Abduction and Induction through Inverse Entailment and Consequence Finding

Katsumi Inoue

National Institute of Informatics, Japan

AIAI'05
Edinburgh, July 29, 2005

Consequence Finding

- Given an axiom set, the task of *consequence finding* or *theorem finding* is to find out some theorems of interest [Lee, 1967].
- Theorems to find out are not given in an explicit way, but are characterized by some properties.
- Task is clearly distinguished from *proof finding* or *theorem proving*. However, theorem proving is a special case of consequence finding.
- Consequence finding is a part of deduction, but can be used for abduction and induction.

Abduction and Induction: Logical Framework

Input:

- B : background theory
- E : (positive) examples ／observations

Output:

- H : hypothesis satisfying that
  - $B \land H \models E$
  - $B \land H$ is consistent.

Abduction and Induction: Logical Framework

- $B \land H \models E$
- $B \land H$ is consistent.

- The logical framework is exactly the same.
- A different formalism exists for induction, e.g., *descriptive induction*, but can be unified with the above framework [Inoue & Saito, ILP'04].
- Induction often gets negative examples, but abduction can be extended too [Inoue & Sakama, IJCAI-95].
- Theoretical results for one can be easily transferred to the other. E.g., The notion of **equivalence** is explored for abduction [Inoue & Sakama, MBR'04; IJCAI-05] and for induction [Sakama & Inoue, ILP'05].
- Computation can also be unified.

Inverse Entailment

Given that

$$B \land H \models E,$$

computing a hypothesis H can be done by

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

I.e., $\neg H$ deductively follows from $B \land \neg E$.

Inverse Entailment

$$\boldsymbol{B}: \quad Human(Socrates),$$

$$\boldsymbol{E}: \quad Mortal(Socrates),$$

$$\boldsymbol{H}: \quad \forall x\,(\,Human(x) \supset Mortal(x)\,)$$

satisfies that:

$$B \land H \models E\,.$$

In fact,

$$B \land \neg E = Human(Socrates) \land \neg Mortal(Socrates)$$

$$\models \exists x\,(\,Human(x) \land \neg Mortal(x)\,) = \neg H\,.$$

IE for Abduction [Inoue, 1992]

$$B \land \neg E \models \neg H$$

- Computation through *consequence finding*
  - E : conjunction of (existentially-quantified) **literals**
  - H : conjunctions of **literals**
  - B : (full) clausal theory (**non-Horn** clauses)
- Note: Both $\neg E$ and $\neg H$ are **clauses**.
- sound and complete

S O L A R

[Nabeshima, Iwanuma & Inoue, 2003]

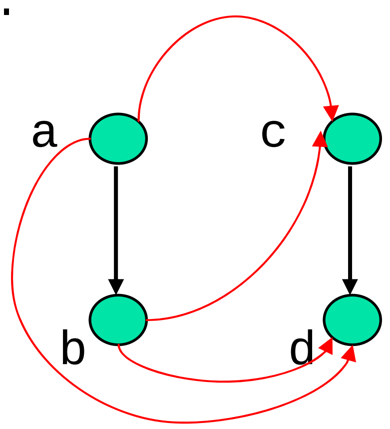
- **Input**: (1) the set of input clauses (**TPTP** library format), (2) the top clause, (3) the production field, and (4) the search strategy.
- For example, the **graph completion problem** (find an arc which enables a path from a to d) is described as follows:

```
input_clause(edge, axiom, [node(a)]).
input_clause(edge, axiom, [node(b)]).
input_clause(edge, axiom, [node(c)]).
input_clause(edge, axiom, [node(d)]).
input_clause(arc, axiom, [arc(a,b)]).
input_clause(arc, axiom, [arc(c,d)]).
input_clause(path, axiom,
                [-node(X), -node(Y), -arc(X,Y), path(X,Y)]).
input_clause(path, axiom, [-node(X), -node(Y), -node(Z),
              -arc(X,Y), -path(Y,Z), path(X,Z)]).
input_clause(observation, top_clause, [-path(a,d)]).

production_field([predicates([-arc(_,_)])]).
```

- SOLAR outputs four consequences:

```
[-arc(a, d)]
[-arc(a, c)]
[-arc(b, d)]
[-arc(b, c)]
```



IE for ILP [Muggleton, 1995]

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

- Use *consequence-finding* procedures twice
  [Yamamoto 1997]

- *B* : **Horn** clausal theory
- *E* : single **Horn clause**
- *H* : single (non-)Horn **clause**

- Note: Neither $\neg E$ nor $\neg H$ is a single clause, and both contain **existentially quantified variables**.

IE with ⊥-clause: Incompleteness

Approach: Compute the ***⊥-clause***:

$$\bot(B, E) = \{ \neg L \mid L \text{ is a literal s.t. } B \wedge \neg E \models L \}.$$

Hypothesis H is constructed by generalizing ⊥-clause:

$$H \models \bot(B, E).$$

- Sound but incomplete for recursive clauses [Yamamoto, 1997]
- Sufficient conditions for completeness [Furukawa et al., 1997; Yamamoto, 1997;1999]
- Incompleteness due to single-clause hypotheses [Ray, 2003]

Complete Calculus for IE

$$B \land \neg E \models \neg H$$

CF-Induction [Inoue, 2001]

- Compute the ***characteristic clauses*** of $B \land \neg E$
- Use any *consequence-finding* procedure.
- Use any *generalizer*.
- Includes the bottom method and abductive computation.

- B : full clausal theory (non-Horn clauses)
- E : full clausal theory (non-Horn clauses)
- H : full clausal theory (non-Horn clauses)

- Sound and complete

CF-Induction: Principle

$$B \wedge H \models E$$

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

$$\Leftrightarrow B \wedge \neg E \models Carc(B \wedge \neg E, \boldsymbol{P}) \models CC(B,E) \models \neg H$$

$$\Leftrightarrow CC(B,E) \subseteq Carc(B \wedge \neg E, \boldsymbol{P}),$$

$$\neg CC(B,E) \equiv F, \quad H \models F \quad \text{(where } F \text{ is CNF)}$$

CF-Induction: Algorithm

1. Compute $Carc(B \wedge \neg E, \boldsymbol{P})$.
2. Construct $CC(B,E)$ such that
   - $CC(B,E) \subseteq Carc(B \wedge \neg E, \boldsymbol{P})$;
   - $CC(B,E) \cap NewCarc(B, \neg E, \boldsymbol{P}) \neq \phi$.
3. Convert $\neg CC(B,E)$ into CNF F .
4. Generalize F to H such that
   - $B \wedge H$ is consistent.

CF-Induction: Generalizers

Given a CNF formula F, find a CNF formula H such that

$$H \models F.$$

- inverse Skolemization
- anti-instantiation
- anti-subsumption (dropping literals from clauses)
- anti-weakening (addition of clauses)
- inverse resolution
- Plotkin's least generalization

CF-Induction: Buntine's Example

$\boldsymbol{B}$: $cat(x) \supset pet(x),$

$small(x) \wedge fluffy(x) \wedge pet(x) \supset cuddly\_pet(x).$

$\boldsymbol{E}$: $fluffy(x) \wedge cat(x) \supset cuddly\_pet(x).$

$NewCarc(B, \neg E, \boldsymbol{P})$:

$fluffy(s_x), cat(s_x), \neg cuddly\_pet\ (s_x),$

$pet(s_x), \neg small(s_x)$

$CC(B,E) = NewCarc(B, \neg E, \boldsymbol{P})$

$\boldsymbol{H}$: $fluffy(x) \wedge cat(x) \wedge pet(x) \supset cuddly\_pet(x) \vee small(x)$

CF-Induction: Yamamoto's Example

B: *even(0),*
odd(x) ⊃ even(s(x)).
E: *odd(s(s(s(0)))).*

NewCarc(B,¬E, ***P****):* *¬odd(s(s(s(0)))).*

CC(B,E) : *even(0), odd(s(0))⊃even(s(s(0))), ¬odd(s(s(s(0)))).*

CNF(*¬CC(B,E)* **):**

even(0) ⊃ odd(s(0)) ∨ odd(s(s(s(0)))),
even(0) ∧ even(s(s(0))) ⊃ odd(s(s(s(0)))).

H: *even(x) ⊃ odd(s(x)).*

Yamamoto & Fronhőfer's Example

B: $dog(x) \wedge small(x) \supset pet(x)$.

E: $pet(c)$.

NewCarc(B,¬E, ***P****)*: $\neg pet(c)$, $\neg dog(c) \vee \neg small(c)$.

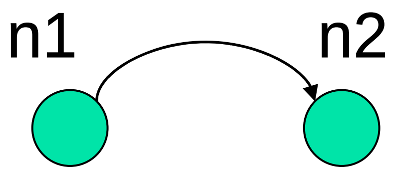
CC(B,E) = NewCarc(B,¬E, ***P****)*.

$\neg CC(B,E)$ **:** $pet(c) \vee (dog(c) \wedge small(c))$.

CNF($\neg CC(B,E)$ **):** $pet(c) \vee dog(c)$, $pet(c) \vee small(c)$.

H: $pet(x) \vee dog(x)$, $pet(x) \vee small(x)$.

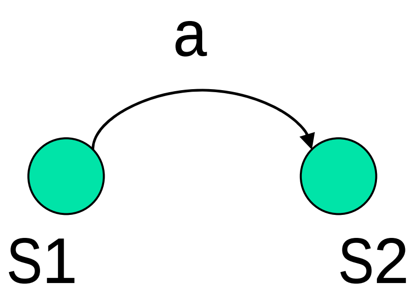
Abduction and Induction in Network Inference



abduction: path(n1,n2)

arc(X,Y) $\supset$ path(X,Y)

$\therefore$ arc(n1,n2)



induction: *Q* ***after*** a

$\therefore$ *a* ***causes*** *Q* ***if*** *P*

where $P \subseteq$ S1, $Q \subseteq$ S2

[Inoue, Bando & Nabeshima, ILP'05]