Minimal Revision of Logical Specification Using Extended Logic Programming: Preliminary Report

Ken Satoh
Division of Electronics and Information Engineering
Hokkaido University
North 13 West 8 Kita-ku Sapporo 060-8628 Japan
E-mail: ksatoh@db-ei.eng.hokudai.ac.jp

Abstract

This paper presents a method of computing minimal revision of logical specification to handle inconsistencies. We have already proposed a formalization of minimal revision of logical specification (Satoh 1998) which is a modification of minimal revision in (Satoh 1988) and related the formalism with Formula Circumscription (McCarthy 1986). Moreover, we have proposed a computational method for a minimal revised logical specification without function symbols by abductive logic programming. However, the proposed computational method in (Satoh 1998) needs a minimality check of abducibles corresponding with revision in order to get a minimal revised specification. In this paper, we have proposed a method which directly computes a minimal revised specification. The technique is an adaptation of computational method of circumscription (Wakaki and Satoh 1997).

Introduction

In software engineering, there are several proposals of logical treatment of "inconsistency" of software specification (Borgida 1985; Balzer 1991; Finkelstein et al. 1994). A survey of these approaches is found in (Nuseibeh 1996). (Borgida 1985) handles the first systematic work on exception handling of integrity constraints in database specification and he proposes an isolation of such an exception from integrity constraints. (Balzer 1991) proposes a recovery of isolation when the exception is resolved for temporary violation of integrity constraints. (Finkelstein et al. 1994) uses non-collapsible "quasi-classical logic" even in the existence of inconsistency and formalizes consistency management between multiple specifications defined by several users.

These researches closely relate with belief revision in AI since managing of inconsistency is a major concern of belief revision. Especially, belief revision researchers study "minimal belief revision" (Katsuno and Mendelzon 1991) which corresponds with minimal revised specification in the presence of inconsistency in software engineering.

We have already proposed a formalization of minimal revision of logical specification (Satoh 1998) which is a modification of minimal revision in (Satoh 1988) and related the formalism with Formula Circumscription (McCarthy 1986). Moreover, we have proposed a computational method for a minimal revised logical specification without function symbols by abductive logic programming. However, the proposed computational method in (Satoh 1998) needs a minimality check of abducibles corresponding with revision in order to get a minimal revised specification. In other words, we have to compute all the possible set of abducibles at first and then select minimal sets among these.

In this paper, we have proposed a method which directly computes a minimal revised specification. The technique is an adaptation of computational method of ordinary circumscription used as a preprocessing to compute prioritized circumscription in (Wakaki and Satoh 1997). In (Wakaki and Satoh 1997), we have proposed a translation method of a set of clauses into an extended logic program and show that common literals included in all the answer sets are derived from circumscription. In this paper, we translate a logical specification into an extended logic program and show that each answer sets actually corresponds with a minimal revised logical specification. By using the correspondence, computation of answer sets for an extended logic program can be used to a minimal revised logical specification.

Formula-based Minimal Belief Revision

In this section, we review our framework of *formula-based minimal belief revision*(Satoh 1998) for maintaining consistency of software specification.

Consider the following example of logical representation of a database and constraints which is inspired by the example in (Borgida 1985, p590).

Example 1
Integrity Constraint meaning that if F is a father of E then the age of E must be under the age of F:

$$(\forall E, F/person)(\forall A1, A2/age)$$
$$(father(F,E) \wedge age(F,A1) \wedge age(E,A2)$$
$$\supset (A2 < A1))$$

A rule of calculating the age:
$(\forall E/person)(\forall Y, Z/year)(\forall A/age)$
$(birth\_year(E,Y) \wedge current\_year(Z) \wedge A = Z - Y$
$\supset age(E,A))$.
s is the father of c:
$father(s,c)$.
The birth year of s is 1985:
$birth\_year(s,85)$.
The birth year of c is 1984:
$birth\_year(c,84)$.
We also assume that there are some possibilities that the information of birth year is wrongly inserted in the database.

Suppose that we add $current\_year(99)$. *The addition of "*$c(99)$*."*[1] *leads to inconsistent database state. To resolve such inconsistency, we would consider the following possibilities.*

S_1: $b(s,35)$ *is incorrectly added and so, we delete this information from the database.*

S_2: $b(c,84)$ *is incorrectly added and so, we delete this information from the database.*

S_3: *We regard this situation as an exceptional situation and so, we change the integrity constraint.*

The first two cases are usual treatment of integrity constraint check and the last is proposed in (Borgida 1985). His idea is to change a part of applicability of the violated integrity constraint in order to restore consistency. In the above example, we change the integrity constraint into the following:

$(\forall E, F/person)(\forall A1, A2/age)$
$(\neg(F = s \wedge E = c \wedge A1 = 14 \wedge A2 = 15) \wedge$
$f(F,E) \wedge a(F,A1) \wedge a(E,A2) \supset (A2 < A1))$.

The above three changes relates with minimal belief revision studied in the field of AI. For a model of each revised specification, minimal deletion of facts or part of integrity constraints is performed. We call this kind of revision *formula-based minimal belief revision* since we minimize the change of satisfaction of considered formulas.

We explain why the above change is *minimal revision* of software specification. We focus on truth of the following three formulas:

$\alpha_1 : b(s,85)$
$\alpha_2 : b(c,84)$
$\alpha_3 : f(s,c) \wedge a(s,14) \wedge a(c,15) \supset (15 < 14)$

Note that the above three formulas are true in a logical model of the original specification. We calculate the difference of true formulas among the above three between a model of the original specification and a model of the new specifications.

[1] From now on, for brevity, we write "f" for "*father*" predicate, "a" for "*age*" "b" for *birth_year*, "c" for *current_year*, respectively

Model	Difference
a model of S_1	α_1
a model of S_2	α_2
a model of S_3	α_3

We can consider other specifications which are consistent with the new information $c(99)$, but we can show that differences between a model of these other specifications and a model the original specification are larger than either model of S_1, S_2 and S_3. Therefore, we say that these specifications realize formula-based minimal belief revision since we consider minimization of the difference of truth of formulas α_1, α_2 and α_3 between a model of the original specification and a model of a new specification.

In (Satoh 1998), we formalize formula-based minimal belief revision in the second-order formula and show that Formula Circumscription (McCarthy 1986) is a special case of formula-based minimal belief revision. Full definition of formula-based minimal belief revision formalizes not only minimal deletion of formula but also minimal addition, but since we only consider a minimal deletion of a part of specification in this paper, it is sufficient to consider a definition formalized in Formula Circumscription according to the relationship with (Satoh 1998) as follows.

Let $K(\Psi)$ and $\alpha(\Psi)$ be consistent first-order formulas which contain a tuple of predicates $\Psi = \langle p_1, ..., p_n \rangle$. We call $K(\Psi)$ *the current knowledge* which corresponds with absolute integrity constraints and absolute views (rules), and $\alpha(\Psi)$ *the added knowledge* which corresponds with additional absolute integrity constraints or additional absolute views (rules).

Let Φ be a tuple of formulas, $\langle \phi_1(\Psi, \mathbf{x}), ..., \phi_m(\Psi, \mathbf{x}) \rangle$ where $\mathbf{x}$ is a tuple of free variables in $\phi_1, ..., \phi_m$. We call $\phi_1, ..., \phi_m$ *focused formulas* since we consider a change of satisfaction of each focused formula between a logical model of the original specification and a model of the new specification.

Definition 1 *(Satoh 1998) Let* Ψ' *be a tuple of predicate variables* $\langle p'_1, ..., p'_n \rangle$ *each of whose arity is the same as the arity of the corresponding predicate in* $\langle p_1, ..., p_n \rangle$. *We define a* formula-based minimal revised belief, $New(\Psi, \Phi)$, *with respect to* $K(\Psi)$, $\alpha(\Psi)$ *and* Φ *as follows.*

$$New(\Psi, \Phi) =$$
$$K(\Psi) \wedge \alpha(\Psi) \wedge$$
$$\neg \exists \Psi' (K(\Psi') \wedge \alpha(\Psi') \wedge$$
$$\textstyle\bigwedge_{i=1}^{m} \forall \mathbf{x}((\phi_i(\Psi, \mathbf{x}) \supset \phi_i(\Psi', \mathbf{x})) \wedge$$
$$\neg \textstyle\bigwedge_{i=1}^{m} \forall \mathbf{x}(\phi_i(\Psi', \mathbf{x}) \supset \phi_i(\Psi, \mathbf{x}))))$$

where $K(\Psi')$ *is a formula obtained by substituting predicate variables of* Ψ' *for any occurrence of corresponding predicate constants in* $K(\Psi)$.

The above definition means that a model of $K(\Psi)$ with the negations of focused formulas minimized corresponds with a model of minimal revised belief.

Example 2 *Consider the specification in Example 1. For brevity, we consider* $f(s,c)$ *for* f *predicate,* $b(s,85)$ *and* $b(c,84)$ *for* b *predicate,* $c(99)$ *for* c *predicate, and* $a(s,14)$ *and* $a(c,15)$ *for* a *predicate.*

In the example, $K(\Psi)$ *corresponds with a conjunctions of the following three formulas:*

$b(s,85) \wedge c(99) \wedge 14 = 99 - 85 \supset a(s,14).$
$b(c,84) \wedge c(99) \wedge 15 = 99 - 84 \supset a(c,15).$
$f(s,c).$

and $\alpha(\Psi)$ *corresponds with*

$c(99).$

and Φ *is a tuple* $\langle \phi_1, \phi_2, \phi_3 \rangle$ *where*

$\phi_1 : b(s,85).$
$\phi_2 : b(c,84).$
$\phi_3 : f(s,c) \wedge a(s,14) \wedge a(c,15) \supset 15 < 14.$

Then, $New(\Psi, \Phi)$ *becomes:*

$(b(s,85) \wedge c(99) \wedge 14 = 99 - 85 \supset a(s,14)) \wedge$
$(b(c,84) \wedge c(99) \wedge 15 = 99 - 84 \supset a(c,15)) \wedge$
$f(s,c) \wedge c(99) \wedge$
$\neg \exists b' \exists c' \exists a' \exists f' ($
$(b'(s,85) \wedge c'(99) \wedge 14 = 99 - 85 \supset a'(s,14)) \wedge$
$(b'(c,84) \wedge c'(99) \wedge 15 = 99 - 84 \supset a'(c,15)) \wedge$
$f'(s,c) \wedge c'(99) \wedge$
$((b(s,85) \supset b'(s,85)) \wedge$
$(b(c,84) \supset b'(c,84)) \wedge$
$((f(s,c) \wedge a(s,14) \wedge a(c,15) \supset 15 < 14) \supset$
$(f'(s,c) \wedge a'(s,14) \wedge a'(c,15) \supset 15 < 14))) \wedge$
$\neg((b'(s,85) \supset b(s,85)) \wedge$
$(b'(s,84) \supset b(s,84)) \wedge$
$((f'(s,c) \wedge a'(s,14) \wedge a'(c,15) \supset 15 < 14) \supset$
$(f(s,c) \wedge a(s,14) \wedge a(c,15) \supset 15 < 14)))).$

Then, we can show that a model of $New(\Psi, \Phi)$ *is either model of* S_1, S_2 *and* S_3.

Computing Minimal Revised Logical Specification

To compute minimal revised logical specification, we introduce the following restriction; we consider a function-free language and assume unique name axioms and domain closure axioms for a logical specification and a logical specification is represented as a set of Horn clauses. We separate a logical specification into two parts; a persistent part which corresponds with $K(\Psi)$ and a temporary part which corresponds with Φ in the above definition.

Definition 2 *(Satoh 1998) Let* T_{pst} *be a set of Horn clauses which are of the form:*

$$B_1, B_2, ..., B_l \supset H.$$

where H, $B_1, ..., B_l$ *are atoms.*

Let T_{tmp} *be a set of labeled Horn clauses which are of the form:*

$$\phi : B_1, B_2, ..., B_l \supset H$$

where ϕ *is a name for the clause.*

A logical specification T *is a pair of* $\langle T_{pst}, T_{tmp} \rangle$ *and we call* T_{pst} *a* persistent part *of* T *and* T_{tmp} *a* temporary part *of* T.

It is possible that minimal revised belief results in a disjunction of conjunctions of Horn clauses. In this case, we let a user to choose one of the disjuncts. A disjunct is a maximally consistent subset of the original logical specification plus a new added formula defined as follows and we call the disjunct *minimal revised specification.*

Definition 3 *Let* S *be a set of function-free Horn clauses. A* maximally consistent subset *of* S *is* S' *such that* S' *is consistent and* $S' \subseteq S$ *and there is no proper superset* S'' *of* S' *such that* S'' *is consistent and* $S'' \subseteq S$.

Definition 4 *(Satoh 1998) Let* T *be a logical specification* $\langle T_{pst}, T_{tmp} \rangle$. *Let* $\Pi_{T_{tmp}}$ *be a set of ground clauses obtained by replacing all variables in each clause in* T_{tmp} *by every constant in* T. *Let* A_{new} *be a set of added clauses.*

A minimal revised specification *w.r.t.* T *and* A_{new} *is* $\langle T_{pst} \cup A_{new}, S \rangle$ *such that* $(T_{pst} \cup A_{new}) \cup S$ *is a maximal consistent subset of* $(T_{pst} \cup A_{new}) \cup \Pi_{T_{tmp}}$

Example 3 *Consider the database specification in the Example 1. The specification corresponds with* $T = \langle T_{pst}, T_{tmp} \rangle$ *where* T_{pst} *is a set of the following formulas:*

$b(E,Y) \wedge c(Z) \wedge A = Z - Y \supset a(E,A).$
$f(s,c).$

and T_{tmp} *is a set of the following formulas:*

$\phi_1 : b(s,85).$
$\phi_2 : b(c,84).$
$\phi_3 : f(F,E) \wedge a(F,A1) \wedge a(E,A2) \supset A2 < A1.$

Let us add $c(99)$. *This means that* A_{new} *is a set* $\{c(99)\}$. *The new specification leads to inconsistency, so we need to revise the specification.*

We firstly show $\Pi_{T_{tmp}}$.

$b(s,85).$
$b(c,84).$
$f(s,c) \wedge a(s,14) \wedge a(c,15) \supset (15 < 14).$
$f(s,s) \wedge a(s,14) \wedge a(s,15) \supset (15 < 14).$
$f(c,s) \wedge a(c,14) \wedge a(s,15) \supset (15 < 14).$
$f(c,c) \wedge a(c,14) \wedge a(c,15) \supset (15 < 14).$
...

Let

S_1 *be* $\Pi_{T_{tmp}} - \{b(s,85)\}$,
S_2 *be* $\Pi_{T_{tmp}} - \{b(c,84)\}$,
S_3 *be* $\Pi_{T_{tmp}} - \{f(s,c) \wedge a(s,14) \wedge a(c,15) \supset (15 < 14)\}$.

Then, since each $(T_{pst} \cup A_{new}) \cup S_i (i = 1,2,3)$ *is a maximal consistent subset of* $(T_{pst} \cup A_{new}) \cup \Pi_{T_{tmp}}$, *each* $\langle T_{pst} \cup A_{new}, S_i \rangle$ *is a minimal revised specification w.r.t.* T *and* A_{new}.

According to the relationship with Formula Circumscription and our framework (Satoh 1998), for every model of minimal revised specification for a function-free Horn clauses (with UNA and DCA), there exists a model of circumscription where persistent rules are axioms and negation of temporary rules are minimized. In other words, minimal revised specification corresponds

with a set of rules where temporary part is retained as much as possible.

To compute a minimal revised specification, we use a translation from a specification to an extended logic program and compute an answer set for the translated program which denotes a deletion of some part of the specification. In (Satoh 1998), we use abductive logic programming to compute a revised specification, but we need to minimality check to obtain a minimal revised specification whereas in this paper, we propose a method to compute a minimal revised specification directly.

Firstly, we define an extended logic program.

Definition 5 *Let* H, $P_1, ..., P_j$, $N_1, ..., N_h$ *be literals. A* rule *is an expression of the form:*

$$H \leftarrow P_1, ..., P_j, \sim N_1, ..., \sim N_h$$

where $\sim$ *represents "Negation as Failure". An* extended logic program *is a set of rules.*

We use a symbol $\overline{A}$ *denoted as a complement literal for* A.

The semantics of extended logic program is based on answer set semantics (Gelfond and Lifschitz 1991), but we use the following correspondence between an answer set and a stable model (Gelfond and Lifschitz 1988) of a normal logic program in order to utilize our proof procedure based on stable model (Satoh and Iwayama 1992).

Definition 6 *Let* P *be an extended logic program and* Π_P *is a set of ground rules obtained by replacing all the variables in every rule of* P *by every constant in* P. *Let* M *be a set of ground literals and* Π_P^M *be the following program.*

$$\Pi_P^M = \{H \leftarrow B_1, ..., B_l | \text{ ``}H \leftarrow B_1, ..., B_l, \sim A_1, ..., \sim A_h.\text{''} \in \Pi_P \text{ and } A_i \notin M \text{ for each } i = 1, ..., h.\}$$

Let $min(\Pi_P^M)$ *be the least model of* Π_P^M. *An* answer set *for an extended logic program* P *is* M *iff* $M = min(\Pi_P^M)$ *and* $\bot \notin M$[2] *and for no literal* $L \in M$, $\overline{L} \in M$.

Now, we define a translation for a consistency management for a logical specification into an extended logic program as follows.

Definition 7 *Let* T *be a logical specification* $\langle T_{pst}, T_{tmp} \rangle$. *A* translation for a consistency management of T *(denoted as* $CM(T)$*) is a set of the following translation from* T *to an extended logic program* P:

- *We translate every ground clause* $(B_1, ..., B_l \supset H)\theta$ *in* $\Pi_{T_{pst}}$ *into the following rules in* P *which corresponds with every contrapositives of the clause:*
  $(H \leftarrow B_1, ..., B_l)\theta$.
  $(\overline{B_1} \leftarrow B_2, ..., B_l, \overline{H})\theta$.
  $(\overline{B_2} \leftarrow B_1, B_3, ..., B_l, \overline{H})\theta$.
  ...
  $(\overline{B_l} \leftarrow B_1, ..., B_{l-1}, \overline{H})\theta$.

[2] $\bot$ means falsity and we use $\bot$ as the head of an integrity constraint.

- *We instantiate every clause* $\phi : (B_1, ..., B_l \supset H) \in T_{tmp}$ *into the ground clause* $(B_1, ..., B_l \supset H)\theta$ *in* $\Pi_{T_{tmp}}$ *and then add the following rules in* P:
  $(H \leftarrow B_1, ..., B_l, \sim del_\phi^*(\mathbf{x}))\theta$.
  $(\overline{B_1} \leftarrow B_2, ..., B_l, \overline{H}, \sim del_\phi^*(\mathbf{x}))\theta$.
  $(\overline{B_2} \leftarrow B_1, B_3, ..., B_l, \overline{H}, \sim del_\phi^*(\mathbf{x}))\theta$.
  ...
  $(\overline{B_l} \leftarrow B_1, ..., B_{l-1}, \overline{H}, \sim del_\phi^*(\mathbf{x}))\theta$.
  $(del_\phi^*(\mathbf{x}) \leftarrow B_1, ..., B_l, \overline{H})\theta$.
  where $\mathbf{x}$ *is a tuple of free variables in the clause. We call a literal with* del_ϕ^* *for a predicate name* a deletion literal.
- *For every ground literal* $p(\mathbf{x})\theta$, *we add the following integrity constraints to* P:

$$(\bot \leftarrow p(\mathbf{x}), \overline{p}(\mathbf{x}))\theta.$$

The following theorem shows that a minimal revised specification can be computed from an answer set.

Theorem 1 *Let* T *be a function-free logic specification* $\langle T_{pst}, T_{tmp} \rangle$ *and* A_{new} *be a set of added clauses.* $\langle T_{pst} \cup A_{new}, (T_{tmp} - T_{del}) \cup T_{new} \rangle$ *is a minimal revised specification if and only if there is an answer set of* $CM(\langle T_{pst} \cup A_{new}, T_{tmp} \rangle)$ *with a deletion literals* Δ *s.t.*
$T_{del} = \{\phi : B_1, ..., B_l \supset H | (del_\phi(\mathbf{x})\theta) \in \Delta\}$ *and*
$T_{new} =$
$\{\phi : B_1, ..., B_l, \neg(EQ(\theta_1)), ..., \neg(EQ(\theta_m)) \supset H |$
$(del_\phi(\mathbf{x})\theta_i) \in \Delta$ *and*
$EQ(\theta_i) = ((x_1 = (x_1\theta_i)) \wedge ... \wedge (x_k = (x_k\theta_i)))$
where $\mathbf{x} = \langle x_1, ..., x_k \rangle\}$.

Example 4 *Consider the database specification* $T = \langle T_{pst}, T_{tmp} \rangle$ *in the Example 3. To compute a minimal revised specification, we translate the specification into the following extended logic program*[3] .

$a(s, 14) \leftarrow b(s, 85), c(99)$.
$\overline{b}(s, 85) \leftarrow \overline{a}(s, 14), c(99)$.
$\overline{c}(99) \leftarrow \overline{a}(s, 14), b(s, 85)$.
$a(c, 15) \leftarrow b(c, 84), c(99)$.
$\overline{b}(c, 84) \leftarrow \overline{a}(c, 15), c(99)$.
$\overline{c}(99) \leftarrow \overline{a}(c, 15), b(c, 84)$.
$f(s, c)$.
$c(99)$.
$b(s, 85) \leftarrow \sim del_{\phi_1}$.
$del_{\phi_1} \leftarrow \overline{b}(s, 85)$.
$b(c, 84) \leftarrow \sim del_{\phi_2}$.
$del_{\phi_2} \leftarrow \overline{b}(c, 84)$.
$\bot \leftarrow f(s, c), a(s, 14), a(c, 15), \sim del_{\phi_3}(s, c, 14, 15)$.
$del_{\phi_3}(s, c, 14, 15) \leftarrow f(s, c), a(s, 14), a(c, 15)$.
$\overline{f}(s, c) \leftarrow a(s, 14), a(c, 15), \sim del_{\phi_3}(s, c, 14, 15)$.
$\overline{a}(s, 14) \leftarrow f(s, c), a(c, 15), \sim del_{\phi_3}(s, c, 14, 15)$.
$\overline{a}(c, 15) \leftarrow f(s, c), a(s, 14), \sim del_{\phi_3}(s, c, 14, 15)$.
$\bot \leftarrow f(s, c), \overline{f}(s, c)$.
$\bot \leftarrow f(s, c), \overline{f}(s, c)$.

[3] After instantiating all the variables, We evaluate numerical predicate such as $14 = 99 - 85$ and $14 \leq 15$ in advance for brevity.

$\bot \leftarrow c(99), \overline{c}(99)$.
$\bot \leftarrow a(c,15), \overline{a}(c,15)$.
$\bot \leftarrow a(s,14), \overline{a}(s,14)$.
$\bot \leftarrow b(s,85), \overline{b}(s,85)$.
$\bot \leftarrow b(c,84), \overline{b}(c,84)$.

Then, we have the following three answer sets, A_1, A_2, A_3.

$$A_1 = \{del_{\phi_1}, \overline{b}(s,85), \overline{a}(s,14), a(c,15), b(c,84), f(s,c), c(99)\}$$

$$A_2 = \{del_{\phi_2}, \overline{b}(c,84), \overline{a}(c,15), a(s,14), b(s,85), f(s,c), c(99)\}$$

$$A_3 = \{del_{\phi_3}(s,c,14,15), a(c,15), b(c,84), a(s,14), b(s,85), f(s,c), c(99)\}$$

Therefore, from Theorem 1, we have the following three minimal revised specification which corresponds with the results in Example 3.

Specification 1:
$\phi_2 : b(c,84)$.
$\phi_3 : f(F,E) \wedge a(F,A1) \wedge a(E,A2) \wedge A1 \leq A2 \supset \bot$.

Specification 2:
$\phi_1 : b(s,85)$.
$\phi_3 : f(F,E) \wedge a(F,A1) \wedge a(E,A2) \wedge A1 \leq A2 \supset \bot$.

Specification 3:
$\phi_1 : b(s,85)$.
$\phi_2 : b(c,84)$.
$\phi_3' : \neg(F = s, E = c, A1 = 14, A2 = 15) \wedge f(F,E) \wedge a(F,A1) \wedge a(E,A2) \wedge A1 \leq A2 \supset \bot$.

To calculate an answer set, we can use our previous proof procedure (Satoh and Iwayama 1992) based on the stable model semantics. We use the correspondence between the answer set semantics for an extended logic program and the stable model semantics for a normal logic program shown in (Gelfond and Lifschitz 1991).

This is originally used to evaluate a query, but in the procedure, we have a rule checking part in which we examine consistency of an addition of a rule and if there is a stable model for the program plus the rule, then the procedure outputs deletion of rules which are necessary to maintain consistency.

For a function-free Horn logical specification, we can calculate all the stable models and therefore, we can calculate all minimal revised specification. In fact, all the results in the above example are computed by our proof procedure.

Conclusion

In this paper, we propose a calculation method of a minimal revised logical specification in the case that the specification consists of Horn clauses. It is done by translating the logical specification into an extended logic program and by showing relationship between deletion information included in an answer set and minimal revised specification.

As a future work we would like to consider not only two levels of retractability of specification but also various levels of retractability. To represent such level, we need prioritization of formulas and manipulation of retracting formulas according to prioritization. *Prioritized Circumscription* (McCarthy 1986) may be used and we extend our mechanism in order to handle prioritized circumscription by adapting a method proposed in (Wakaki and Satoh 1997).

Acknowledgments

This research is partly supported by Grant-in-Aid for Scientific Research on Priority Areas, "Principles for Constructing Evolutionary Software", The Ministry of Education, Japan. We thank a referee for comments to improve the paper.

References

Balzer, R. 1991. Tolerating Inconsistency. *Proc. of ICSE-13*, pp. 158 - 165.

Borgida, A. 1985. Language Features for Flexible Handling of Exceptions in Information Systems. *ACM Transactions on Database Systems*, **10**, pp. 565 - 603.

Finkelstein, A. C. W., Gabbay, D., Hunter, A., Kramer, J. and Nuseibeh, B. 1994. Inconsistency Handling in Multiperspective Specifications *IEEE Transactions on Software Engineering*, 20, pp. 569 - 578.

Gelfond, M., Lifschitz, V. 1988. The Stable Model Semantics for Logic Programming. *Proc. of LP'88*, pp. 1070 - 1080.

Gelfond, M., Lifschitz, V. 1991. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9, pp. 365 - 385.

Katsuno, H., Mendelzon, A. O. 1991. Propositional Knowledge Base Revision and Minimal Change. *Artificial Intelligence*, **52**, pp. 263 - 294 (1991).

McCarthy, J. 1986. Applications of Circumscription to Formalizing Common-Sense Knowledge. *Artificial Intelligence*, **28**, pp. 89 - 116.

Nuseibeh, B. 1996. To Be and Not to Be: On Managing Inconsistency in Software Development. *Proc. of 8th IEEE International Workshop on Software Specification and Design (IWSSD-8)* pp. 164 - 169.

Satoh, K. 1988. Nonmonotonic Reasoning by Minimal Belief Revision. *Proc. of FGCS'88*, pp. 455 - 462.

Satoh, K., Iwayama, N. 1992. A Query Evaluation Method for Abductive Logic Programming. *Proc. of JICSLP'92*, pp. 671 - 685

Satoh, K. 1998. Computing Minimal Revised Logic Program by Abduction. *Proc. of the International Workshop on the Principles of Software Evolution*, pp. 177 - 182.

Wakaki, T., Satoh K. 1997. Compiling Prioritized Circumscription into Extended Logic Programs. *Proc. of IJCAI-97*, pp. 182 - 187.