Abduction and Logic Programming

Lorenz Leutgeb

2017-06-19

L. Console, D. T. Dupré, and P. Torasso. On the relationship between abduction and deduction.
J. Log. Comput., 1(5):661–690, 1991.
doi: 10.1093/logcom/1.5.661.
URL http://dx.doi.org/10.1093/logcom/1.5.661

T. Eiter, G. Gottlob, and N. Leone. Abduction from logic programs: Semantics and complexity.
Theor. Comput. Sci., 189(1-2):129–177, 1997.
doi: 10.1016/S0304-3975(96)00179-X.
URL http://dx.doi.org/10.1016/S0304-3975(96)00179-X

Agenda

1. Deduction, Induction, Abduction [4]
2. Logic Programming & Logic Programs
3. Deduction via Abduction [2]
4. A Framework for Abduction Problems [3]

Deduction

	Rule.	All the beans from this bag are white.
	Case.	These beans are from this bag.
$\therefore$	*Result.*	These beans are white.

Example: Mathematics heavily depend on deduction to derive more explicit knowledge from basic axioms.

Induction

	Case.	These beans are from this bag.
	Result.	These beans are white.
$\therefore$	*Rule.*	All the beans from this bag are white.

Example: Physicists employ experiments and their results to derive the laws of physics, which can be regarded as induction.

Abduction

	Rule.	All the beans from this bag are white.
	Result.	These beans are white.
$\therefore$	*Case.*	These beans are from this bag.

Example: A physician uses medical knowledge to explain the symptoms of her patient, a diagnosis obtained by abduction.

Logic Programming

- A branch of declarative programming that remixes first order logic.
- Basic idea: Not *how* but *what*.
- Sudoku solver in less than 20 LOCs.
- Stand on the shoulders of highly optimized solvers.
- User interaction essentially does not work (yet).
- Declarative Problem Solving (184.701, UE)

Logic Programs

Definition

A *rule* is an ordered pair of the form

$$a_1 \vee \ldots \vee a_m \leftarrow b_1 \wedge \ldots \wedge b_k \wedge \text{not } b_{k+1} \wedge \ldots \wedge \text{not } b_n$$

where $a_1, \ldots, a_m, b_1, \ldots, b_n$ are literals, not is *negation as failure* (or *default negation*), $a_1 \vee \ldots \vee a_m$ is the *head* of r and $b_1, \ldots, b_k, \text{not } b_{k+1}, \ldots, \text{not } b_n$ is the *body* of r.

Abduction via Deduction (1/4)

Given a theory . . .

$$\begin{aligned} \mathit{grass\_is\_wet} &\leftarrow \mathit{rained\_last\_night} \\ \mathit{grass\_is\_wet} &\leftarrow \mathit{sprinkler\_was\_on} \\ \mathit{grass\_is\_cold\_and\_shiny} &\leftarrow \mathit{grass\_is\_wet} \\ \mathit{shoes\_are\_wet} &\leftarrow \mathit{grass\_is\_wet} \end{aligned}$$

Abduction via Deduction (1/4)

Given a theory ...

$$
\begin{aligned}
grass\_is\_wet &\leftarrow rained\_last\_night \\
grass\_is\_wet &\leftarrow sprinkler\_was\_on \\
grass\_is\_cold\_and\_shiny &\leftarrow grass\_is\_wet \\
shoes\_are\_wet &\leftarrow grass\_is\_wet
\end{aligned}
$$

We observe that ...

$$grass\_is\_cold\_and\_shiny$$

Abduction via Deduction (1/4)

Given a theory ...

$$
\begin{aligned}
grass\_is\_wet &\leftarrow rained\_last\_night\\
grass\_is\_wet &\leftarrow sprinkler\_was\_on\\
grass\_is\_cold\_and\_shiny &\leftarrow grass\_is\_wet\\
shoes\_are\_wet &\leftarrow grass\_is\_wet
\end{aligned}
$$

We observe that ...

$$grass\_is\_cold\_and\_shiny$$

And ask ourselves: Why?

Abduction via Deduction (1/4)

Given a theory ...

$$\begin{aligned} grass\_is\_wet &\leftarrow rained\_last\_night \\ grass\_is\_wet &\leftarrow sprinkler\_was\_on \\ grass\_is\_cold\_and\_shiny &\leftarrow grass\_is\_wet \\ shoes\_are\_wet &\leftarrow grass\_is\_wet \end{aligned}$$

We observe that ...

$$grass\_is\_cold\_and\_shiny$$

And ask ourselves: Why?

Abduction via Deduction (2/4)

Our theory after “completion” of non-abducibles [1]:

$$\begin{array}{lcl} grass\_is\_wet & \leftrightarrow & rained\_last\_night \lor sprinkler\_was\_on \\ grass\_is\_cold\_and\_shiny & \leftrightarrow & grass\_is\_wet \\ shoes\_are\_wet & \leftrightarrow & grass\_is\_wet \end{array}$$

Definition ([2])

The emcompletion LP_C is a set of equivalences $\{p_i \leftrightarrow D_i | i = 1, \ldots, n\}$, where $p_1, \ldots p_n$ are all the non-abdicuble atoms in LP and $D_i \equiv Q_{i1} \lor \ldots \lor Q_{im}$ in case $\{Q_{ij} \to p_i | j = 1, \ldots, m\}$ is the set of clauses in LP having p_i as their head.

Abduction via Deduction (3/4)

Hypotheses:

$$rained\_last\_night$$
$$sprinkler\_was\_on$$

Abduction via Deduction (4/4)

Console et al. [2] show that this “object level” characterization of abduction corresponds to the “meta level ” characterization.
They generalize the above approach for:

1. *taxonomic* or *abstraction relationships* between abducible atoms, i.e. of the form

$$\alpha \rightarrow \beta \qquad \alpha(X) \rightarrow \beta(X)$$

2. *constraints* between abducible atoms in the form of denials/nogoods, i.e. of the form

$$\neg(\alpha_1 \wedge \ldots \wedge \alpha_n) \qquad \neg(\alpha_1(t_1) \wedge \ldots \wedge \alpha_n(t_n))$$

A Framework for Abduction Problems (1/3)

Definition ([3, Definition 1, p. 140])

Let V be a set of propositional atoms. A logic programming abduction problem (LPAP) $\mathscr{P}$ over V consists of a tuple $\langle H, M, LP, \models \rangle$ where $H \subseteq V$ is a finite set of hypotheses, $M \subseteq V \cup \{\neg v | v \in V\}$ is a finite set of manifestations, LP is a propositional logic program on V and $\models$ is an inference operator.

A Framework for Abduction Problems (2/3)

Interesting inference operators: $\models_{wf}$, $\models_{st}^{b}$, $\models_{st}^{c}$

Definition ([3, p. 137])

Brave reasoning (or *credulous reasoning*) infers that a literal Q is true in LP (denoted $LP \models_{st}^{b} Q$) iff Q is true with respect to M for **some** $M \in \mathrm{STM}(LP)$.

Definition ([3, p. 137])

Cautious reasoning (or *skeptical reasoning*) infers that a literal Q is true in LP (denoted $LP \models_{st}^{b} Q$) iff (1) Q is true with respect to M for **all** $M \in \mathrm{STM}(LP)$ and (2) $\mathrm{STM}(LP) \neq \emptyset$.

A Framework for Abduction Problems (3/3)

Interesting problems:

1. Solution verification ($S \in \mathrm{Sol}(\mathscr{P})$),
2. consistency checking ($\mathrm{Sol}(\mathscr{P}) \neq \emptyset$),
3. and $\preceq$-relevance, -necessity of some $h \in H$ for LPAPs and disjunctive LPAPs where preference $\preceq$ is either minimality with respect to inclusion ($\subseteq$), cardinality ($\leq$) or no preference ($=$)

…along all three inference operators $\models_{wf}$, $\models_{st}^{b}$, $\models_{st}^{c}$.

Recap

1. Deduction, Induction, Abduction [4]
2. Logic Programming & Logic Programs
3. Deduction via Abduction [2]
4. A Framework for Abduction Problems [3]

[1] K. L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Data Bases, Symposium on Logic and Data Bases, Centre d'études et de recherches de Toulouse, 1977.*, Advances in Data Base Theory, pages 293–322, New York, 1977. Plemum Press. ISBN 0-306-40060-X.

[2] L. Console, D. T. Dupré, and P. Torasso. On the relationship between abduction and deduction. *J. Log. Comput.*, 1(5): 661–690, 1991. doi: 10.1093/logcom/1.5.661. URL http://dx.doi.org/10.1093/logcom/1.5.661.

[3] T. Eiter, G. Gottlob, and N. Leone. Abduction from logic programs: Semantics and complexity. *Theor. Comput. Sci.*, 189(1-2):129–177, 1997. doi: 10.1016/S0304-3975(96)00179-X. URL http://dx.doi.org/10.1016/S0304-3975(96)00179-X.

[4] C. S. Peirce. Illustrations of the Logic of Science VI: Deduction, Induction, and Hypothesis. *The Popular Science Monthly*, 13, 1878.