Abductive Logic Programming by Nonground Rewrite Systems

Fangzhen Lin
Hong Kong University of Science and Technology, China

Jia-Huai You
University of Alberta, Canada
Abduction

The general problem of abduction is to show that an observation is explained by a reasoning process supported by hypotheses while satisfying stated constraints.

Logic programs with negation consisting of normal rules of the form:

\[ A \leftarrow B_1, \ldots, B_k, \text{not } C_1, \ldots, \text{not } C_n \]

have been considered a suitable yet powerful formalism for abductive logic programming [Kakas et al. ’95].
Previous Work

• Under the (partial) stable model semantics: Programs are *ground* [Eshghi & Kowalski ’89; Kakas et al. ’00; Lin & You ’02]

• Under the completion semantics: Programs can be *nonground*; but with severe restrictions and formulations are complex
  – Using iff definitions as rewrite rules [Console et al. ’91; Fung & Kowalski ’97; Endreiss et al. ’04]
  – SLDNFA procedure [Denecker & De Schreye ’98]

• Reasons for the restrictions and complication: Queries may have **universal variables**; *Unsafe* computations can be avoided by *non-floundering queries or allowedness conditions*
Results of this paper

• Show a simple view of abduction in this context for the completion semantics.

• Under our formulation, the problem becomes one of solving quantified equations and disequations.

• Approach is sound for standard 2-valued logic and complete for 3-valued logic.

• No safety condition (what’s-so-ever) is necessary, since there is no notion of safe or unsafe computations.
Abductive programs

An abductive program is a triple $\langle T, IC, Ab \rangle$.

1. $T$ is a finite set of iff definitions of the form

$$p(X_1, \ldots, X_n) \leftrightarrow D_1 \lor \cdots \lor D_m$$

We assume that such an iff definition is a completed definition of a predicate by normal program rules that define predicate $p/n$, whose logic formula is:

$$\forall \underline{X} \ p(\underline{X}) \leftrightarrow \bigvee_{i=1}^{m} \exists \underline{Y}_i [ (\underline{X} = \underline{s}_i) \land \Phi_i ) ] \quad (1)$$

- each disjunct corresponds to a $D_i$ above
- $\underline{X} = \underline{s}_i$ is the conjunction of equations representing unifiability
- $\underline{Y}_i$ denotes the variables other than $\underline{X}$ appearing in the disjunct
- $\Phi_i$ is the rule body with not replaced by $\neg$. 
2. *IC* is a consistent finite set of constraints of the form

\[ \bot \leftarrow A_1, \ldots, A_k, \neg B_1, \ldots, \neg B_m \]

where \( A_i \) and \( B_i \) are atoms and all variables are universally quantified.

A constraint can be written as a disjunction

\[ \neg A_1 \lor \ldots \lor \neg A_k \lor B_1 \lor \ldots \lor B_m \]

3. \( Ab \) is a finite set of predicate symbols, called *abducibles*, which are different from \( = \) and any defined predicate symbol.

A query (or a goal) is a formula of form

\[ \overline{\Theta} \Phi \]

where \( \overline{\Theta} \) is a tuple of quantifiers and \( \Phi \) a quantifier-free formula with negation appearing only in front of an atom.
Answers to a query

Given an abducible program \( \langle T, IC, Ab \rangle \) and a query \( G \), the initial query is a formula

\[
\forall \overline{X}. G \land IC
\]

where \( IC \) is the conjunction of all constraints, and \( \overline{X} \) the tuple of variables in \( IC \).

An answer to a query \( G \) is a pair \( (\Delta, \sigma) \), where \( \Delta \) is a finite set of ground abducible atoms, and \( \sigma \) is a substitution of ground terms for variables in \( V(G) \), such that

\[
T \cup \text{comp}(\Delta) \cup \text{CET} \models G\sigma \land \forall \overline{X}IC. \tag{2}
\]

where CET is the Clark’s equality theory (essentially, all syntactically distinct ground terms are not equal, and functions are one-to-one).
Simple view

An initial query is rewritten to a formula of the form

$$
\overline{\Theta}(E \lor \Phi)
$$

where $\overline{\Theta}$ is a tuple of quantifiers, $E \lor \Phi$ is quantifier-free, and $E$ contains only $=$ and abducibles.

Then, answers are extracted from $\overline{\Theta}E$. The process of answer extraction is essentially one of solving quantified equations and disequations.

Rewrite rules: iff definitions plus some logic equivalences and rules for unification.
In general,

$$\Theta(E \lor \Phi) \neq \Theta E \lor \Theta \Phi$$

But in some cases, e.g. when $E$ and $\Phi$ do not share universal variables, we have

$$\Theta(E \lor \Phi) \equiv \Theta E \lor \Theta \Phi$$
Simplification rules

Let $\overline{\Theta} \Pi$ be a goal formula. $\Pi$ can be rewritten according to the following rules (and their symmetric cases).

SR1. $F \lor \Phi \rightarrow \Phi$  
SR2. $F \land \Phi \rightarrow F$

SR3. $T \land \Phi \rightarrow \Phi$  
SR4. $T \lor \Phi \rightarrow T$

SR5. $(\Phi_1 \lor \Phi_2) \land \Phi_3 \rightarrow (\Phi_1 \land \Phi_3) \lor (\Phi_2 \land \Phi_3)$
Formulas for Unfolding

Unfolding a negative literal:

\[ \neg p(t) \leftrightarrow \bigwedge_{i=1}^{m} \forall Y_i [t \neq s_i \lor \neg \Phi_i] \]  \hspace{1cm} (i)

\[ \leftrightarrow \bigwedge_{i=1}^{m} \forall Y_i [t \neq s_i \lor [t = s_i \land \neg \Phi_i]] \]  \hspace{1cm} (ii)

\[ \leftrightarrow \bigwedge_{i=1}^{m} \forall \overline{Y_i}^f [t \neq s_i] \lor \exists Z_i \forall R_i [(t = s_i \land \neg \Phi_i)\{\overline{Y_i}^f/Z_i\}] \]  \hspace{1cm} (iii)

where \( \overline{Y_i}^f = V(s_i) \), which are renamed to \( Z_i \) in the second disjunct (by substitution \( \{\overline{Y_i}^f/Z_i\} \)) and become existentially quantified, and \( R_i = Y_i \setminus \overline{Y_i}^f \) are the remaining variables.
Example

Given

\[ p(V) \leftrightarrow \exists X, Y, Z(V = s(X, Y) \land \neg q(Y, f(Z))), \]

for \( \neg p(f(X)) \), the formula (i) above is

\[ \forall X_1, X_2, X_3(f(X) \neq s(X_1, X_2) \lor q(X_2, f(X_3))), \]

and the formula (iii) is

\[ \forall X_1, X_2(f(X) \neq s(X_1, X_2)) \lor \\
    \exists Y_1, Y_2 \forall X_3(f(X) = s(Y_1, Y_2) \land q(Y_2, f(X_3))). \]
Correctness of unfolding

Lemma

Given a program $P$, a predicate $p$, and a completed definition in the form described above, let $p(\overline{t})$ be an atom such that $V(\overline{t}) \cap \overline{Y_i} = \emptyset$ for each $i$. Then, $\text{comp}(P) \cup \text{CET}$ entails

(a) $\forall \ p(\overline{t}) \leftrightarrow \bigvee_{i=1}^{m} \exists Y_i[(\overline{t} = \overline{s_i}) \land \Phi_i]$ 

(b) $\forall \neg p(\overline{t}) \leftrightarrow$ the formula in (iii) above.
Unfolding

Quantifiers can always be moved to the left of a goal.

Let $\Pi = \Theta \Phi$ be a goal formula.

If $p(\overline{t})$ occurs in $\Phi$ positively (i.e. not under $\neg$ operator), then rewrite $\Pi$ into the following goal formula

$$\Theta \exists Y_1 \cdots Y_m \Phi',$$

where $\Phi'$ is the result of replacing this occurrence of $p(\overline{t})$ by

$$\bigvee_{i=1}^{m} (\overline{t} = \overline{s}_i) \land \Phi_i.$$
Extract answers

Let $\langle T, IC, Ab \rangle$ be an abductive program and $G$ a goal. Assume the initial query $\forall X. G \land IC$ has been rewritten to $\Theta(E \lor \Phi)$, where $E$ only mentions $=$ and abducibles.

Let $\Delta$ be a finite set of ground abducible atoms.

1. Abduction

We abduce $\Theta E$ to $\Theta E_\Delta$ by replacing any occurrence of $ab(\overline{t})$ in $E$ by $\forall_i \overline{t} = \overline{s_i}$, and $\neg ab(\overline{t})$ by $\land_i \overline{t} \neq \overline{s_i}$, where $ab(s_i) \in \Delta$.

2. Extraction

Let $\sigma$ be a substitution for variables in $V(G)$.

$(\Delta, \sigma)$ is extracted as an answer to $G$ if $CET \models \Theta E_\Delta \sigma$. 
Soundness

Theorem (Soundness)
Let \( \langle T, IC, Ab \rangle \) be an abductive program and \( G \) a goal.

(1) Suppose rewriting from \( \forall X. G \land IC \) generates \( \overline{\Theta}(E \lor \Phi) \) such that \( (\Delta, \sigma) \) is extracted as an answer to \( G \), based on \( E \). Then,
\[
T \cup \text{comp}(\Delta) \cup \text{CET} \models G\sigma \land \forall X IC.
\]

(2) If rewriting from \( \forall X. G \land IC \) generates \( F \), then \( T \cup \text{CET} \cup \forall X IC \models \neg \exists G \).
Completeness

Theorem (Completeness) Let \( \langle T, IC, Ab \rangle \) be an abductive program and \( G \) a goal. Suppose \( (\Delta, \sigma) \) is an answer to \( G \) under 3-valued logic: \( T \cup \text{comp}(\Delta) \cup \text{CET} \models_3 G\sigma \land \forall XIC \). Then, there is a derivation from \( \forall X.G \land IC \) to a goal formula \( \Theta(E \lor \Phi) \), where \( E \) is not further reducible by any defined predicates, such that an answer \( (\Delta', \sigma') \) can be extracted, based on \( E \), where \( \Delta' \) is a subset of \( \Delta \) and \( \sigma' \) is more general than \( \sigma \).

The difficulty for the 2-valued completion semantics is known to be caused by loops, e.g., with \( T = \{p \leftrightarrow \neg p\} \), any goal would follow in 2-valued logic.
Negation handled soundly: a side product

Example

Let $\langle T, \emptyset, \emptyset \rangle$ be an abductive program, where $T$ is

\[
\{ p(X) \leftrightarrow \neg r(X, Y); \quad r(X, Y) \leftrightarrow X = a \land Y = b \}
\]

$T$ is the completion of the normal program:

\[
p(X) \leftarrow \text{not } r(X, Y). \quad r(a, b).
\]

Clearly, $T \cup \text{CET} \models p(t)$, for any $t$ in the language (including at least $a$ and $b$). This can be shown using our rewrite system:

\[
p(V) \Rightarrow \exists Y \neg r(V, Y) \Rightarrow \exists Y (V \neq a \lor Y \neq b)
\]

On the other hand, we expect $\neg p(V)$ to be proved false.

\[
\neg p(V) \Rightarrow \forall Y r(V, Y) \Rightarrow \forall Y (V = a \land Y = b) \Rightarrow F
\]
Consider the faulty-lamp example. Suppose the abductive program is \( \langle T, \emptyset, \{\text{power\_failure}, \text{empty}\} \rangle \), where \( T \) is

\[
\begin{align*}
\text{faulty\_lamp} & \leftrightarrow \text{power\_failure}(X) \land \neg \text{backup}(X) \\
\text{backup}(X) & \leftrightarrow \text{battery}(X, Y) \land \neg \text{empty}(Y) \\
\text{battery}(X, Y) & \leftrightarrow X = b \land Y = c
\end{align*}
\]

The abbreviations used below should be clear.

\[
\begin{align*}
fl & \Rightarrow [pf(X_{\exists 1}), \neg \text{backup}(X_{\exists 1})] \\
& \Rightarrow [pf(X_{\exists 1}), [\neg \text{batt}(X_{\exists 1}, X_{\forall 2}) \lor \text{emp}(X_{\forall 2})]] \\
& \Rightarrow [pf(X_{\exists 1}), [X_{\exists 1} \neq b \lor X_{\forall 2} \neq c \lor \text{emp}(X_{\forall 2})]] \\
& \Rightarrow [pf(X_{\exists 1}), X_{\exists 1} \neq b] \lor [pf(X_{\exists 1}), [X_{\forall 2} \neq c \lor \text{emp}(X_{\forall 2})]]
\end{align*}
\]

The first disjunct gives answer \( (\{pf(t)\}, \emptyset) \), for any \( t \neq b \); for the second, we have \( \Delta = \{pf(t), \text{emp}(c)\} \), for any \( t \) in our language.
Future work

- The approach requires an efficient reasoner to reason about quantified equations and disequations.
- May further reduce the dependencies of disjuncts in a goal formula $\overline{\Theta(E \lor \Phi)}$.
- Rewrite strategies need to be addressed.