# On the Completion of Inverse Entailment for Mutual Recursion and its Application to Self Recursion

Koichi FURUKAWA[1,2] and Tomonobu OZAKI[1]

[1] Graduate School of Media and Governance, Keio University
5322 Endo, Fujisawa, Kanagawa, 252, Japan
{furukawa,tozaki}@sfc.keio.ac.jp
[2] Department of Computer Science, The University of York
Heslington, York, YO10 5DD United Kingdom
furukawa@cs.york.ac.uk

**Abstract.** Muggleton gave an algorithm to augment his original MSH to make it complete. However his algorithm failed to keep the augmented MSH sound. On the other hand, we gave another method to realize completion of the MSH for definite clauses background knowledge, ground-atom positive examples and mutual recursive hypotheses in our previous study. The algorithm aimed to achieve both completeness and soundness of the MSH. However, we found that the algorithm has four drawbacks: 1. it fails to find an appropriate literal to be added when more than one mutual recursions are required in proving the given positive example; 2. it cannot treat cases other than linear recursive hypotheses; 3. it includes an entailment test in the algorithm which prevents the algorithm terminate; and 4. it excludes the treatment of self recursion. In this paper, we utilize the computation of bottom clauses in Muggleton [4] and therefore we assume that hypotheses belong to depth-bounded definite moded language $\mathcal{L}_i(M)$ and resolution steps to obtain refutation proof from background knowledge and the bottom clause is bounded to $h$. We pose two further restrictions to hypotheses class: first, we deal only the case that the target predicate is the same as examples' one; second, we set an upper limit to the size of terms appearing in hypotheses. Under those conditions, we propose a new algorithm for definite clauses background knowledge, ground-atom positive examples and mutual recursive hypotheses which avoids the above four defects altogether. It computes a set of maximally specific hypotheses (MaxSHs) at least one of which is always subsumed by any sound hypothesis.

## 1 Introduction

Inductive logic programming based on inverse entailment is noteworthy in a sense that it is based on an inverse operation of entailment which plays a central role in deductive reasoning. However, Yamamoto[7] pointed out that the most specific hypothesis (MSH) obtained by a naive algorithm (*Algorithm IE*) proposed by

Muggleton [4] is incomplete in a sense that some correct hypotheses cannot entail it.

Yamamoto [8] defined the problem class in which the *Algorithm IE* works properly in terms of $SB$-Resolution by Plotkin[6]. It says that it computes the right MSH when we do not need more than one occurrences of the hypothesis in the proof tree of a positive example to be generalized.

Furthermore, Muggleton [5] succeeded in obtaining a complete MSH by extending the literal set computed by *Algorithm IE*. Furukawa [3] also proposed a method to extend the literal set to make it complete. However it was shown that they have defects respectively; the former loses its soundness in a sense that the obtained literal set may produce hypotheses which do not entail the given positive example together with background knowledge, and the latter is defective in four points: first, it cannot discover a necessary literal when more than one mutual recursions appear in a proof of the positive example, second, it cannot deal with recursive programs other than linear recursion, third, it includes an entailment test in the algorithm which prevents the algorithm terminate, and fourth, it cannot deal with self recursion.

In this paper, we utilize the computation of bottom clauses in Muggleton [4] and therefore we assume that hypotheses belong to depth-bounded definite moded language $\mathcal{L}_i(M)$ and resolution steps to obtain refutation proof from background knowledge and the bottom clause is bounded to $h$ in order to avoid the termination problem.

We pose two further restrictions to hypotheses class: first, we deal only the case that the target predicate is the same as examples' one; second, we set an upper limit to the size of terms appearing in hypotheses. Cases when learned concepts and examples are different are treated differently and reported in Furukawa [2].

Then we propose an improved version of an algorithm proposed by [3] which solves the above first two problems by inventing a new method including least generalization of two different occurrences of the hypothesis to be obtained. Our algorithm computes a set of maximally specific hypotheses one of which is always subsumed by any sound hypothesis. We also extend our algorithm to treat nonlinear recursion. This extension can be achieved easily by adding further literals to initial hypothesis. For the treatment of the last self recursion problem, we will show that our algorithm can be applied by transforming given problems to mutual recursive problems.

## 2 Preliminaries

In this section, we introduce definitions of terms used later in this paper.

**Logical setting of ILP**
Basic components of ILP in the logical setting are background knowledge, a positive example, and hypotheses generated from them. Background knowledge can be arbitrary clausal form; a positive example and any generated hypothesis

are both a single clause. In actual situation, there are more than one positive example. One example is chosen and is used to generate a hypothesis and set-cover algorithm is applied to enumerate those examples covered by the obtained hypothesis. Then they are removed from the example set and the algorithm repeats the same process to the rest of the examples. Then, the following relations hold:

1. For any positive example $E$ and background knowledge $B$, $B \not\models E$ holds.
2. Any hypothesis $H$ satisfies $B \wedge H \models E$.

The statement 1. claims that a positive example $E$ cannot be derived only from background knowledge $B$ and 2. claims that that $B$ together with $H$ can explain $E$.

In this paper, we restrict positive examples to ground atoms and background knowledge to a set of definite clauses.

**Definition 1.** *A hypotheses generator* **G** *is* **sound** *if it always produces hypotheses satisfying the above condition* 2. **G** *is* **complete** *if it generates all hypotheses satisfying the condition* 2.

Note that we consider only positive (but not negative) examples for defining soundness and completeness in this paper.

**Definition 2.** *The second formula in the logical setting of ILP can be transformed to*

$$B \wedge \neg E \models \neg H.$$

*This formula is called as* **inverse entailment** *of the original formula.*

The inverse entailment expression suggests a possibility that a hypothesis can be obtained by utilizing deductive computation from background knowledge and negation of a positive example.

**Definition 3.** *We denote the conjunction of all (positive and negative) ground literals which are true in all Herbrand models of* $B \wedge \neg E$ *by* $\neg bot(B, E)$.

**Definition 4.** *A predicate which represents the target concept of learning is called* **a target predicate.**

We restrict our learning tasks to those when concepts or predicate symbols to be learned are the same as examples' ones. Therefore, examples are defined by a target predicate. We call this setting as **normal setting** compared with the case for learning a different target concept from examples, to which we refer as **abductive setting**.

**Definition 5.** *Let* $B$ *be background knowledge,* $E$ *be a positive example and* $H_0$ *be a finite set of single clause hypotheses* $H_{0_i}$. $H_0$ *is a* **maximally specific hypotheses set (MaxSHs) with respect to** $B$ **and** $E$ *iff*

1. *for every* $H_{0_i} \in H_0$, $B \wedge H_{0_i} \models E$ *and there exists no other hypothesis* $H'$ *such that* $H_{0_i}$ *subsumes* $H'$ *and* $B \wedge H' \models E$, *and*
2. *for any hypothesis* $H$ *such that* $B \wedge H \models E$, *there is an* $H_{0_i} \in H_0$ *such that* $H$ *subsumes* $H_{0_i}$.

*Each* $H_{0_i} \in H_0$ *is called a maximally specific hypothesis.*

Since we want to exclude cases when entailment tests do not terminate, we need to define $h$-easy models:

**Definition 6.** *A model is* $h$**-easy** *if there exists such a theory* $T$ *such that the truth value of almost all atoms* $\alpha_i$ *(except finite atoms) in the model can be deductively determined from* $T$ *by the time less than or equal to* $h(i)$ *for some total recursive function* $h$.

In the above definition, a theory $T$ corresponds to the union of background knowledge and a hypothesis in this paper.

In case of Progol, the above total recursive function is set to a constant function whose value is $h$. In this paper, we also adopt this choice.

Lastly, we define a hypothesis having only terms whose size is less than a given size $N$ where the term size is defined as a total number of functors in the term including constant symbols and variables (the size of any variable is 1).

**Definition 7.** *A hypothesis clause is* $N$**-bounded** *if it has only terms whose size is less than a given bound* $N$.

Since we are interested in obtaining only a compressed hypothesis whose length is smaller than the total length of covered positive examples, we define the above bound simply as a product of a maximum term length appeared in $bot(B, E)$ and the number of terms in all examples and we refer this bound as **term size bound** for a given learning problem.

Since $bot(B, E)$ is a finite set under the assumption of depth-bound definite moded language $\mathcal{L}_i(M)$ and resolution steps bound $h$ [4], the term size bound becomes finite.

## 3 Model Constraining Clauses

In this section, we define model constraining clauses (MCC), reduced-MCC and its associated schema. We then introduce two quasi-MSHs in terms of reduced-MCC schema. We also give related lemmas which play essential roles for building a complete algorithm for computing the MaxSHs. In this section, we assume that a sound hypothesis has a form of linear mutual recursive clause which means that it will form linear recursion together with background clauses. More general case will be treated later in section 4.2.

**Definition 8.** *We define a* **model constraining clause (MCC)** *inductively.*

1. *A clause in the background knowledge $B$ containing a negative disjunct of the target predicate is a model constraining clause.*
2. *A clause obtained by applying a folding operation with respect to a model constraining clause to an appropriate clause in $B$ is a model constraining clause.*
3. *Only clauses obtained by 1. and 2. are model constraining clauses.*

*A* **model constraining head literal (MC-head)** *is the head literal in the MCC. We express a MC-head as $Head\{\mathbf{X}\}$ where $\mathbf{X}$ is a vector of variables. A* **model constraining body literal (MC-body)** *is the body literal of the target predicate in the MCC. Like MC-head, we express a MC-body as $Body\{\mathbf{Y}\}$ where $\mathbf{Y}$ is a vector of variables.*

The above algorithm computes folding closure of the initial MCC obtained in Step 1. The reason why we need to compute folding closure will be stated later. We then define a reduced-MCC.

**Definition 9.** *A* **reduced-MCC** *is a ground instantiation of a clause $Head\{\mathbf{X}\} \leftarrow Body\{\mathbf{Y}\}$ which is entailed by background knowledge and a given hypothesis clause.*

Note that a reduced-MCC contains only the MC-body in its body. Other literals which may have appeared in the original MCC are thrown away.

Note also that the above MC-head and MC-body need not be taken from the same MCC. This is the reason why we introduce different vectors of variables $\mathbf{X}$ and $\mathbf{Y}$. Later we will see the case where a reduced-MCC can be obtained by partially evaluating the original MCC possibly containing another or more occurrences of the same MCC during the partial evaluation and the resulting body literal may come from different occurrence than the original MCC.

A reduced-MCC constrains possible models of $B \wedge \neg E$. We state this fact as the following lemma.

**Lemma 1.** *Let $Head\{\mathbf{a_1}\} \leftarrow Body\{\mathbf{a_2}\}$ be a reduced-MCC under background knowledge and a given hypothesis clause. Then, any Herbrand model of $B \wedge \neg E$ satisfies one of the following conditions: (1) it does not contain $Body\{\mathbf{a_2}\}$ or (2) it contains both $Head\{\mathbf{a_1}\}$ and $Body\{\mathbf{a_2}\}$.*

**Proof** Since the implication $Head\{\mathbf{a_1}\} \leftarrow Body\{\mathbf{a_2}\}$ is entailed by background knowledge and a given hypothesis clause, the case where $Body\{\mathbf{a_2}\}$ is true whereas $Head\{\mathbf{a_1}\}$ is false can be excluded. Therefore only remaining cases are the above two ones. □

This is the reason why we call this kind of clauses as (reduced) model constraining clauses. We divide the models of $B \wedge \neg E$ into two kinds by this reduced-MCC as described above. For the later discussion, we assume that we have only one reduced model constraining clause to make the problem simple. The way how to extend the following arguments to multiple reduced model constraining clauses case will be stated later in this paper.

We build a reduced-MCC from a corresponding MCC during the construction of MaxSHs in our new algorithm introduced later. It could be obtained by partially evaluating the instantiation of the MC-head using background knowledge and the current hypothesis. Note that it should be obtained only after computing an appropriate current hypothesis. This means that we cannot use it in computing the hypothesis itself. To break this causality loop, we need to variablize the reduced-MCC using Skolem constants (since those variables stands for some constants, logical variables should not be used here) and apply generate-and-test strategy to find correct ground substitutions for those Skolem constants. For this purpose, we then introduce a **reduced-MCC schema** as follows:

**Definition 10.** *Let* $Head\{\mathbf{a_1}\} \leftarrow Body\{\mathbf{a_2}\}$ *be a reduced-MCC. A* **reduced-MCC schema** *can be obtained by substituting constants* $\mathbf{a_1}$ *and* $\mathbf{a_2}$ *to* $\mathbf{c_1}$ *and* $\mathbf{c_2}$*, respectively, where* $\mathbf{c_1}$ *and* $\mathbf{c_2}$ *are Skolem constants corresponding to* $\mathbf{a_1}$ *and* $\mathbf{a_2}$*, respectively.*

Note that this substitution is not the one appearing in usual Skolemization process since it substitutes constants to Skolem constants.

**Definition 11.** *Let* $B$ *is background knowledge,* $C = Head\{\mathbf{c_1}\} \leftarrow Body\{\mathbf{c_2}\}$ *be the only reduced-MCC schema obtained from* $B$*, and* $E$ *be a positive example. Then, two* **quasi-MSH***s* $bot^+(B,E)$ *and* $bot^-(B,E)$ *are defined by:*

$$bot^+(B,E) = bot(B,E) \cup \{\neg Head(\mathbf{c_1})\}$$
$$bot^-(B,E) = (bot(B,E) \backslash \{E\}) \cup \{Body(\mathbf{c_2})\}$$

In Lemma 1, we showed that every Herbrand model of $B \wedge \neg E$ either does not contain $Body\{\mathbf{a_2}\}$ or contains both $Head\{\mathbf{a_1}\}$ and $Body\{\mathbf{a_2}\}$. $bot^+(B,E)$ corresponds to the latter and $bot^-(B,E)$ to the former. We omitted $\neg Body(\mathbf{c_2})$ in the definition of $bot^+(B,E)$. Similarly, we omitted $\{E\}$ in the definition of $bot^-(B,E)$. These omissions are guaranteed by the following lemma:

**Lemma 2.** *Let a positive example be* $E$*, background knowledge be* $B$*, and a sound hypothesis having the same target predicate as* $E$ *be* $H$*. If* $H$ *does not entail* $bot(B,E)$*, it should contain a generalization of a MC-body and of a MC-head in the consequent and in the antecedent, respectively.*

**Proof**
[MC-body in the consequent] Since $H$ does not entail $bot(B,E)$, there must be at least twice occurrence of $H$ in the proof of $E$. Since the instantiation of an occurrence other than the first call by $E$ should be different from $E$, it should be called by a $MCC$ which contains a predicate having the same target predicate, which is in fact the MC-body of the MCC.
[MC-head in the antecedent] This follows as a direct conclusion of *Theorem 3.1* in [3]. □

Actually we use folding closure in the latter proof. For the detail, see [3].

In our algorithm, we will obtain an appropriate hypothesis by taking lgg of $bot^+(B,E)$ and $bot^-(B,E)$. This lemma implies that we can omit $\neg Body(\mathbf{c_2})$ in

the definition of $bot^+(B,E)$ and similarly $E$ in the definition of $bot^-(B,E)$ since we do not generalize them in the lgg operation.

Examples of reduced-MCC, $bot^+(B,E)$ and $bot^-(B,E)$ are shown in the next section.

We will show the relationship between $\{bot^+(B,E), bot^-(B,E)\}$ and occurrences of the hypothesis in the proof of $E$:

**Lemma 3.** *$bot^+(B,E)$ and $bot^-(B,E)$ correspond to the first and other occurrence of the hypothesis in the proof of $E$, respectively.*

**Proof**

Since the only positive literal in $bot^+(B,E)$ is $E$, this corresponds to the first call (note that $E$ is omitted from $bot^-(B,E)$). Similarly, the only positive literal in $bot^-(B,E)$ is $Body(\mathbf{c_2})$ and this is called from the reduced-MCC $Head\{\mathbf{a_1}\} \leftarrow Body\{\mathbf{a_2}\}$ (note that the Skolem constant $c_2$ should be properly substituted to the right value $a_2$ after proper hypothesis has been created). □

## 4 A Complete Algorithm for Computing MaxSHs

### 4.1 Linear mutual recursive case

First, we consider the case when hypotheses are linear mutual recursive programs. We treat the more general case in the next subsection.

We call the algorithm for linear mutual recursion as **Augmented Inverse Entailment algorithm for Linear Mutual Recursion, AIELMR**.

Before the description of the algorithm, we will give a sketch of a proof tree of a given example $E$ corresponding to linear recursion case in Fig.1.

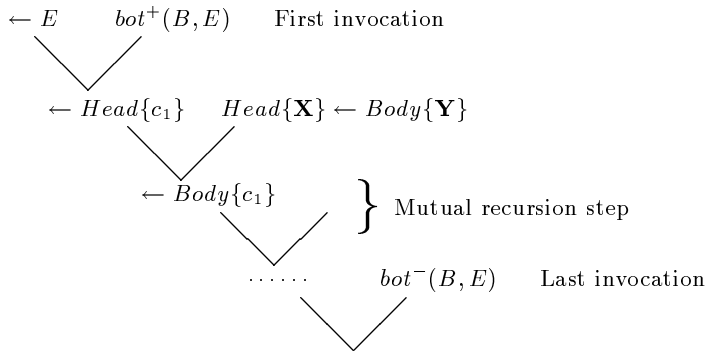


**Fig. 1.** A proof tree corresponding to linear recursion.

The following algorithm computes MaxSHs at least one of which is always subsumed by any sound hypothesis.

**Algorithm** *AIELMR*

1. Let $B$ be an arbitrary set of definite clauses, $C = Head\{\mathbf{c_1}\} \leftarrow Body\{\mathbf{c_2}\}$ be an associated reduced-MCC schema, $E$ be ground-atom positive example, $N$ be term size bound for the given learning problem and $MaxSHs$ be a variable representing MaxSHs initially set to $\{bot(B, E)\}$.
2. First, we compute two quasi-MSHs:

$$bot^+(B, E) = bot(B, E) \cup \{\neg Head\{\mathbf{c_1}\}\}$$
$$bot^-(B, E) = (bot(B, E) \backslash \{E\}) \cup \{Body\{\mathbf{c_2}\}\}$$

3. Repeat the following computation while the sizes of generated Herbrand universe elements are both less than or equal to $2N$:
   (a) Substitute generated Herbrand universe elements $a_1$ and $a_2$ to the Skolem constants vector $c_1$ and $c_2$, respectively.
   (b) Compute the least generalization of the $bot^+(B, E)$ and the $bot^-(B, E)$, which we denote as $lgg$.
   (c) Assume $Body\{\mathbf{a_2}\}$ holds and test whether $Head\{\mathbf{a_1}\} \leftarrow$ can be derived from $B$ and $lgg$ or not. If not, do nothing. Else add $lgg$ to $MaxSHs$.

Note that we initialize $MaxSHs$ to $\{bot(B, E)\}$ for the case where no solutions will be found for mutual recursion.

*Example 1.*

$$\begin{cases} E_1 = odd(s^5(0)). \\ B_1 = \{even(s(X)) \leftarrow odd(X).\ even(0).\} \end{cases}$$

Then, the reduced-MCC schema $RMCC_1$, $bot(B_1, E_1)$, $bot^+(B_1, E_1)$ and $bot^-(B_1, E_1)$ become

$$\begin{aligned} RMCC_1 &= even(s(c_1)) \leftarrow odd(c_2) \\ bot(B_1, E_1) &= \{odd(s^5(0)), \neg even(0)\} \\ bot^+(B_1, E_1) &= \{odd(s^5(0)), \neg even(0), \neg even(s(c_1))\} \\ bot^-(B_1, E_1) &= \{odd(c_2), \neg even(0)\} \end{aligned}$$

respectively. After substituting $c_1$ to $s^3(0)$ and $c_2$ to $s(0)$, we compute the least generalization of $bot^+(B_1, E_1)$ and $bot^-(B_1, E_1)$, which we denote as $lgg_1$:

$$lgg_1 = \{odd(s(X)), \neg even(X), \neg even(0))\}.$$

Finally, we try to prove $even(s^4(0)) \leftarrow$ from $B_1$ and $lgg_1$ assuming that $odd(s(0))$ holds. We can prove easily that this holds. Therefore we add $lgg_1$ to $MaxSHs$. Since there are no further solutions, this is our final result.

*Example 2.*

$$\begin{cases} E_2 = anc(a,e). \\ B_2 = \{p(a,b).p(b,c).p(c,d).p(d,e). \\ \qquad\quad anc1(X,Y) \leftarrow p(X,Y). \\ \qquad\quad anc1(X,Y) \leftarrow anc(X,Y).\} \end{cases}$$

Then,$RMCC_2$, $bot(B_2,E_2)$, $bot^+(B_2,E_2)$ and $bot^-(B_2,E_2)$ become

$$\begin{array}{ll} RMCC_2 & = anc1(b_X,b_Y) \leftarrow anc(c_X,c_Y) \\ bot(B_2,E_2) & = \{\ anc(a,e), \neg p(a,b), \neg p(b,c), \neg p(c,d), \neg p(d,e), \\ & \quad \neg anc1(a,b), \neg anc1(b,c), \neg anc1(c,d), \neg anc1(d,e)\ \} \\ bot^+(B_2,E_2) & = \{\ anc(a,e), \neg p(a,b), \neg p(b,c), \neg p(c,d), \neg p(d,e), \\ & \quad \neg anc1(a,b), \neg anc1(b,c), \neg anc1(c,d), \neg anc1(d,e), \\ & \quad \neg anc1(b_X,b_Y)\ \} \\ bot^-(B_2,E_2) & = \{\ anc(c_X,c_Y), \neg p(a,b), \neg p(b,c), \neg p(c,d), \neg p(d,e), \\ & \quad \neg anc1(a,b), \neg anc1(b,c), \neg anc1(c,d), \neg anc1(d,e)\ \} \end{array}$$

respectively. After substituting $b_X$ to $b$, $b_Y$ to $e$, $c_X$ to $c$ and $c_Y$ to $e$, we compute the least generalization, which we denote as $lgg_2$, of $bot^+(B_2,E_2)$ and $bot^-(B_2,E_2)$ and obtain

$$\begin{array}{l} lgg_2 = \{anc(X,e),\ \neg p(X,Y),\ \neg anc1(Y,e),\ \neg anc1(X,Y), \\ \qquad\qquad\qquad\qquad\qquad\qquad \neg p(Y,Z),\ \neg anc1(Y,Z),\ \cdots\ \} \end{array}$$

Finally, we test whether $anc1(b,e) \leftarrow$ can be derived from $B_2$ and $lgg_2$ or not, assuming that $anc(c,e)$ holds. We can prove easily that this holds. Therefore we obtain $lgg_2$ as one possible solution. In fact, since we cannot find any other solutions, this is our final result.

Now we prove the soundness and the completeness of the Algorithm *AIELMR*:

**Theorem 1.** *The algorithm AIELMR is sound and complete for linear mutual recursive case under the normal setting whose hypotheses are restricted to h-easy classes where the function h is a constant function having the value h, and whose terms are bounded by the term size bound for the give problem.*

**Proof**

[Soundness] We show the soundness by proving the goal $\leftarrow E$. Let $H$ be a hypothesis obtained by the algorithm *AIELMR*. Since the first invocation of $H$ corresponds to $bot^+(B,E)$ by Lemma 3, the resolvent $R$ includes $\neg Head\{\mathbf{a_1}\}$. Since all other body literals come from $bot(B,E)$, they can be resolved by the clauses in $B \cup \{\neg E\}$. Now, we show that $RMCC : Head\{\mathbf{a_1}\} \leftarrow Body\{\mathbf{a_2}\}$ is a reduced-MCC. This is obvious since the truthness of this formula is proved in the algorithm. Then, we can resolve $R$ and $RMCC$ to obtain $\neg Body\{\mathbf{a_2}\}$. Then it can be resolved by $H$ again, and the invocation of this time corresponds

to $bot^-(B, E)$ and therefore the resolvent only contains literals from $bot(B, E)$, which can be resolved by the clauses in $B \cup \{\neg E\}$.

[Completeness] First, we show that the algorithm terminates. The termination proof contains two parts. The first one is to show that we can use the term size bound in restricting the generation of Herbrand universe elements in step 3. Since the counter part of the least generalization for $\neg Head\{\mathbf{c_1}\}$ in $bot^+(B, E)$ are found only in the negative literals in $bot(B, E)$, the maximum size of the terms in the result of the least generalization exceeds $2N - N = N$ when we generate terms having size $2N$ or more. Similarly, we can show that the result of the least generalization corresponding to $Body\{\mathbf{c_2}\}$ exceeds $N$ when we generate terms having size $2N$ or more. Since we do not need to generate hypotheses having terms whose size are bigger than $N$, we can stop the generation of the Herbrand Universe elements.

The second one is to show that the entailment tests terminate. Since we limit our hypotheses classes only $h$-easy models where the total recursive function is a constant function, we can guarantee that the entailment tests terminate in almost all cases.

Next, we show the completeness of the algorithm. An important observation is that when we use the newly introduced reduced-MCC in the proof of the given example, the hypothesis $H$ occurs only twice. This is obvious from the proof trace of $\leftarrow E$ given in the soundness proof. As a result, the hypothesis $H$ is invoked only twice. Those invocations are the above two occurrences. The algorithm *AIELMR* computes all such hypotheses under the restrictions stated above. Since the reduced-MCC is the result of partially evaluating the instantiation of the MC-head using background knowledge and the current hypothesis, any multiple occurrences of the hypothesis in the proof of $\leftarrow E$ is absorbed in it except two occurrences, the first one and the last one. Since the algorithm *AIELMR* computes all hypotheses invoked twice, we succeed in making it complete. □

When we have more than one reduced-MCC, we can simply apply the algorithm *AIELMR* to each of the reduced-MCCs and compute the union of the hypothesis set.

### 4.2 Extension of the algorithm to non-linear cases

In order to extend the algorithm to, say, a double recursive case, we need to add one more negative example $\neg Head\{\mathbf{c_{12}}\}$ to the $bot^(+)(B, E)$ and expand it to

$$bot^+ = bot(B, E) \cup \{\neg Head\{\mathbf{c_{11}}\}, \neg Head\{\mathbf{c_{12}}\}\}.$$

This expansion can be performed dynamically when the entailment test $3(c)$ fails. However, there arises a new problem when to stop this new introduction of a mutual recursive invocation literal. Since we take the least generalization of $bot^+(B, E)$ and $bot^-(B, E)$, we need different counter parts against these newly introduced negative literals respectively. We can make the bound as the number

of occurrences of negative literals in $bot(B,E)$ having the same predicate as $Head\{\mathbf{c_{1i}}\}$.

*Example 3.* Let

$$\begin{cases} E_3 = fib(s^4(0), s^5(0)). \\ B_3 = \{\ fib_1(0, s(0)).\ fib_1(s(0), s(0)).\ fib_1(X,Y) \leftarrow fib(X,Y). \end{cases}$$

Then, $bot(B_3,E_3), bot^+(B_3,E_3), bot^-(B_3,E_3)$ become as follows:

$$\begin{aligned} bot(B_3,E_3) &= \{\ fib(s^4(0),s^5(0)),\ \neg fib_1(0,s(0)),\ \neg fib_1(s(0),s(0)), \\ &\quad \neg add(0,s(0),s(0)),\ \cdots,\ \neg add(s(0),s(0),s^2(0)), \\ &\quad \neg add(s(0),s^2(0),s^3(0)),\ \neg add(s^2(0),s^3(0),s^5(0)),\ \cdots\ \} \\ bot^+(B_3,E_3) &= \{\ fib(s^4(0),s^5(0)),\ \neg fib_1(0,s(0)),\ \neg fib_1(s(0),s(0)), \\ &\quad \neg add(0,s(0),s(0)),\ \cdots,\ \neg add(s(0),s(0),s^2(0)), \\ &\quad \neg add(s(0),s^2(0),s^3(0)),\ \neg add(s^2(0),s^3(0),s^5(0)),\ \cdots, \\ &\quad \neg fib_1(c_X,c_Y),\ \neg fib_1(c_Z,c_U)\ \} \\ bot^-(B_3,E_3) &= \{\ fib(c_V,c_W),\ \neg fib_1(0,s(0)),\ \neg fib_1(s(0),s(0)), \\ &\quad \neg add(0,s(0),s(0)),\ \cdots,\ \neg add(s(0),s(0),s^2(0)), \\ &\quad \neg add(s(0),s^2(0),s^3(0)),\ \neg add(s^2(0),s^3(0),s^5(0)),\ \cdots\ \} \end{aligned}$$

Now, we substitute $c_X = s^2(0), c_Y = s^2(0), c_Z = s^3(0), c_U = s^3(0), c_V = s^2(0), c_W = s^2(0)$. Then, we compute the least generalization, which we denote $lgg_3$, of $bot^+(B_3,E_3)$ and $bot^-(B_3,E_3)$ and obtain

$$\begin{aligned} lgg_3 = \{&fib(s^2(X),s^2(Y)),\ \neg fib_1(s(X),s(X)),\ \neg fib1(X,s(Z)), \\ &\neg add(s(Z),s(X),s^2(Y)),\ \cdots\ \} \end{aligned}$$

Finally, we test whether $fib_1(s^3(0),s^3(0))$ can be deduced from $B_3$ and $lgg_3$ or not, assuming that $fib(s^2(0),s^2(0))$ holds. We can prove easily that this holds. Therefore we obtain $lgg_3$ as one of our solutions.

## 5 Transformation of self recursion to mutual recursion

Since self recursive problems do not contain model constraining clauses, we cannot apply the algorithm *AIELMR* directly. To get rid of the problem, we add a clause to the background knowledge to trigger a mutual recursive call. After that we transform an obtained MaxSH to a self recursive one. We will show the algorithm in the following:

1. For a set of positive examples $E$, background knowledge $B$ and a target predicate $p(\mathbf{X})$, we introduce a new predicate $p'(\mathbf{X})$ and define a new background knowledge $B'$ by:
$$B' = B \cup \{p'(\mathbf{X}) \leftarrow p(\mathbf{X})\}.$$
where the clause $p'(\mathbf{X}) \leftarrow p(\mathbf{X})$ becomes the model constraining clause. The corresponding reduced-MCC shcema can be expressed as $p'(\mathbf{c_1}) \leftarrow p(\mathbf{c_2})$.

2. Compute MaxSHs from $E$ and $B'$ using the algorithm *AIELMR*.
3. Replace the predicate name $p'$ to $p$ in any clauses in the MaxSHs.

*Example 4.*

$$Let \begin{cases} B_4 = \{nat(s(0)).\} \\ E_4 = nat(s^4(0)). \end{cases}$$

Then, $B'_4 = \{nat(s(0)) \leftarrow, nat'(\mathbf{X}) \leftarrow nat(\mathbf{X})\}$. $bot(B'_4, E_4)$, reduced-MCC $RMCC_4$, $bot^+(B'_4, E_4)$ and $bot^-(B'_4, E_4)$ become as follows:

$$\begin{array}{ll} bot(B'_4, E_4) & = \{nat(s^4(0)), \neg nat(s(0)), \neg nat'(s(0))\} \\ RMCC_4 & = nat'(c_1) \leftarrow nat(c_2) \\ bot^+(B'_4, E_4) & = \{nat(s^4(0)), \neg nat(s(0)), \neg nat'(s(0)), \neg nat'(c_1)\} \\ bot^-(B'_4, E_4) & = \{\neg nat(s(0)), \neg nat'(s(0)), nat(c_2)\} \end{array}$$

By applying the substitution $c_1 = s^3(0)$ and $c_2 = s^2(0)$ to $bot^+(B'_4, E_4)$ and $bot^-(B'_4, E_4)$ respectively, we can compute their least generalization and obtain

$$lgg_4 = \{\ nat(s^2(X)), \neg nat'(s(X)), \neg nat(s(0)), \neg nat'(s(0))\ \}$$

Finally we test whether $nat'(s^3(0))$ holds or not under $B_4$ and $lgg_4$ assuming $nat(s^2(0))$. Since this is easily shown, we replace the predicate name $nat'$ to $nat$ in $lgg_4$ and obtain one of the MaxSHs:

$$lgg_4 = \{\ nat(s^2(X)),\ \neg nat(s(X)),\ \neg nat(s(0)),\ \neg nat(s(0))\ \}$$

## 6 Conclusion

This paper proposed an algorithm for computing sound and complete MaxSHs in linear mutual recursive cases under the condition that 1. background knowledge is a set of definite clauses and the examples are ground atoms, and the setting is the normal setting and 2. hypotheses are restricted to $h$-easy classes where the function $h$ is a constant function having the value $h$, and whose terms are bounded by the term size bound for the give problem. Although these restrictions seem severe, they are realistic and almost all problems in the real applications satisfy these constraints. Furthermore, most of the predictive ILP systems used in real applications adopt these restrictions. Since we can convert self recursion to mutual recursion as shown in the previous section, our algorithm covers self recursive cases as well. Furthermore, we showed that our algorithm can be extended to deal with non-linear recursive cases also.

For the efficient implementation, we need to optimize the least generalization to get rid of unnecessary computation taking mode information into account. We have developed an efficient algorithm to compute lgg of two clauses by computing literal lggs only for those literal pairs whose all input variables have been paired by previous lggs. We call such lgg as **data flow lgg** by the resemblance of its computation mechanism to data flow computing. We have been implementing the proposed algorithm using data flow lgg. Applications of this algorithm are another future work to be done.

## Acknowledgement

This paper was motivated by a discussion among one of the authors, Stephen Muggleton and Akihiro Yamamoto after the DS'98. We would like to express our thanks to both of them. We would like to thank also people in Artificial Intelligence group in the University of York who gave suggestions for improvement of the paper.

## References

1. Furukawa, K., Murakami, T., Ueno, K., Ozaki, T. and Shimazu, K. On a Sufficient Condition for the Existence of Most Specific Hypothesis in Progol, Proc. of Seventh International Workshop on Inductive Logic Programming, Lecture Notes in Artificial Intelligence 1297, Springer, 157-164, 1997.
2. Furukawa, K. On the Completion of the Most Specific Hypothesis in Inverse Entailment for Mutual Recursion and Abductive ILP Setting, Proceedings of 32nd SIG-FAI, JSAI, March, 1998 (in Japanese).
3. Furukawa, K. On the Completion of the Most Specific Hypothesis Computation in Inverse Entailment for Mutual Recursion, Proc. of Discovery Science 1998, Lecture Notes in Artificial Intelligence 1532, Springer, 1998.
4. Muggleton, S. Inverse Entailment and Progol, New Generation Computing, Vol.13, 245-286, 1995.
5. Muggleton, S. Completing Inverse Entailment, Proc. of Eighth International Conference on Inductive Logic Programming, Lecture Notes in Artificial Intelligence 1446, Springer, 244-249, 1998.
6. Plotkin, G.D. Automatic Method of Inductive Inference, Ph.D. thesis, Edinburgh University, 1971.
7. Yamamoto, A. Improving Theories for Inductive Logic Programming Systems with Ground Reduced Programs. Technical Report, Forschungsbericht AIDA-96-19 FG Intellektik FB Informatik TH Darmstadt, 1996.
8. Yamamoto, A. Which Hypotheses Can Be Found with Inverse Entailment? Proc. of Seventh International Workshop on Inductive Logic Programming, Lecture Notes in Artificial Intelligence 1297, Springer, 296-308, 1997.