Table of contents

Minimal Model Reasoning

- Conflicts between defaults in Default Logic lead to multiple extensions.
- Each extension corresponds to a maximal set of non-violated defaults.
- Reasoning with defaults can also be achieved by a simpler mechanism: predicate or propositional logic + minimize the number of cases where a default (expressed as a conventional formula) is violated $\Longrightarrow$ minimal models.
- Notion of minimality: cardinality vs. set-inclusion.

Entailment with respect to Minimal Models

Definition

Let A be a set of atomic propositions. Let Φ be a set of propositional formulae on A, and $B \subseteq A$ a set of abnormalities.
Then $\Phi \models_B \psi$ (ψ B-minimally follows from Φ) if $\mathcal{I} \models \psi$ for all interpretations $\mathcal{I}$ such that $\mathcal{I} \models \Phi$ and there is no $\mathcal{I}'$ such that $\mathcal{I}' \models \Phi$ and $\{b \in B | \mathcal{I}' \models b\} \subset \{b \in B | \mathcal{I} \models b\}$.

Minimal models: example

$$\Phi = \left\{ \begin{array}{ll} \text{student} \wedge \neg\text{ABstudent} \rightarrow \neg\text{earnsmoney}, & \text{student}, \\ \text{adult} \wedge \neg\text{ABadult} \rightarrow \text{earnsmoney}, & \text{student} \rightarrow \text{adult} \end{array} \right\}$$

Φ has the following models.

$\mathcal{I}_1 \models$ student $\wedge$ adult $\wedge$ earnsmoney $\wedge$ ABstudent $\wedge$ ABadult
$\mathcal{I}_2 \models$ student $\wedge$ adult $\wedge$ $\neg$earnsmoney $\wedge$ ABstudent $\wedge$ ABadult
$\mathcal{I}_3 \models$ student $\wedge$ adult $\wedge$ earnsmoney $\wedge$ ABstudent $\wedge$ $\neg$ABadult
$\mathcal{I}_4 \models$ student $\wedge$ adult $\wedge$ $\neg$earnsmoney $\wedge$ $\neg$ABstudent $\wedge$ ABadult

Relation to Default Logic

We can embed propositional minimal model reasoning in the propositional Default Logic.

Theorem

Let A be a set of atomic propositions. Let Φ be a set of propositional formulae on A, and $B \subseteq A$.
Then $\Phi \models_B \psi$ if and only if ψ follows from $\langle D, W \rangle$ skeptically, where

$$D = \left\{ \frac{:\neg b}{\neg b} \middle| b \in B \right\} \text{ and } W = \Phi.$$

Relation to Default Logic: Proof

Proof sketch.

$\Rightarrow$ Assume there is extension E of $\langle D, W \rangle$ such that $\psi \notin E$. Hence there is an interpretation $\mathcal{I}$ such that $\mathcal{I} \models E$ and $\mathcal{I} \models \neg\psi$.
By the fact that there is no extension F such that $E \subset F$, $\mathcal{I}$ is a B-minimal model of Φ. Hence ψ does not B-minimally follow from Φ.
$\Leftarrow$ Assume ψ does not B-minimally follow from Φ. Hence there is an B-minimal model $\mathcal{I}$ of Φ such that $\mathcal{I} \not\models \psi$. Define $E = \text{Th}(\Phi \cup \{\neg b | b \in B, \mathcal{I} \models \neg b\})$. Now $\mathcal{I} \models E$ and because $\mathcal{I} \not\models \psi$, $\psi \notin E$.
We can show that E is an extension of $\langle D, W \rangle$.
Because there is extension E such that $\psi \notin E$, ψ does not skeptically follow from $\langle D, W \rangle$. □

Nonmonotonic Logic Programs: Background

- Answer set semantics: a formalization of negation-as-failure in logic programming (Prolog)
- Other formalizations: well-founded semantics, perfect-model semantics, inflationary semantics, ...
- Can be viewed as a simpler variant of *default logic*.
- A better alternative to *the propositional logic* in some applications.

Nonmonotonic Logic Programs

- Rules $c \leftarrow b_1, \ldots, b_m, \text{not } d_1, \ldots, \text{not } d_k$ where $\{c, b_1, \ldots, b_m, d_1, \ldots, d_k\} \subseteq A$ for a set $A = \{a_1, \ldots, a_n\}$ of propositions.
- Meaning similar to default logic: If
  1. we have derived $b_1, \ldots, b_m$ and
  2. cannot derive any of $d_1, \ldots, d_k$,

  then derive c.
- Rules without right-hand side: $c \leftarrow$
- Rules without left-hand side: $\leftarrow b_1, \ldots, b_m, \text{not } d_1, \ldots, \text{not } d_k$

Answer Sets – Formal Definition

- Reduct P^Δ of a program P with respect to a set of atoms $\Delta \subseteq A$:

$$\{c \leftarrow b_1, \ldots, b_m \mid (c \leftarrow b_1, \ldots, b_m, \text{not } d_1, \ldots, \text{not } d_k) \in P, \{d_1, \ldots, d_k\} \cap \Delta = \emptyset\}$$

- *Closure dcl*$(P) \subseteq A$ of a set P of rules without **not** is defined by iterative application of the rules in the obvious way.
- A set of propositions $\Delta \subseteq A$ is an answer set of P iff $\Delta = \text{dcl}(P^\Delta)$.

Examples

- $P_1 = \{a \leftarrow, \quad b \leftarrow a, \quad c \leftarrow b\}$
- $P_2 = \{a \leftarrow b, \quad b \leftarrow a\}$
- $P_3 = \{p \leftarrow \text{not } p\}$
- $P_4 = \{p \leftarrow \text{not } q, \quad q \leftarrow \text{not } p\}$
- $P_5 = \{p \leftarrow \text{not } q, \quad q \leftarrow \text{not } p, \quad \leftarrow p\}$

Complexity: existence of answer sets is NP-complete

1. Membership in NP: Guess $\Delta \subseteq A$ (*nondet. polytime*), compute P^Δ, compute its closure, compare to Δ (*everything det. polytime*).
2. NP-hardness: Reduction from 3SAT: an answer set exists iff clauses are satisfiable:

$$p \leftarrow \text{not } \hat{p}$$
$$\hat{p} \leftarrow \text{not } p$$

for every proposition p occurring in the clauses, and

$$\leftarrow \text{not } l'_1, \text{not } l'_2, \text{not } l'_3$$

for every clause $l_1 \vee l_2 \vee l_3$, where $l'_i = p$ if $l_i = p$ and $l'_i = \hat{p}$ if $l_i = \neg p$.

Programs for Reasoning with Answer Sets

- smodels (Niemelä & Simons), dlv (Eiter et al.), ...
- Schematic input:

```
p(X) :- not q(X).    anc(X,Y) :- par(X,Y).
q(X) :- not p(X).    anc(X,Y) :- par(X,Z), anc(Z,Y).
r(a).                par(a,b). par(a,c). par(b,d).
r(b).                female(a).
r(c).                male(X) :- not(female(X)).
                     forefather(X,Y) :-
                               anc(X,Y), male(X).
```

Difference to the Propositional Logic

- The *ancestor* relation is the transitive closure of the *parent* relation.
- Transitive closure cannot be (concisely) represented in propositional/predicate logic.

  par(X,Y) → *anc(X,Y)*
  par(X,Z) ∧ *anc(Z,Y)* → *anc(X,Y)*

  The above formulae only guarantee that *anc* is a *superset* of the transitive closure of *par*.
- For transitive closure one needs the minimality condition in some form: nonmonotonic logics, fixpoint logics, ...

Stratification

The reason for multiple answer sets is the fact that a may depend on b and simultaneously b may depend on a.
The lack of this kind of circular dependencies makes reasoning easier.

Definition
A logic program P is stratified if P can be partitioned to $P = P_1 \cup \cdots \cup P_n$ so that for all $i \in \{1, \ldots, n\}$ and $(c \leftarrow b_1, \ldots, b_m, \text{not } d_1, \ldots, \text{not } d_k) \in P_i$,

1. there is no not c in P_i and
2. there are no occurrences of c anywhere in $P_1 \cup \cdots \cup P_{i-1}$.

Stratification

Theorem
A stratified program P has exactly one answer set. The unique answer set can be computed in polynomial time.

Example
Our earlier examples with more than one or no answer sets:

$$P_3 = \{p \leftarrow \text{not } p\}$$
$$P_4 = \{p \leftarrow \text{not } q, \quad q \leftarrow \text{not } p\}$$

Applications of Logic Programs

1. Simple forms of default reasoning (inheritance networks)
2. A solution to the frame problem: instead of using frame axioms, use defaults

   $$a_{t+1} \leftarrow a_t, \text{not } \neg a_{t+1}$$

   By default, truth-values of facts stay the same.
3. deductive databases (Datalog$^{\neg}$)
4. et cetera: Everything that can be done with propositional logic can also be done with propositional nonmotononic logic programs.

Literature

M. Gelfond and V. Lifschitz, The stable model semantics for logic programming, *Proceedings of the Fifth International Conference on Logic Programming*, The MIT Press, 1988.

I. Niemelä and P. Simons. Smodels - an implementation of the stable model and well-founded semantics for normal logic programs, *Proceedings of the 4th International Conference on Logic Programming and Non-monotonic Reasoning*, 1997.

T. Eiter, W. Faber, N. Leone, and G. Pfeifer. Declarative problem solving using the dlv system. In J Minker, editor, Logic Based AI, Kluwer Academic Publishers, 2000.