Declarative Problem Solving through Abduction

Antonis C. Kakas
Department of Computer Science
University of Cyprus
CYPRUS (CHYPRE)
antonis@ucy.ac.cy
(Subject Title: “DPS-Paris2007”)

PART 3

11-18 January, 2007

Paris, France
Lecture 2
Abductive Logic Programming (ALP)

- An ALP theory (or model) of a domain is \(<T, H, IC>:\)
  
  - **T** (theory presentation) is a *normal logic program*
  - **H** (candidate hypotheses) is a set of *undefined predicates*
  - **IC** (integrity constraints) is a set of (FOL) sentences

- **O** (observation or goal) is a conjunction of **literals**

- **Semantics:** Given \(<T, H, IC>\) \(E\) is an *explanation* of \(O\) iff
  1) \(T \cup E\) entails \(O\)
  2) \(T \cup E\) satisfies \(IC\)
  3) \(E \subseteq \text{Ground}(H)\)

  where entails and satisfies are *model theoretic notions*, e.g. truth in a canonical model of \(T \cup E\).
Abductive Logic Programming (ALP)

Given $T= \langle P, H, IC \rangle$ where $P$ is a definite logic program then $E$ is an explanation of $O$ iff:

1) $O$ is true in $M$;
2) $IC$ are true in $M$ (epistemic view)
3) $E \subseteq G(H)$

where $M$ is the least Herbrand model of $P \cup E$;

When $P$ is a normal logic program, i.e. containing NAF, then $M$ is any stable model of $P \cup E$ in the above definition.
Abductive Logic Programming (ALP)

- Given $T = \langle P, H, IC \rangle$ where $P$ is a normal logic program then $M$ is a Generalized Model or Extension of $T$ iff there exists $E \subseteq G(H)$ such that:
  
  1) $M$ is a stable model of $P \cup E$
  2) $IC$ are classically true in $M$

- Given $T = \langle P, H, IC \rangle$ a goal/query, $G$, is:
  
  - credulous entailed by $T$ iff there exists an extension $M$ of $T$ such that $M \models G$.
  - sceptically entailed by $T$ iff for every extension $M$ of $T$, $M \models G$. 
Course Breakdown

• Introduction
• Abduction – General Introduction
• Modelling Problems for Abduction and DPS

• Computational Logic & PROLOG – Background

• Abductive Logic Programming – Semantics
• Abductive Logic Programming – Computation
• ALP for Declarative Problem Solving – Diagnosis

• Projects – Discussion

cnt...
Lecture 3: Knowledge Representation in ALP for Diagnosis

• **Aim:** To show how the problem of Diagnosis can be addressed within our approach of modelling within the framework of ALP.
What is Diagnosis?

- Diagnosis is the problem of trying to find what is wrong with some system based on:
  - knowledge about the design/structure of the system
  - possible malfunctions in the system
  - observations of the behaviour of the system.

- Compare with modelling in ALP!
Approaches to Diagnosis

- **Two (extreme) approaches to diagnosis:**

  - **Normal Model:** Knowledge is about how components work normally. There is no knowledge as to how malfunctions occur and manifest themselves.
    - **Diagnosis:** isolate the deviations from normal behaviour, i.e. which components of the system are broken.
    - **Called:** Consistency-based diagnosis

  - **Abnormal/Anomalous Model:** Knowledge about faults and their symptoms. We have knowledge about the faulty operation of the system.
    - **Diagnosis:** (abductive) explanation of the abnormal observations.
    - **Called:** Abductive diagnosis
Diagnosis for Digital Circuits

- Our knowledge consists of:
  - A Description of the circuit
  - Truth tables of the gates (Normal Behaviour)
  - General laws of Logic Circuits

- Description of a Circuit:
  - Names of devices, e.g. x1,x2,...,a1,a2, ...  
  - Names of inputs/outputs to a device,  
    e.g. in(1,x1) names the first input to device x1  
    in(2,x1) names the second input to device x1  
    etc  
    out(1,x1) names the first output from device x1  
    out(2,x1) names the second output from device x1  
    etc
Diagnosis for Digital Circuits

- **Description of a Circuit:**
  - **Predicates stating the type of a device, e.g.:**
    - `gate(x1,xor)`
    - `gate(x2,xor)`
    - `gate(a1,and)`
    - `gate(o1,or)`
  - **Predicates describing the connections in the circuit, e.g.:**
    - `connect(input(1),in(1,x1))`
    - `connect(out(1,x1),in(1,x2))`
    - `connect(out(1,x2),output(1))`
Diagnosis for Digital Circuits

- **Truth Tables**
  - `ttable(Type,In1,In2,Out)`
    - `ttable(and,on,on,on)`
    - `ttable(and,off,any,off)`
    - `ttable(and,any,off,off)`
    - `ttable(or,off,off,off)`
    - `ttable(or,on,any,on)`
    - `ttable(or, any,on,on)`
    - `etc`
Diagnosis for Digital Circuits

- **Causal operation of circuits:**
  - Observable Predicate: `value(Node,Value)`

  - `value(out(1,Device), Out1):-`
    - `gate(Device,Type),`  
    - `value(in(1,Device), In1),`  
    - `value(in(2,Device),In2),`  
    - `ttable(Type, In1, In2, Out1),`  
    - `normal(Device).`  

  - `value(Node,Val):-`  
    - `connect(Node1,Node),`  
    - `value(Node1, Val).`
Diagnosis for Digital Circuits

- Normal operation of circuits:
  - `value(out(1,Device), Out):-
    gate(Device,Type),
    value(in(1,Device), In1),
    value(in(2,Device),In2),
    ttable(Type, In1, In2, Out),
    normal(Device).`

- Abnormal/anomalous operation of circuits:
  - `value(out(1,Device), EOut):-
    gate(Device,Type),
    value(in(1,Device), In1),
    value(in(2,Device),In2),
    ttable(Type, In1, In2, NOut),
    abnormal(Device),
    opposite(NOut,EOut).`
Diagnosis for Digital Circuits

**Summary:** The ALP $<P,H,IC>$ representation is:

- **Observable Predicate:** `value(Node,Value)`
- **Abducible Predicate:** `normal(Device)`

**Program, $P$, of the causal mechanism:**
- `value(out(1,Device), Out1):-
  gate(Device,Type),
  value(in(1,Device), In1),
  value(in(2,Device),In2),
  ttable(Type, In1, In2, Out1),
  normal(Device).`
- etc

**Integrity constraints, $IC$,:**
- `value(Node, Val1), value(Node, Val2) => Val1 = Val2`
- `normal(Device), abnormal(Device) => false`
Example Circuit: Circuit 1

\[ F = X + Y'Z \]

Given:
- value(in(1,circ1), off)
- value(in(2,circ1), on)
- value(in(3,circ1), off)

Observation:
- value(out(1,circ1), on)
Example Circuit: Circuit 1

Given:
- value(in(1, circ1), off)
- value(in(2, circ1), on)
- value(in(3, circ1), off)
- value(out(1, circ1), on)

Explanation/Diagnosis:
- E1 = \{normal(g3), abnormal(g2), normal(g1)\}
- E2 = \{normal(g3), abnormal(g2), abnormal(g1)\}
- E3 = \{abnormal(g3), normal(g2), normal(g1)\}

Can all three components be abnormal?

F = X + Y'Z
Example Circuit: Circuit 1

\[ F = X + Y'Z \]

Explanation/Diagnosis:
- E1 = \{normal(g3), abnormal(g2), normal(g1)\}
- E2 = \{normal(g3), abnormal(g2), abnormal(g1)\}
- E3 = \{abnormal(g3), normal(g2), normal(g1)\}

- Prefer E1 over E2 as it is minimal in abnormality.
- How would we distinguish between E1 and E3?
  - Get additional observation of the output of g2.

Obs={value(out(1,circ1), on), value(out(1,g2), on)}
- E'1 ={abnormal(g2), normal(g3), normal(g1)}
- E'2 ={abnormal(g2), normal(g3), abnormal(g1)}

are the only explanations.

We prefer E’1 as it is minimal in abnormality.
Normally gates are ok

The default assumption is that gates are normal. We can use NAF to represent this:

- \( \text{value(out(1,Device), Out1)} :\)
  \( \text{gate(Device,Type),} \)
  \( \text{value(in(1,Device), In1),} \)
  \( \text{value(in(2,Device),In2),} \)
  \( \text{ttable(Type, In1, In2, Out1),} \)
  \( \text{not abnormal(Device).} \)

- \( \text{value(out(1,Device), EOut)} :\)
  \( \text{gate(Device,Type),} \)
  \( \text{value(in(1,Device), In1),} \)
  \( \text{value(in(2,Device),In2),} \)
  \( \text{ttable(Type, In1, In2, NOut),} \)
  \( \text{abnormal(Device),} \)
  \( \text{opposite(NOut,EOut).} \)
Example Circuit: Circuit 1 (with NAF)

Given:
- value(in(1,circ1), off)
- value(in(2,circ1), on)
- value(in(3,circ1), off)
- value(out(1,circ1), on)

Explanation/Diagnosis:
- $E_1 = \{\text{abnormal}(g2)\}$
- $E_2 = \{\text{abnormal}(g2), \text{abnormal}(g1)\}$
- $E_3 = \{\text{abnormal}(g3)\}$

$E_1$ and $E_3$ are minimal.
Example Circuit: Circuit 1 (with NAF)

Explanation/Diagnosis:
- E1 = \{abnormal(g2)\}
- E2 = \{abnormal(g2), abnormal(g1)\}
- E3 = \{abnormal(g3)\}

Credulous and Sceptical Conclusions:
- e.g. abnormal(g2) is credulous
- e.g. value(out(1,g2), on) is credulous
- e.g. value(out(1,g2), off) is credulous
- No sceptical conclusions except the observation: value(out(1,circ1), on)
Example Circuit 1: Further Assertional Knowledge

- **Gate g2 is ok:**
  - New ic: abnormal(g2) => false
  - Explanation/Diagnosis:
    - E3 = {abnormal(g3)}

- **Gate g2 is ok when its first input is off:**
  - New ic: value(in(1,g2), off) => ¬abnormal(g2)
  - Explanation/Diagnosis:
    - E = {abnormal(g3)}
    - E = {abnormal(g1), abnormal(g2)}

- **The two inputs of gate g2 always the same:**
  - New ic: value(in(1,g2), In1), value(in(1,g2), In2) => In1=In2
  - Explanation/Diagnosis:
    - E = {abnormal(g3)}
    - E = {abnormal(g2)}

\[ F = X + Y'Z \]
Fault Models in Circuits

- Our model so far assumed that a faulty device, abnormal(D), is faulty only for the input that is given at the time. There is no assumption that faulty gates always produce the wrong output.

- Anomalous fault models. In practice, gates fail by either:
  - Outputs being stuck on or off
  - Inputs being stuck on or off.

- We can incorporate fault models to describe in a refined way the faulty behaviour.
Description of Faulty Behaviour

- Abnormal predicate has an extra argument now:
  - abnormal(Device,FaultStatus)
  - FaultStatus takes values:
    - stuckon – output is on nomatter
    - stuckoff – output is off nomatter
    - stuck_in1 – input 1 is stuck to make the gate act like a wire from input 2
    - stuck_in2 – input 2 is stuck ...

- We introduce a new predicate fault(Type,FaultStatus,In1,In2,Out) to describe the faulty behaviour:
  - fault(Type,stuckon,any,any,on)
  - fault(Type,stuckoff,any,any,off)
  - fault(Type,stuck_in1,any,on,off)
  - fault(Type,stuck_in1,any,off,off)
Description of Faulty Behaviour (cnt.)

- The logic program, P, in our ALP theory \(<P,H,IC>\) is:

  - value(out(1,Device), Out1):-
    gate(Device,Type),
    value(in(1,Device), In1),
    value(in(2,Device),In2),
    ttable(Type, In1, In2, Out1),
    not abnormal(Device,FStatus).

  - value(out(1,Device), FOut):-
    gate(Device,Type),
    value(in(1,Device), In1),
    value(in(2,Device),In2),
    abnormal(Device, FStatus),
    fault(Type, FStatus,In1, In2, FOut).
Multiple Observations (experiments)

- We introduce a time stamp on our observations:
  
  - `value(out(1,Device), Out1, Time):-
    gate(Device,Type),
    value(in(1,Device), In1, Time),
    value(in(2,Device), In2, Time),
    ttable(Type, In1, In2, Out1),
    not abnormal(Device,FStatus).`
  
  - `value(out(1,Device), FOut, Time):-
    gate(Device,Type),
    value(in(1,Device), In1, Time),
    value(in(2,Device), In2, Time),
    abnormal(Device, FStatus),
    fault(Type, FStatus,In1, In2, FOut).`

- Together with the integrity constraint:
  
  - `abnormal(Dev,FSt1), abnormal(Dev,FSt2) => FSt1 = FSt2`
Example Circuit: Circuit 1

Multiple Observations/Experiments:

- **t1:** value(in(1,circ1), off, t1), value(in(2,circ1), on, t1), value(in(3,circ1), off, t1) and value(out(1,circ1), off, t1)
- **t2:** value(in(1,circ1), off, t2), value(in(2,circ1), off, t2), value(in(3,circ1), on, t2) and value(out(1,circ1), on, t2)
- **t3:** value(in(1,circ1), off, t3), value(in(2,circ1), off, t3), value(in(3,circ1), off, t3) and value(out(1,circ1), off, t3)

**Explain Obs:** \{value(out(1,circ1), off, t1), value(out(1,circ1), on, t2), value(out(1,circ1), off, t3)\}

**Explanation E =** \{abnormal(g3, suckin_in1)\}

\[ F = X + Y'Z \]