finiteness.

However, in concrete examples, Shapiro uses rsize. In Sect. 3, we have shown that if p subsumes q, then rsize(p) can be smaller as well as larger than rsize(q). In the latter situation, q is not even regarded as a refinement of p, so there surely is no chain from p to q. Shapiro uses non-decreasing substitutions to prevent rsize to decrease. An argument in favor of this approach is that if $p\theta = q$ then there is always a subset p' of p such that $p'\theta = q$ and θ is non-decreasing w.r.t. p'. Since p' also subsumes q, and assuming that p' can be derived from the empty sentence, q can still be derivable via p'. This gives two problems. Firstly, as we have shown before, non-decreasing substitutions are not compatible with reduced sentences. Secondly, suppose that we already have some background information about the theory to be inferred, say that we know that a given sentence p subsumes a sentence q that has to be found. Then it is much more efficient to search for a chain from p to q than from the empty sentence to q. What we want is strong completeness, and non-decreasing substitutions do not fit in this approach.

All complications seem to be caused by adapting the refinement operator to the definition of size. If we know that we are looking for properly subsumed sentences, why don't we define refinement concretely by proper subsumption? This prevents cycles to occur. Size is then only needed to restrict the search space to a finite number of sentences.

Definition 16. Given a sentence $p \in \mathcal{L}$:
newsize(p) = (maxsize(p), $|p|$), where
maxsize(p) = $max\{\text{rsize}(L) | L \in p\}$ and $|p|$ is the number of literals in p.

It is easy to prove that if p subsumes q then maxsize(p) $\leq$ maxsize(q), from this it follows that if $p \sim q$ then maxsize(p) = maxsize(q). Also, if maxsize(p) >

maxsize(q) then p cannot subsume q. Contrary to rsize, maxsize has a natural relationship with subsumption.

Since the number of literals of rsize $\leq k$ is finite, the number of sentences satisfying newsize(p) $\leq (k, m)$ (maxsize(p) $\leq k$ and $|p| \leq m$) is also finite.

We now redefine the notions of refinement and refinement operator in the context of subsumption.

Definition 17. A sentence q is called a *refinement* of a sentence p iff p properly subsumes q. A *refinement operator* is a mapping from sentences to a subset of their refinements, such that for any $p \in \mathcal{L}$ and any $k, m > 0$ the set of all $\rho(p)(k, m)$, that is the set $\rho(p)$ restricted to sentences q such that newsize(q) $\leq (k, m)$, is computable.

In this definition 'computability' is guaranteed if every sentence that is involved to compute $\rho(p)$ satisfies newsize $\leq (k, m)$. We should add this condition to the definition of ρ_{r} (i.e., p, q, p' and q' have a newsize $\leq (k, m)$). This also restricts the sentences to be generated by the inverse reduction algorithm to sentences with less than or equal to m literals. Notice that if p properly subsumes q and both p and q satisfy newsize $\leq (k, m)$, then every related sentence to find r in Lemma 14 and 15 satisfies newsize $\leq (k, m)$.

Since every element in a chain of refinements properly subsumes its successor and subsumption is transitive, a chain cannot contain cycles.

These observations together with the following lemma imply that ρ_{r} is a refinement operator, complete for reduced sentences.

Lemma 18. *Let* $p_0, p_1, p_2 \ldots$ *be a* ρ_{r}*-chain, where* newsize(p_i) $\leq (k, m)$ *for every* p_i. *Then this chain is of finite length.*

Proof. There are finitely many sentences such that newsize(p) $\leq (k, m)$, so there are finitely many different sentences in every ρ_{r}-chain. Since for every two reduced sentences $p, q \in \mathcal{L}$ such that $q \in \rho_{\mathrm{r}}(p)$, $p \succ q$, no sentence can occur more than once in a ρ_{r}-chain. □

The properties of newsize and its strong relation with subsumption as described above, are a motivation for adopting it as a complexity measure to restrict the search space of refinements.

Using these new definitions, ρ_{r} is a refinement operator and it behaves like Shapiro thought ρ_0 would do, it is complete for reduced sentences.

5 Comparison with Lairds' ρ_{L}

Laird [2] has also defined a refinement operator, ρ_{L}. He uses a different notation to define his refinement operator. Instead of sentences $C \leftarrow D$ where C and D are sets of atoms, Laird considers clauses of a language $\mathcal{L}_{\mathrm{L}}$ where repetition of literals is allowed, substitutions are never decreasing, and also non-reduced clauses are allowed. The price we have to pay for this is the presence of many equivalent hypotheses in the search space.

Definition 19. [2] Let $p = C \leftarrow D$ be a clause in the language $\mathcal{L}_{\mathrm{L}}$. Then $q \in \rho_{\mathrm{L}}(p)$ 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, i.e., all variables in t are distinct and do not occur in p.
3. $q = C \vee L \leftarrow D$, where L is a most general atom.
4. $q = C \leftarrow D \wedge L$, where L is a most general atom.

In [11], 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 (clauses).

Laird does not give a proof of completeness of his version of ρ_0, instead he refers to the proof of Shapiro's Theorem 5.14. Moreover, Laird does not mention the difference between his and Shapiro's operator. In the working report [12] a proof of the completeness of ρ_{L} can be found.

Let $\mathcal{L}_{\mathrm{L}}$ be a language that contains one constant a, one 1-place function f, and one 2-place predicate P. Suppose at some moment, $P(a,a) \leftarrow$ is a clause that has to be refined. The set of one-step refinements will contain $P(a,a) \vee P(x,y) \leftarrow$, equivalent to $P(a,a) \leftarrow$. All one-step refinements of this clause are $P(a,a) \vee P(x,x) \leftarrow$, $P(a,a) \vee P(a,y) \leftarrow$, $P(a,a) \vee P(x,a) \leftarrow$, $P(a,a) \vee P(x,y) \vee P(v,w) \leftarrow$, $P(a,a) \vee P(f(z),y) \leftarrow$, $P(a,a) \vee P(x,f(z)) \leftarrow$ and $P(a,a) \vee P(x,y) \leftarrow P(v,w)$. The first four of these seven refinements are all equivalent to $P(a,a) \leftarrow$. In the next refinement steps this number will increase even faster. In fact we have a gigantic search space which contains a lot of equivalent clauses. All these (equivalent) clauses are regarded as different, all are kept in memory, and are subjected to refinement separately.

Laird has pointed out that in an implementation of ρ_{L}, variants of clauses can be treated as identical, and one can avoid generating variants of the same clause in computing $\rho_{\mathrm{L}}(p)$. In this way repeated literals and sentences that are equal up to renaming variables are prohibited. None of the sentences in our example would be avoided in this way.

In our approach, only properly subsumed reduced sentences are refinements. Of every equivalence class at most one representative is refined. When we refine a reduced sentence, inverse reduction is used. The time-complexity of inverse reduction, to generate equivalent sentences that are refined, is not very attractive. These sentences, however, can be thrown away immediately after refinement. In Lairds' approach these non-reduced equivalent sentences will also be generated when application of ρ_{L} results in non-reduced sentences. They will be kept in memory seperately until they are refined. Since only one sentence is refined at a time, our memory requirements are much smaller Lairds'. We therefore think, that with 'interesting problems', i.e., theories with a high complexity (measure), the extra time needed to compute refinements can be compensated for by the much smaller memory requirements since every equivalence class under subsumption has only one representative. This is subject of future research.

6 Refinement in a Wider Framework

In this section we will briefly introduce the connections between Shapiro's model inference and inverse resolution. It is said that model inference and inverse resolution reach the same destination from opposite directions: The first uses specialization, the second generalization. First we will invert the specialization operator defined in this paper to obtain a generalization operator, then we will relate it to inverse resolution and model inference using generalization.

6.1 Generalization Operators

Our refinement operator ρ_r can easily be inverted to a generalization operator δ_r. Given a reduced sentence q, a reduced sentence $p \in \delta_r(q)$ if $q \in \rho_r(p)$.

In [3] Ling has described so-called abstraction operators for atoms and Horn clauses. These operators are similar to the inverted versions of Shapiro's ρ_1 and ρ_2. They are used in a system called SIM which is roughly a system that works like Shapiro's MIS the other way around. Starting with some positive examples as hypotheses, generalizations are found by applying an abstraction operator to hypotheses if the hypotheses are too weak. An advantage of this specific to general approach over Shapiro's general to specific approach is that in SIM the positive examples play a more important role in determining the target theory.

δ_r can be viewed as a theoretically interesting generalization operator for the domain of reduced first order sentences, for example in a system like MIS.

6.2 Inverse Resolution

Given a logic program, we can use it to derive its logical consequences by using resolution. To reverse this process, we ask ourselves the following question: Given some positive examples that cannot be derived from the given program, how can we extend this program so that the new examples can be derived from it?

One possible answer is using Ling's sytem SIM, as described in the last subsection. Another approach is inverse resolution. In inverse resolution operators are used that invert one or more resolution steps. One of these operators is the so-called V-operator [5]: given two sentences p and r, a V-operator finds different sentences q such that r is a resolvent (or instance of a resolvent) of p and q.

For example, let $\mathcal{L}$ be a language that contains the predicate *even*, a constant symbol 0, and a function s (successor). Given the sentences

$p = \{even(0)\} \leftarrow$, and
$r = \{even(s(s(0)))\} \leftarrow$,

a V-operator should be able to derive

$q = \{even(s(s(x)))\} \leftarrow \{even(x)\}$.

In general there are many solutions for q. These depend on many choices. Some of the choices are: which literal L_1 in p is resolved with a literal L_2 in q; and what are the substitutions θ_1 and θ_2^{-1} such that $L_2 = L_1\theta_1\theta_2^{-1}$. However, Muggleton [4] has shown that for every choice of L_1 and θ_1 there is a unique least general solution q^*. He notes that every solution q subsumes q^*. In order to determine (all or some) solutions q when q^* is found, δ_{r}, the inverse of ρ_{r}, might prove useful. In the example above, δ_r could be used to derive q from the most specific solution

$q^* = \{even(s(s(0)))\} \leftarrow \{even(0)\}$.

For more detail on most specific V-operators we refer to [4].

6.3 Model Inference and Generalization

Shapiro's Model Inference System is concerned with finding a theory that is consistent with the given examples. Starting from the most general theory, a refinement operator is used to replace too strong hypotheses by logically weaker ones. This process can also be reversed. We do not know any learning system that uses generalization operators and restricts the search space to reduced sentences. We are thinking of a MIS- (or SIM-)like system that works with reduced sentences only. Starting with positive examples as hypothese, a generalization operator like δ_{r} is used to generalize too weak hypotheses like in Ling's MIS [3].

When the target theory contains recursive predicates or when auxiliary predicates occur in it, literals have to be added to hypotheses. We think that a V-operator is very useful for this part of the system. Guided by the positive examples and the predicates in the background theory, only the least general solutions of this V-operator will be accepted as new hypotheses. Too weak solutions can be generalized by applying δ_{r}. The formulation of an inductive inference algorithm that operates in this way is a subject of future research.

7 Conclusions

In this article we showed by concrete examples that ρ_0 is not complete for reduced sentences. The reasons behind this incompleteness were given by analyzing the special properties of subsumption and a complexity measure size. We noticed that it is most important that refinements of a sentence are properly subsumed by it. Size is used only to limit the number of refinements. Therefore, we redefined the notion of a refinement operator. Also, we defined a new refinement operator ρ_{r}, complete for reduced sentences, and a new complexity measure to limit the search space of refinements. In the end, we related our new refinement operator to generalization operators such as δ_{r} and the V-operator used in inverse resolution. In the future, we hope to use these operators to solve the model inference problem by generalization.

Acknowledgements. We thank Cor Bioch for the suggestions that lead to the subject of this article. We thank Leon van der Torre and Arie de Bruin for the discussions and support. Also, we thank Mark Vermeer for reading and commenting this article.

References

1. M. Kirschenbaum and L.S. Sterling. Refinement strategies for inductive learning of simple prolog programs. In *Proceedings of IJCAI-91*, Sydney, Australia, 1991. Morgan Kaufmann.
2. P.D. Laird. *Learning from Good and Bad Data.* Kluwer Academic Publishers, 1988.
3. X. Ling. Inductive learning from good examples. In *Proceedings of IJCAI-91*, Sydney, Australia, 1991. Morgan Kaufmann.
4. S.H. Muggleton. Inductive logic programming. In *First Conference on Algorithmic Learning Theory*, Ohmsha, Tokyo, 1990. Invited paper.
5. S.H. Muggleton and W. Buntine. Machine invention of first-order predicates by inverting resolution. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 339–352. Kaufmann, 1988.
6. T. Nibblet. A study of generalisation in logic programs. In *Proceedings of EWSL-88*, London, 1988. Pittman.
7. W.E. Nijenhuis and C. Witteveen. Constructive identification with Poole's default logic. Technical Report 90-96, Faculty of Technical Mathematics and Informatics, TU-Delft, 1990.
8. G.D. Plotkin. A Note on Inductive Generalization. In *Machine Intelligence 5*, pages 153–163. Edinburgh University Press, Edinburgh, 1970.
9. 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.
10. E.Y. Shapiro. Inductive Inference of Theories from Facts. Technical Report 624, Department of Computer Science, Yale University, New Haven. CT., 1981.
11. E.Y. Shapiro. *Algorithmic program debugging.* MIT Press, 1983.
12. P.R.J. Van der Laag. A Most General Refinement Operator for Reduced Sentences. Technical Report EUR-CS-92-03, Erasmus University Rotterdam, Dept. of Computer Science, June 1992.