(Default) Negation for Logic Programming in Algebraic Domains

Pascal Hitzler

Dagstuhl Seminar 02221, Schloss Dagstuhl, May 2002

Slide 1

Background

[RZ01] William C. Rounds and Guo-Qiang Zhang,
Clausal Logic and Logic Programming in Algebraic Domains,
Information and Computation 171(2) (2001) 156–182.

Notion of resolution / logic programming in coherent algebraic domains.

Artificial Intelligence Institute, Dresden University of Technology, Dresden, Germany

Slide 2

Contents

We explore:

How does [RZ01] relate to established theory in logic programming and nonmonotonic reasoning?

▶ Resolution Theorem

Obtain analogue to: $T \models X$ iff $T \cup \{\neg X\} \vdash \{\}$.
(T theory, X clause, $\{\}$ empty clause)

▶ Default negation

(more later)

Thanks for discussions to
William C. Rounds, Anthony K. Seda, Pawel Waszkiewicz, Guo-Qiang Zhang.

Slide 3

Coherent algebraic cpos

A *cpo* $(D, \sqsubseteq)$ is a poset with bottom element $\perp$
such that every directed subset A of D has a supremum $\bigsqcup A$.

$c \in D$ is *compact* if whenever $c \sqsubseteq \bigsqcup L$ with L directed
then there exists $e \in L$ with $c \sqsubseteq e$.

$\mathsf{K}(D)$ is the set of all compact elements of D.

A cpo is *algebraic* if for every $e \in D$ we have $e = \bigsqcup\{c \in \mathsf{K}(D) \mid c \sqsubseteq e\}$.

Slide 4

Coherent algebraic cpos

An algebraic cpo is *coherent* if
the intersection of any two Scott-compact-open sets is compact-open.

Coherency implies: If $a_1, \dots, a_n \in \mathsf{K}(D)$ then the set $\mathrm{mub}\{a_1, \dots, a_n\}$
of all minimal upper bounds of the a_i is finite.

Slide 5

Intuition

Standard example: $\mathbb{T}^\omega = \mathbb{T}^\mathcal{V}$

$\mathbb{T} = \{\mathbf{f}, \mathbf{u}, \mathbf{t}\}$.
$\mathcal{V}$ countable set of propositional variables.
$\mathbb{T}^\omega$ set of all interpretations in three-valued logic.
Typical compact element in $\mathbb{T}^\mathcal{V}$: $p \wedge \overline{q} \wedge r$

Clauses are finite sets of compact elements in $\mathbb{T}^\mathcal{V}$.
Clauses are disjunctive normal forms.

analogon to negation in domains: inconsistency

Slide 6

Clausal logic on coherent algebraic cpos

Clauses: finite subsets of $\mathsf{K}(D)$. $\{\}$ empty clause.

Theory: set of clauses. $\emptyset$ empty theory.

In the following: T, S theories, X clause, $w \in D$.

$w \models X$ if there exists $x \in X$ with $x \sqsubseteq w$.
$w \models T$ if $w \models X$ for all $X \in T$.
$T \models X$ if $w \models T$ implies $w \models X$.

T *logically closed* if $T \models X$ implies $X \in T$.
T *consistent* if $T \not\models \{\}$.

Slide 7

Some results from [RZ01]

The set of all consistent closed theories over D, ordered by inclusion, is isomorphic to the collection of all non-empty Scott-compact saturated subsets of D, ordered by reverse inclusion.

A theory is logically closed if and only if it is an ideal with respect to the Smyth preorder.

A clause is a logical consequence of a theory if and only if it is a logical consequence of a finite subtheory.

S is *saturated* if it is the intersection of all Scott-open sets which contain it, i.e. it is upward-closed.
Ideals are directed downward-closed sets.
Smyth preorder: $X \sqsubseteq^\sharp Y$ iff for all $y \in Y$ exists $x \in X$ with $x \sqsubseteq y$.

Slide 8

A Sound and Complete System

"Move downwards"

$$\frac{X; \qquad \{a, y\} \subseteq X; \qquad y \sqsubseteq a}{X \setminus \{a\}}$$

$$\frac{X; \qquad y \in \mathsf{K}(D)}{\{y\} \cup X}$$

"Move upwards"

$$\frac{X_1 \quad X_2; \qquad a_1 \in X_1 \quad a_2 \in X_2}{\mathrm{mub}\{a_1, a_2\} \cup (X_1 \setminus \{a_1\}) \cup (X_2 \setminus \{a_2\})}$$

Slide 9

Alternative Set of Rules

Hyperresolution rule (hr)

$$\frac{X_1 \quad X_2 \quad \ldots \quad X_n; \qquad a_i \in X_i \text{ for } 1 \leq i \leq n; \qquad \mathrm{mub}\{a_1, \ldots, a_n\} \models Y}{Y \cup \bigcup_{i=1}^{n} (X_i \setminus \{a_i\})}$$

$$\frac{\{\}; \qquad Y \text{ clause}}{Y}$$

(hr) contains resolution.

$$\frac{X_1 \quad X_2; \qquad a_1 \in X_1 \quad a_2 \in X_2; \qquad a_1 \not\uparrow a_2}{(X_1 \setminus \{a_1\}) \cup (X_2 \setminus \{a_2\})} \qquad \text{(r)}$$

Slide 10

Resolution Theorem

Important for Prolog-style logic programing: Resolution proofs.

$$T \models X \text{ iff } T \cup \{\neg X\} \vdash \{\}$$

In order to obtain a similar result in algebraic domains
we need to have forms of

- ▶ distributivity
- ▶ negation

in the domain.

Slide 11

Enforcing Distributivity

$a \in D$ is an *atom* if whenever $x \sqsubseteq a$ then $x = a$ or $x = \bot$.
$\mathsf{A}(D)$ set of all atoms of D.

D (coherent algebraic cpo) is an *atomic domain* if:

- $\mathsf{A}(c) = \{p \in \mathsf{A}(D) \mid p \sqsubseteq c\}$ is finite for compact c and
- $c = \bigsqcup \mathsf{A}(c)$.

Obtain representations of clause as set of *atomic* clauses.

$$X \qquad \sim \qquad \{\{b_1, \ldots, b_n\} \mid b_i \in \mathsf{A}(a_i) \text{ for all } i = 1, \ldots, n\}$$

Slide 12

A Form of Negation

An atomic domain is *negated* if there exists a

- Scott-continuous involution $^{-} : D \to D$ with:
- $^{-}$ maps $\mathsf{A}(D)$ onto $\mathsf{A}(D)$
- $\bigsqcup A$ exists for all finite $A \subset \mathsf{A}(D)$ with pairwise consistent elements.
- $p \not\uparrow q$ if and only if $q = \overline{p}$ (for all $p, q \in \mathsf{A}(D)$).

T theory, X atomic clause. Then

$$T \models X$$

$$\text{iff}$$

$$T \cup \{\{\bar{a}\} \mid a \in X\} \vdash \{\}.$$

Slide 13

Logic Programming in Algebraic Domains

Logic programs are sets of rules $Y \leftarrow X$, where X, Y are clauses.

$e \in D$ *model* of P iff for all $Y \leftarrow X$ in P: if $e \models X$ then $e \models Y$.

Clause Y *logical consequence* of P ($Y \in \mathrm{cons}(P)$)
if every model of P satisfies Y.

Differs from theories: $\mathrm{cons}(T)$ is set of all logical consequences of T.

Slide 14

Logic Programming in Algebraic Domains

Propagation rule $\mathrm{CP}(P)$:

$$\frac{X_1 \quad \dots \quad X_n; \quad a_i \in X_i; \quad Y \leftarrow Z \in P; \quad \mathrm{mub}\{a_1, \dots, a_n\} \models Z}{Y \cup \bigcup_{i=1}^{n} (X_i \setminus \{a_i\})}$$

Semantic operator on theories:

$$\mathcal{T}_P(T) = \mathrm{cons}\left(\{Y \mid Y \text{ is a } \mathrm{CP}(P)\text{-consequence of } T\}\right).$$

- ▶ $\mathcal{T}_P$ is Scott continuous [RZ01].
- ▶ $\mathrm{fix}\{\mathcal{T}_P\} = \mathrm{cons}(P)$.

Slide 15

Default Negation

$$\texttt{flies}(x) \leftarrow \texttt{bird}(x) \wedge \neg\texttt{penguin}(x)$$

$$\texttt{bird}(\mathrm{Bob}) \leftarrow$$

Does Bob fly?

Slide 16

Encoding

Use $\mathbb{T}^\omega$:

$$\{f\} \leftarrow \{b\bar{p}\}$$
$$\{b\} \leftarrow \{\bot\}$$

$\{b\}$ is logical consequence, but not $\{f\}$.

Alternative:

$$\{f, p\} \leftarrow \{b\}$$
$$\{b\} \leftarrow \{\bot\}$$

$\{b\}$ and $\{f, p\}$ are logical consequences, but not $\{f\}$.

Extended Programs

Slide 17

Extended precondition: (c, N), c compact, N clause.
Extended clause: Finite set of extended preconditions.
Extended rule: $Y \leftarrow X$, Y clause, X extended clause.
Extended program: Set of extended rules.

Example revisited:

$$\{f\} \leftarrow \{(b, \{p\})\}$$
$$\{b\} \leftarrow \{\bot, \{\}\}$$

$T \models (c, N)$ iff $T \models \{c\}$ and $T \not\models \{d\}$ for all $d \in N$.

Can obtain *stable model semantics* for this paradigm.
Direct generalization from NMR.

Stable Models

Slide 18

T theory. P/T is the program obtained as follows.

Delete all ext. precond. (c, N) with $T \models \{d\}$ for some $d \in N$.

From all remainaing (c, N), delete all d from N with $T \not\models \{d\}$.

Default operator $\mathcal{D}_P(T) = \operatorname{fix}\left(\mathcal{T}_{P/T}\right)$.

The *stable models* of P are fixed-points of $\mathcal{D}_P$.

Bob example: $\{f\}$ is in unique stabel model.

Default Negation

Slide 19

$\mathcal{D}_P$ antitonic (order-reversing).

$\mathcal{D}_P^2$ monotonic.

Alternating fixed-point theory; well-founded models.

Analogous developments as in answer set programming.

So?

Slide 20

In order to get the *Resolution Theorem*, we had to enforce strong conditions on the domain.

In order to get *default negation*, we had to enhance the [RZ01] logic programming paradigm.

In the early days of logic programming and nonmonotonic reasoning, confusion was caused due to a “identification” of classical and default negation.

The [RZ01] framework *forces* us to seperate the issues.

Quo Vadis?

Nice: conceptually clean distinction between background knowledge and program.

Slide 21

To what extent can we enhance the paradigm?
Does it provide an expressive and useful setting for knowledge representation and reasoning?

Reasoning on concept lattices?