

The Complexity of Reasoning for Fragments of Default Logic

Arne Meier, O. Beyersdorff, M. Thomas, H. Vollmer

30. June, 2009

Overview

- 1 Default Logic
 - Syntax and Semantics
- 2 Post's Lattice
 - Universal Algebra
 - Post's Lattice and Computational Complexity
- 3 The Complexity of Default Logic
 - Extension Existence
 - Credulous Reasoning
 - Skeptical Reasoning
- 4 Summary

What is Default Logic?

What is Default Logic?

- a non-monotone logic, introduced 1980 by Reiter
- models common-sense reasoning
- extends classical logic with *default rules*
- undecidable for first order logic (Reiter)
- **here:** propositional logic

Default Rules and Theories

Definition (Reiter 80)

A **default rule** is a triple $\frac{\alpha : \beta}{\gamma}$, where

α is called the **prerequisite**,

β is called the **justification**, and

γ is called the **consequent**,

for α, β, γ propositional formulae.

Informally: infer a formula γ from a set of formulae W by a default rule

$\frac{\alpha : \beta}{\gamma}$, if $\alpha \in W$ and $\neg\beta \notin W$.

Default Theories

Definition (Reiter 80)

A *default theory* is a tuple $\langle W, D \rangle$, where W is a set of formulae and D is a set of default rules.

Example: Playing Football with Default Rules

$$W = \{ \text{football}, \text{rain}, \text{cold} \wedge \text{rain} \rightarrow \text{snow} \}$$

$$D = \left\{ \frac{\text{football} : \neg \text{snow}}{\text{takesPlace}} \right\}$$

$\neg \text{snow}$ is consistent with W . Hence we can infer *takesPlace*.

Default Theories

Definition (Reiter 80)

A **default theory** is a tuple $\langle W, D \rangle$, where W is a set of formulae and D is a set of default rules.

Example: Playing Football with Default Rules

$$W = \{ \text{football}, \text{rain}, \text{cold} \wedge \text{rain} \rightarrow \text{snow}, \text{cold} \}$$

$$D = \left\{ \frac{\text{football} : \neg \text{snow}}{\text{takesPlace}} \right\}$$

snow is consistent with W . Hence we **cannot** infer *takesPlace*.

Default logics are **non-monotone**!

Stable Extensions

Semantics: A Stage Construction (Reiter 80)

Given: a default theory $\langle W, D \rangle$ and set of formulae E :

$$E_0 := W$$

$$E_{i+1} := \text{Th}(E_i) \cup \left\{ \gamma \mid \frac{\alpha : \beta}{\gamma} \in D, \alpha \in E_i \text{ and } \neg\beta \notin E_i \right\}.$$

Then: E is a stable extension of $\langle W, D \rangle$ iff $E = \bigcup_{i \in \mathbb{N}} E_i$.

Stable Extensions

Semantics: A Stage Construction (Reiter 80)

Given: a default theory $\langle W, D \rangle$ and set of formulae E :

$$E_0 := W$$

$$E_{i+1} := \text{Th}(E_i) \cup \left\{ \gamma \mid \frac{\alpha : \beta}{\gamma} \in D, \alpha \in E_i \text{ and } \neg\beta \notin E_i \right\}.$$

Then: E is a stable extension of $\langle W, D \rangle$ iff $E = \bigcup_{i \in \mathbb{N}} E_i$.

Stable Extensions

Semantics: A Stage Construction (Reiter 80)

Given: a default theory $\langle W, D \rangle$ and set of formulae E :

$$E_0 := W$$

$$E_{i+1} := \text{Th}(E_i) \cup \{ \gamma \mid \frac{\alpha : \beta}{\gamma} \in D, \alpha \in E_i \text{ and } \neg\beta \notin E_i \}.$$

Then: E is a stable extension of $\langle W, D \rangle$ iff $E = \bigcup_{i \in \mathbb{N}} E_i$.

Stable extensions correspond to possible views of an agent
on the basis of $\langle W, D \rangle$.

A Second Approach: Generating Defaults

Semantics: Generating Defaults (Reiter 80)

Given: a default theory $\langle W, D \rangle$ and set of formulae E :

Define the set of **generating defaults** as

$$G := \left\{ \frac{\alpha : \beta}{\gamma} \in D \mid \alpha \in E \text{ and } \neg\beta \notin E \right\}.$$

If: E is a stable extension of $\langle W, D \rangle$ then

$$E = \text{Th} \left(W \cup \left\{ \gamma \mid \frac{\alpha : \beta}{\gamma} \in G \right\} \right).$$

Example: There can be more stable extensions!

Recap

$$G := \left\{ \frac{\alpha:\beta}{\gamma} \in D \mid \alpha \in E \text{ and } \neg\beta \notin E \right\} \text{ (generating defaults),}$$

$$E = \text{Th} \left(W \cup \left\{ \gamma \mid \frac{\alpha:\beta}{\gamma} \in G \right\} \right) \text{ (stable extension } E \text{).}$$

Example 1

$$W = \{B \rightarrow \neg A \wedge \neg C\}, D = \left\{ \frac{\top:A}{A}, \frac{\top:B}{B}, \frac{\top:C}{C} \right\}$$

$$E_1 = \text{Th}(W \cup \{A, C\}), E_2 = \text{Th}(W \cup \{B\})$$

Example: There can be no stable extension!

Recap

$$G := \left\{ \frac{\alpha:\beta}{\gamma} \in D \mid \alpha \in E \text{ and } \neg\beta \notin E \right\} \text{ (generating defaults),}$$
$$E = \text{Th} \left(W \cup \left\{ \gamma \mid \frac{\alpha:\beta}{\gamma} \in G \right\} \right) \text{ (stable extension } E \text{).}$$

Example 2

$$W = \emptyset, D = \left\{ \frac{\top:A}{\neg A} \right\}$$

$\langle W, D \rangle$ has no stable extension.

Three Important Decision Problems

EXT, CRED, SKEP

Extension Existence Problem

Instance: a default theory $\langle W, D \rangle$

Question: Does $\langle W, D \rangle$ have a stable extension?

Three Important Decision Problems

EXT, CRED, SKEP

Extension Existence Problem

Instance: a default theory $\langle W, D \rangle$

Question: Does $\langle W, D \rangle$ have a stable extension?

Credulous Reasoning Problem

Instance: a formula φ and a default theory $\langle W, D \rangle$

Question: Is there a stable extension of $\langle W, D \rangle$ that includes φ ?

Three Important Decision Problems

EXT, CRED, SKEP

Extension Existence Problem

Instance: a default theory $\langle W, D \rangle$

Question: Does $\langle W, D \rangle$ have a stable extension?

Credulous Reasoning Problem

Instance: a formula φ and a default theory $\langle W, D \rangle$

Question: Is there a stable extension of $\langle W, D \rangle$ that includes φ ?

Skeptical Reasoning Problem

Instance: a formula φ and a default theory $\langle W, D \rangle$

Question: Does **every** stable extension of $\langle W, D \rangle$ include φ ?

Known Complexity Results

Theorem (Gottlob 92)

- 1 *The Extension Existence Problem is Σ_2^P -complete.*
- 2 *The Credulous Reasoning Problem is Σ_2^P -complete.*
- 3 *The Skeptical Reasoning Problem is Π_2^P -complete.*

Motivation

What about the complexity if ...

- ... we allow only the Boolean function \wedge for formulae in W ?
- ... we allow only monotone functions in the default rules?

Motivation

What about the complexity if ...

- ... we allow only the Boolean function \wedge for formulae in W ?
- ... we allow only monotone functions in the default rules?

↪ Parameterization of the three decision problems by a set B of Boolean functions

Motivation

What about the complexity if ...

- ... we allow only the Boolean function \wedge for formulae in W ?
- ... we allow only monotone functions in the default rules?

↪ Parameterization of the three decision problems by a set B of Boolean functions

↪ We need a suitable characterisation for sets of Boolean functions.

A Little Bit of Universal Algebra

Definition

- A **clone** is a set B of Boolean functions that contains all projections and is closed under composition.
- For a set B of Boolean functions, we denote by $[B]$ the smallest clone containing B .
- B is called a **base** for $[B]$.

A Little Bit of Universal Algebra

Definition

- A **clone** is a set B of Boolean functions that contains all projections and is closed under composition.
- For a set B of Boolean functions, we denote by $[B]$ the smallest clone containing B .
- B is called a **base** for $[B]$.

Thus:

- $[B]$ consists of those functions that can be computed by a Boolean circuit with basis B (gates from B).
- $[B]$ consists of those functions that can be defined by a propositional formula with connectives from B .

So what?

Π – computational problem defined over **Boolean circuits**

$\Pi(B)$ – the restriction of Π to circuits with gates from B ,
e.g., **CVP**(B), the circuit value problem for circuits with
gates from B

So what?

Π – computational problem defined over **Boolean circuits**

$\Pi(B)$ – the restriction of Π to circuits with gates from B ,
e.g., $CVP(B)$, the circuit value problem for circuits with
gates from B

Then: If $B \subseteq [B']$ then $\Pi(B) \leq_m^P \Pi(B')$ ($CVP(B) \leq_m^P CVP(B')$).

So what?

Π – computational problem defined over **Boolean circuits**

$\Pi(B)$ – the restriction of Π to circuits with gates from B ,
e.g., $CVP(B)$, the circuit value problem for circuits with
gates from B

Then: If $B \subseteq [B']$ then $\Pi(B) \leq_m^P \Pi(B')$ ($CVP(B) \leq_m^P CVP(B')$).

Upper bounds carry downwards.

Lower bounds carry upwards.

So what?

Π – computational problem defined over Boolean circuits

$\Pi(B)$ – the restriction of Π to circuits with gates from B ,
e.g., $CVP(B)$, the circuit value problem for circuits with
gates from B

Then: If $B \subseteq [B']$ then $\Pi(B) \leq_m^P \Pi(B')$ ($CVP(B) \leq_m^P CVP(B')$).

Upper bounds carry downwards.

Lower bounds carry upwards.

Complexity of $\Pi(B)$ is determined by the clone $[B]$.

So what?

Π – computational problem defined over **propositional formulas**

$\Pi(B)$ – the restriction of Π to circuits with connectives from B ,
e.g., $SAT(B)$, the satisfiability problem for formulas with
connectives from B

Then: If $B \subseteq [B']$ then $\Pi(B) \leq_m^P \Pi(B') \sim (SAT(B) \leq_m^P SAT(B'))$.

Upper bounds carry downwards.

Lower bounds carry upwards.

Complexity of $\Pi(B)$ is determined by the clone $[B]$.

So what?

Π – computational problem defined over propositional formulas

$\Pi(B)$ – the restriction of Π to circuits with connectives from B ,
e.g., $SAT(B)$, the satisfiability problem for formulas with
connectives from B

Then: If $B \subseteq [B']$ then $\Pi(B) \leq_m^P \Pi(B') \sim (SAT(B) \leq_m^P SAT(B'))$.

Upper bounds carry downwards.

Lower bounds carry upwards.

Complexity of $\Pi(B)$ is determined by the clone $[B]$.

Caveat: Explosion of formula size!
(Usually does not happen ...)

Properties of Boolean Functions

Some Properties of Boolean Functions

- f is **c -reproducing** if $f(c, \dots, c) = c$, $c \in \{0, 1\}$.
- f is **monotone** if $a_1 \leq b_1, \dots, a_n \leq b_n$ implies $f(a_1, \dots, a_n) \leq f(b_1, \dots, b_n)$.
- f is **c -separating** if there exists an $i \in \{1, \dots, n\}$ such that $f(a_1, \dots, a_n) = c$ implies $a_i = c$, $c \in \{0, 1\}$.
- f is **self-dual** if $f \equiv \text{dual}(f)$, where $\text{dual}(f)(x_1, \dots, x_n) = \neg f(\neg x_1, \dots, \neg x_n)$.
- f is **linear** if $f \equiv x_1 \oplus \dots \oplus x_n \oplus c$ for a constant $c \in \{0, 1\}$ and variables x_1, \dots, x_n .

Important Boolean Clones

Name	Definition	Base
BF	All Boolean functions	$\{\wedge, \neg\}$
R_0	$\{f : f \text{ is 0-reproducing}\}$	$\{\wedge, \neq\}$
R_1	$\{f : f \text{ is 1-reproducing}\}$	$\{\vee, \rightarrow\}$
M	$\{f : f \text{ is monotone}\}$	$\{\vee, \wedge, 0, 1\}$
S_0	$\{f : f \text{ is 0-separating}\}$	$\{\rightarrow\}$
S_1	$\{f : f \text{ is 1-separating}\}$	$\{\neq\}$
D	$\{f : f \text{ is self-dual}\}$	$\{(x \wedge \bar{y}) \vee (x \wedge \bar{z}) \vee (\bar{y} \wedge \bar{z})\}$
L	$\{f : f \text{ is linear}\}$	$\{\oplus, 1\}$
V	$\{f : f \equiv c_0 \vee \bigvee_{i=1}^n c_i x_i\}$	$\{\vee, 0, 1\}$
E	$\{f : f \equiv c_0 \wedge \bigwedge_{i=1}^n c_i x_i\}$	$\{\wedge, 0, 1\}$
N	$\{f : f \text{ depends on only one variable}\}$	$\{\neg, 0, 1\}$
I	$\{f : f \text{ is a projection or constant}\}$	$\{\text{id}, 0, 1\}$

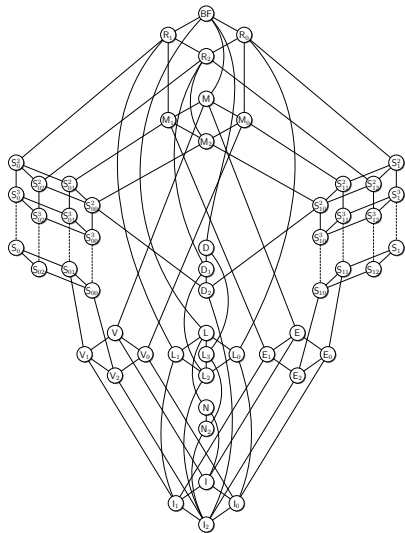
Important Boolean Clones

Name	Definition	Base
BF	All Boolean functions	$\{\wedge, \neg\}$
R_0	$\{f : f \text{ is 0-reproducing}\}$	$\{\wedge, \neq\}$
R_1	$\{f : f \text{ is 1-reproducing}\}$	$\{\vee, \rightarrow\}$
M	$\{f : f \text{ is monotone}\}$	$\{\vee, \wedge, 0, 1\}$
S_0	$\{f : f \text{ is 0-separating}\}$	$\{\rightarrow\}$
S_1	$\{f : f \text{ is 1-separating}\}$	$\{\neq\}$
D	$\{f : f \text{ is self-dual}\}$	$\{(x \wedge \bar{y}) \vee (x \wedge \bar{z}) \vee (\bar{y} \wedge \bar{z})\}$
L	$\{f : f \text{ is linear}\}$	$\{\oplus, 1\}$
V	$\{f : f \equiv c_0 \vee \bigvee_{i=1}^n c_i x_i\}$	$\{\vee, 0, 1\}$
E	$\{f : f \equiv c_0 \wedge \bigwedge_{i=1}^n c_i x_i\}$	$\{\wedge, 0, 1\}$
N	$\{f : f \text{ depends on only one variable}\}$	$\{\neg, 0, 1\}$
I	$\{f : f \text{ is a projection or constant}\}$	$\{\text{id}, 0, 1\}$

Important Boolean Clones

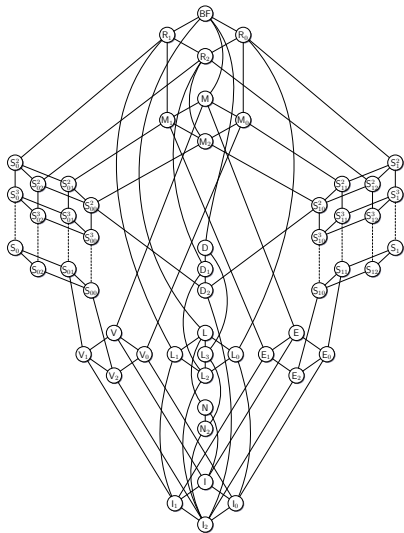
Name	Definition	Base
BF	All Boolean functions	$\{\wedge, \neg\}$
R_0	$\{f : f \text{ is 0-reproducing}\}$	$\{\wedge, \neq\}$
R_1	$\{f : f \text{ is 1-reproducing}\}$	$\{\vee, \rightarrow\}$
M	$\{f : f \text{ is monotone}\}$	$\{\vee, \wedge, 0, 1\}$
S_0	$\{f : f \text{ is 0-separating}\}$	$\{\rightarrow\}$
S_1	$\{f : f \text{ is 1-separating}\}$	$\{\neq\}$
D	$\{f : f \text{ is self-dual}\}$	$\{(x \wedge \bar{y}) \vee (x \wedge \bar{z}) \vee (\bar{y} \wedge \bar{z})\}$
L	$\{f : f \text{ is linear}\}$	$\{\oplus, 1\}$
V	$\{f : f \equiv c_0 \vee \bigvee_{i=1}^n c_i x_i\}$	$\{\vee, 0, 1\}$
E	$\{f : f \equiv c_0 \wedge \bigwedge_{i=1}^n c_i x_i\}$	$\{\wedge, 0, 1\}$
N	$\{f : f \text{ depends on only one variable}\}$	$\{\neg, 0, 1\}$
I	$\{f : f \text{ is a projection or constant}\}$	$\{\text{id}, 0, 1\}$

Post's Lattice



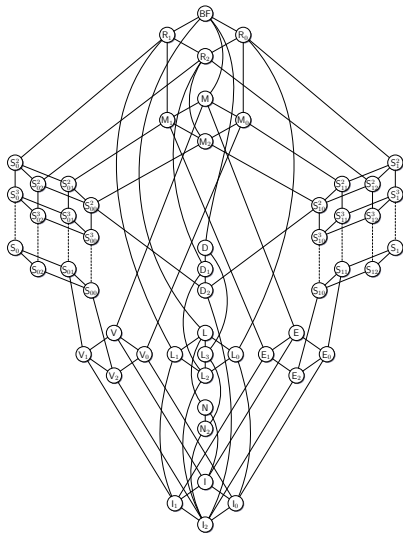
- Refers to Emil Post, 1941.

Post's Lattice



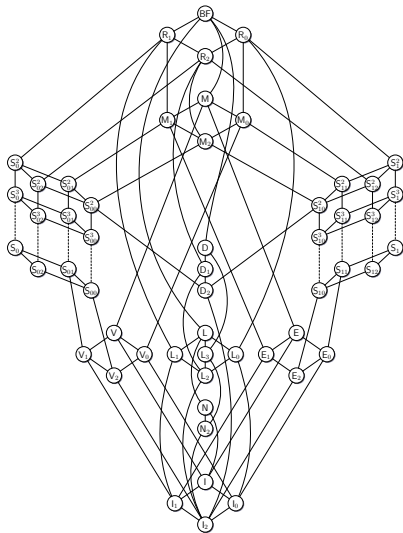
- Refers to Emil Post, 1941.
- Each node is a finite set of Boolean functions (*basis*).

Post's Lattice



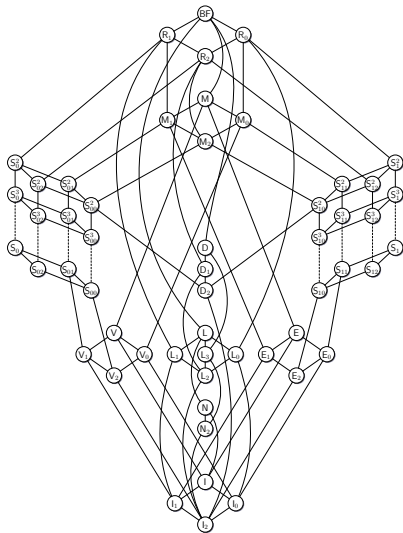
- Refers to Emil Post, 1941.
- Each node is a finite set of Boolean functions (*basis*).
- Many new decision problems in default logic arise (for each clone).

Post's Lattice



- Refers to Emil Post, 1941.
- Each node is a finite set of Boolean functions (*basis*).
- Many new decision problems in default logic arise (for each clone).
- Hardness results carry upwards.

Post's Lattice



- Refers to Emil Post, 1941.
- Each node is a finite set of Boolean functions (*basis*).
- Many new decision problems in default logic arise (for each clone).
- Hardness results carry upwards.
- Membership results carry downwards.

Post's Lattice and Computational Complexity

Motivation for a fine classification using Post's Lattice

Given a difficult problem involving Boolean functions, we aim to

- identify the sources of hardness of the general problem, and
- design more efficient algorithms for tractable fragments.

Post's Lattice and Computational Complexity

Motivation for a fine classification using Post's Lattice

Given a difficult problem involving Boolean functions, we aim to

- identify the sources of hardness of the general problem, and
- design more efficient algorithms for tractable fragments.

Important problems have been classified via this approach:

- Lewis 79: classification of SAT
SAT(B) is NP-complete iff $\nrightarrow \in [B]$.
- Reith, Wagner 05: circuit value problem, QBF
- nonclassical logics: LTL, CTL, ...

Post's Lattice and Default Logic

The "new" decision problems

- We restrict the allowed Boolean functions for formulae in W and D (also for given φ) to functions from $[B]$.
- $\text{EXT}(B)$ is the extension existence problem for B -default theories.
- $\text{CRED}(B)$ and $\text{SKEP}(B)$ are defined analogously (also restricted φ).

Extension Existence Problem

EXT(B):

Instance: a B -default theory $\langle W, D \rangle$

Question: Does $\langle W, D \rangle$ have a stable extension?

Extension Existence Problem

Theorem

Let B be a finite set of Boolean functions. Then $\text{EXT}(B)$ is

- 1 Σ_2^P -complete if $S_1 \subseteq [B] \subseteq \text{BF}$ or $D \subseteq [B] \subseteq \text{BF}$,
- 2 NP-complete if $[B] \in \{N, N_2, L, L_0, L_3\}$,
- 3 trivial in all other cases (i. e., if $[B] \subseteq R_1$ or $[B] \subseteq M$).

S_1 : base $\{\neg\}$

N : base $\{\neg, 0, 1\}$

M : monotone functions

D : $f \equiv \neg f(\neg x_1, \dots, \neg x_n)$

L : base $\{\oplus, 1\}$

R_1 : $f(1, \dots, 1) = 1$

Credulous Reasoning, Skeptical Reasoning

Credulous Reasoning Problem, CRED(B)

Instance: a B -formula φ and a B -default theory $\langle W, D \rangle$

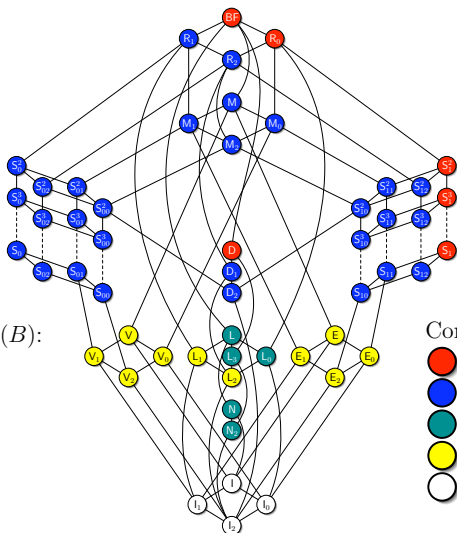
Question: Is there a stable extension of $\langle W, D \rangle$ that includes φ ?

Skeptical Reasoning Problem, SKEP(B)

Instance: a B -formula φ and a B -default theory $\langle W, D \rangle$

Question: Does every stable extension of $\langle W, D \rangle$ include φ ?

Complexity Overview: Credulous Reasoning

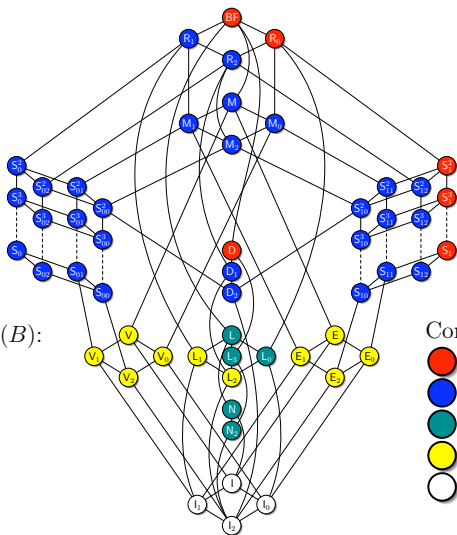
Complexity of $\text{CRED}(B)$:

- Σ_2^P -complete
- coNP-complete
- NP-complete
- P-complete
- NL-complete

Complexity of $\text{SKEP}(B)$:

- Π_2^P -complete
- coNP-complete
- coNP-complete
- P-complete
- NL-complete

Complexity Overview: Credulous/Skeptical Reasoning



Complexity of CRED(B):

- Σ_2^P -complete
- coNP-complete
- NP-complete
- P-complete
- NL-complete

Complexity of SKEP(B):

- Π_2^P -complete
- coNP-complete
- coNP-complete
- P-complete
- NL-complete

Summary

Complexity for ...

- $\text{EXT}(B)$ is a trichotomy (Σ_2^P -, NP-complete, and trivial cases)
- $\text{CRED}(B)$ is a pentachotomy (Σ_2^P -, coNP-, NP-, P-, and NL-complete cases)
- $\text{SKEP}(B)$ is a tetrachotomy (Π_2^P -, coNP-, P-, and NL-complete cases)

The complexity is determined by two parameters:

- whether there exist unique extensions, and
- how hard it is to test for formula implication.