

The Many Faces of Modal Logic

Day 4: Algorithms

Dirk Pattinson¹ and Lutz Schröder²

¹Australian National University, Canberra

²Friedrich-Alexander-Universität Erlangen-Nürnberg

NASSLLI 2014, College Park, MD

Automated Reasoning, Or: Decidability and Complexity

Yesterday. Coalgebraic logics have the small model property.

However. Decidability is **not** automatic by FMP

Reason. If T maps finite sets to infinite sets, the set of transition structures

$$\{\gamma: C \rightarrow TC \mid \gamma \text{ a function}\}$$

may well be infinite even for finite C .

Example. Game frames, probabilistic frames, multigraph frames

Plus: Recall we know **nothing** about T and it may well encode the halting problem.

Wishful Thinking

Two Options for $\chi = \bigwedge_i \heartsuit_i \phi_i \rightarrow \bigvee_j \heartsuit_j \psi_k$:

Semantically

$\neg \chi$ satisfiable \implies

$\forall \phi / \psi \in \mathcal{R}, \sigma : V \rightarrow \mathcal{F}(\Lambda).$

$\psi \sigma \vdash_{\text{PL}} \chi \implies \neg \phi \sigma$ satisfiable

Countermodels of the $\phi \sigma$

Countermodel of χ

Syntactically.

χ provable \implies

$\exists \phi / \psi \in \mathcal{R}, \sigma : V \rightarrow \mathcal{F}(\Lambda)$

$\phi \sigma$ provable and $\psi \sigma \vdash_{\text{PL}} \chi$

Proof of some $\phi \sigma$

Proof of χ

Problems.

1. checking single rules is insufficient, and
2. there may be infinitely many rules to check!

Example

Consequences of **multiple rules**

$$\frac{\frac{a \wedge (b \wedge c) \rightarrow a \wedge b \wedge c}{\Box a \wedge \Box (b \wedge c) \rightarrow \Box (a \wedge b \wedge c)} \quad \frac{b \wedge c \rightarrow b \wedge c}{\Box b \wedge \Box c \rightarrow \Box (b \wedge c)}}{\Box a \wedge \Box b \wedge \Box c \rightarrow \Box (a \wedge b \wedge c)}$$

Infinitely many **possible premises**

$$\frac{\vdash a \rightarrow b?}{\Box a \rightarrow \Box b \vdash \Box a \rightarrow \Box b} \quad \frac{\vdash a \wedge a \rightarrow b?}{\Box a \wedge \Box a \rightarrow \Box b \vdash_{\text{PL}} \Box a \rightarrow \Box b}$$
$$\frac{\vdash a \wedge a \wedge a \rightarrow b?}{\Box a \wedge \Box a \wedge \Box a \rightarrow \Box b \vdash_{\text{PL}} \Box a \rightarrow \Box b} \quad \text{etc.}$$

→ Need admissibility of **cut** and **contraction** in a Sequent Calculus.

Syntactic and Semantic Approach

Problem 1. Eliminate the need for checking multiple rules.

Semantically

$$\chi = \bigwedge \heartsuit_i \phi_i \rightarrow \bigvee_j \phi_j \text{ valid}$$

\implies

$\exists \phi / \psi \in \mathcal{R}$ and subst'n σ :

- ▶ $\text{PL} \vdash \psi \sigma \rightarrow \chi$
- ▶ $\phi \sigma$ valid

Syntactically.

$$\chi = \bigwedge \heartsuit_i \phi_i \rightarrow \bigvee_j \phi_j \text{ provable}$$

\implies

$\exists \phi / \psi \in \mathcal{R}$, subst'n σ

- ▶ $\text{PL} \vdash \psi \sigma \rightarrow \chi$
- ▶ $\phi \sigma$ provable

Semantic Approach. Stronger coherence condition: valid clause consequence of **single** rule conclusion

Syntactically. Propositional combinations of rule conclusions are conclusions of **single** rule.

Formal Framework: Sequent Calculus

Sequents are **multisets** of formulas. Write $\Gamma, \Delta := \Gamma \cup \Delta$; $\Gamma, A := \Gamma, \{A\}$

Propositional Rules

$$\frac{\Gamma, A}{\Gamma, \neg\neg A} \quad \frac{\Gamma, A \quad \Gamma, B}{\Gamma, A \wedge B} \quad \frac{\Gamma, \neg A, \neg B}{\Gamma, \neg(A \wedge B)}$$

Modal Rules from a one-step rule ϕ/ψ

$$\frac{\text{Lit}(\phi_1)\sigma \quad \dots \quad \text{Lit}(\phi_n)\sigma}{\text{Lit}(\psi)\sigma, \Delta}$$

where σ subst'n, $\text{cnf}(\phi) = \phi_1 \wedge \dots \wedge \phi_n$, and $\text{Lit}(\cdot)$ is the **set** of literals occurring in a clause.

Notation. $G, \mathcal{R} \vdash \Gamma$ if Γ can be derived using the propositional rules and the “imported” modal rules.

Rule Sets in Sequent Form (without side conditions)

Modal Logic E .

$$\frac{\neg p, q \quad p, \neg q}{\neg \Box p, \Box q}$$

Modal Logic K .

$$\frac{\neg p_1, \dots, \neg p_n, p_0}{\neg \Box p_1, \dots, \neg \Box p_n, p_0}$$

Graded Modal Logic.

$$\frac{\sum_{i=1}^n p_i \leq \sum_{j=1}^m q_j}{\neg \Diamond_{k_1} p_1, \dots, \neg \Diamond_{k_n} p_n, \Diamond_h q_1, \dots, \Diamond_{l_j} q_j}$$

Probabilistic Modal Logic.

$$\frac{\sum_{i=1}^n p_i + u \leq \sum_{j=1}^m q_j}{\neg L_{u_1} p_1, \dots, \neg L_{u_n} p_n, L_{v_1} q_1, \dots, L_{v_m} q_m}$$

Conditional Logic.

$$\frac{\neg p_0, p_1 \quad \neg p_1, p_0 \quad \dots \quad \neg p_0, p_n \quad \neg p_n, p_0 \quad \neg q_1, \dots, \neg q_n, q_0}{\neg (p_1 \Rightarrow q_1), \dots, \neg (p_n \Rightarrow q_n), (p_0 \Rightarrow q_0)}$$

Coalition Logic.

$$\frac{\neg p_1, \dots, \neg p_n}{\neg [C_1] p_1, \dots, \neg [C_n] p_n}$$

$$\frac{\neg p_1, \dots, \neg p_n, q, r_1, \dots, r_m}{\neg [C_1] p_1, \dots, \neg [C_n] p_n, [D] q, [N] r_1, \dots, [N] r_m}$$

Sequent Calculi: Simple Structural Properties

Structural Rules

$$(W) \frac{\Gamma}{\Gamma, \Delta} \quad (\text{Con}) \frac{\Gamma, A, A}{\Gamma, A} \quad (\text{Cut}) \frac{\Gamma, A \quad \Delta, \neg A}{\Gamma, \Delta}$$

Notation. $\mathcal{G}\mathcal{R}\{W, \text{Con}, \text{Cut}\} \vdash \Gamma$ if Γ can be derived using indicated structural rules.

Weakening Lemma.

- ▶ $\mathcal{G}\mathcal{R}W \vdash \Gamma$ implies $\mathcal{G}\mathcal{R} \vdash \Gamma$ (W is **admissible**)

Inversion Lemma.

- ▶ $\mathcal{G}\mathcal{R} \vdash \Gamma, A \wedge B$ implies $\mathcal{G}\mathcal{R} \vdash \Gamma, A$ and $\mathcal{G}\mathcal{R} \vdash \Gamma, B$
 - ▶ $\mathcal{G}\mathcal{R} \vdash \Gamma, \neg(A \wedge B)$ implies $\mathcal{G}\mathcal{R} \vdash \Gamma, \neg A, \neg B$
- (I.e. **inversion rules** are admissible)

Why Sequent Calculi?

Backwards Proof Search in Hilbert Calculi, e.g. modus ponens:

$$\frac{A \rightarrow B \quad A}{B}$$

- ▶ infinitely many possibilities for A , failure of subformula property in Sequent Calculi, e.g. $(\neg\wedge)$

$$\frac{\Gamma, \neg A, \neg B}{\Gamma, \neg(A \wedge B)}$$

- ▶ premiss mentions only subformulae of conclusion, finitely many possible applications

The Enemies.

$$(\text{Cut}) \frac{\Gamma, A \quad \Delta, \neg A}{\Gamma, \Delta} \quad (\text{Con}) \frac{\Gamma, A, A}{\Gamma, A}$$

Goal: Cut and Contraction-Free Calculi

Suppose we've already succeeded.

$$\mathsf{GR} \vdash \Gamma \iff T \models \bigvee \Gamma$$

Decidability.

- ▶ proof trees are of finite depth and finitely branching
- ▶ decidability as long as rules are effective

Complexity.

- ▶ proof trees are polynomially deep
- ▶ PSPACE if rules are 'in NP'

The Easy Part

Soundness. $G\mathcal{R} \vdash \Gamma$ implies $T \models \Gamma$

Completeness with Cut. If \mathcal{R} is one-step complete, then $G\mathcal{R}\text{ConCut} \vdash \Gamma$ whenever $T \models \bigvee \Gamma$.

Proof. Use completeness of Hilbert system; cut rule simulates modus ponens.

The Hard Part. Get rid of cut and contraction.

Semantic Cut Elimination

Stronger **Coherence Condition**: \mathcal{R} is **one-step cutfree complete** if

$$\text{If } X, \sigma \models \chi \implies \exists \phi / \psi \in \mathcal{R}, \tau : V \rightarrow V \text{ s.t. } X, \sigma \models \phi \tau \text{ and } \psi \tau \vdash_{\text{PL}} \chi$$

(χ clause over $\Lambda(V)$, $\sigma : V \rightarrow \mathcal{P}(X)$).

Intuition. Valid modalized formulas are derivable using a **single** rule.

Cutfree complete rule sets exist. The set of all one-step sound one-step rules is one-step cutfree complete (as seen earlier).

Cut-Free Completeness

Examples. All rule sets we have seen today are one-step cutfree complete.

Thm. Suppose \mathcal{R} is one-step cutfree complete. Then

$$T \models \bigvee \Gamma \quad \text{iff} \quad \text{G}\mathcal{R}\text{Con} \vdash \Gamma.$$

Proof Sketch. By contraposition: If $\text{G}\mathcal{R}\text{Con} \not\vdash \Gamma$

- ▶ all rules applicable to Γ have a premiss that is not derivable
- ▶ countermodel construction with $C =$ modal nodes (no top-level propositional connectives)
- ▶ construct $\gamma : C \rightarrow TC$ using cutfree completeness
- ▶ by induction: $T \not\models \bigvee \Gamma$

Syntactic Cut Elimination

Defn. A rule set \mathcal{R} **absorbs cut** if for all $\chi = \bigwedge \heartsuit_i p_i \rightarrow \bigvee \heartsuit_j p_j$

- ▶ if $\psi_1 \sigma_1, \dots, \psi_n \sigma_n \vdash_{\text{PL}} \chi$ for rules ϕ_i / ψ_i , renamings σ_i
- ▶ then there is $\phi / \psi \in \mathcal{R}$, renaming σ such that
- ▶ $\phi_1 \sigma_1, \dots, \phi_n \sigma_n \vdash_{\text{PL}} \phi \sigma$ and $\psi \sigma \vdash_{\text{PL}} \chi$

(combinations of rule conclusions are subsumed by by single rules)

Thm. Let \mathcal{R} be one-step complete and absorb cut. Then

$$T \models \bigvee \Gamma \text{ iff } \text{G}\mathcal{R}\text{Con} \vdash \Gamma.$$

Proof Sketch. If $T \models \bigvee \Gamma$ then $\mathcal{R} \vdash \bigvee \Gamma$. Consequently:

- ▶ for every component χ of the cnf of $\bigvee \Gamma$
- ▶ we can find $\phi / \psi \in \mathcal{R}$ and $\sigma : V \rightarrow \mathcal{F}(\Lambda)$ s.t. $\psi \sigma \vdash_{\text{PL}} \chi$ and $\mathcal{R} \vdash \phi \sigma$

All these steps can be simulated in $\text{G}\mathcal{R}\text{Con}$.

Cut Elimination Proof

Recall. $\mathcal{R} \vdash \forall \Gamma \iff \text{G}\mathcal{R}\text{CutCon} \vdash \Gamma$.

Alternative Approach. Cut admissibility.

Thm. If \mathcal{R} absorbs cut then $\text{G}\mathcal{R}\text{CutCon} \vdash \Gamma$ iff $\text{G}\mathcal{R}\text{Con} \vdash \Gamma$.

Proof Sketch. By induction on proofs with cut, show that cut can be eliminated.

- ▶ cuts between propositional rules: propagate upwards in proof tree
- ▶ cuts between modal rules: by cut-free completeness, cut on $\heartsuit A$ can be replaced by cuts on A
- ▶ cuts between modal / propositional rules: don't occur on $\heartsuit A$ and propagate.

Technically: double induction on size of proof and size of cut formula.

Cut-Free Completeness vs Strict Completeness

Recall. The set of all valid one-step rules is

- ▶ one-step cut-free complete (by inspection of proof)
- ▶ absorbs cut (by a simple semantical argument)

Thm. TFAE:

1. \mathcal{R} is one-step complete for T and absorbs cut
2. \mathcal{R} is one-step cutfree complete for T

Our Second Enemy.

$$(\text{Con}) \frac{\Gamma, A, A}{\Gamma, A}$$

- ▶ **multisets** make contraction explicit, and contraction gives arbitrarily large proof trees

Question. Can we circumnavigate the problem by having sequents as **sets** instead? (think about it . . .)

Contraction

\mathcal{R} **absorbs contraction** if for each $\phi/\psi \in \mathcal{R}$ over V and each $\sigma : V \rightarrow W$, there exist $\phi'/\psi' \in \mathcal{R}$ over V' and $\sigma' : V' \rightarrow W$ s.t.

$$\phi\sigma \vdash_{\text{PL}} \phi'\sigma' \qquad \psi'\sigma' \vdash_{\text{PL}} \psi\sigma$$

and

$$\sigma' : \psi' \rightarrow \psi'\sigma' \quad \text{injective.}$$

Example: K

$$\frac{\phi}{\psi} = \frac{\neg a_1, \neg a_2, b}{\neg \Box a_1, \neg \Box a_2, \Box b} \qquad \sigma(a_1) = \sigma(a_2) = a$$

may be replaced by

$$\frac{\phi'}{\psi'} = \frac{\neg a, b}{\neg \Box a, \Box b} \qquad \sigma' = id$$

Example: GML

$$\frac{\sum_{i=1}^n p_i \leq \sum_{j=1}^m q_j}{\neg \diamond_{k_1} p_1, \dots, \neg \diamond_{k_n} p_n, \diamond_{l_1} q_1, \dots, \diamond_{l_j} q_j} \quad (\sum_{i=1}^n (k_i + 1) > \sum_{j=1}^m l_j)$$

fails to absorb contraction \rightarrow extend the rule set:

$$\frac{\sum_{i=1}^n r_i p_i \geq 0}{\text{sgn}(r_1) \diamond_{k_1} p_1, \dots, \text{sgn}(r_n) \diamond_{k_n} p_n} \quad (r_i \in \mathbb{Z} - \{0\}; \sum_{r_i < 0} |r_i| (k_i + 1) > \sum_{r_i > 0} r_i k_i)$$

Suppose we have a rule set absorbing cut and contraction

PSPACE Bounds via non-deterministic proof search:

- ▶ Have polynomial bound on the height of the proof tree

Require additionally \mathcal{R} **PSPACE-tractable**:

- ▶ for every sequent Γ , the (codes of) rules that entail Γ can be guessed and checked in (nondet.) polynomial time
 - ▶ *Main point*: polynomially large codes must suffice
- ▶ for every (code of a) rule, its premises can be (guessed and) checked in (nondet.) polynomial time.

Thm. If \mathcal{R} is PSPACE-tractable and absorbs cut and contraction then \mathcal{R} -provability is in PSPACE.

Proof Sketch.

- ▶ Implement proof search on an alternating Turing machine:
 - ▶ existentially guess a matching proof rule
 - ▶ universally check provability of all premises
- ▶ Proof trees are of polynomial height
- ▶ Runs in $\text{APTIME} = \text{PSPACE}$.

Examples

Modal Logics K , E , coalition logic and conditional logic:

- ▶ earlier rule sets are already contraction closed, and clearly PSPACE-tractable

Graded Modal Logic. Recall rule:

$$\frac{\sum_{i=1}^n r_i p_i \geq 0}{\text{sgn}(r_1) \diamond_{k_1} p_1, \dots, \text{sgn}(r_n) \diamond_{k_n} p_n} \quad (r_i \in \mathbb{Z} - \{0\}; \sum_{r_i < 0} |r_i|(k_i + 1) > \sum_{r_i > 0} r_i k_i)$$

PSPACE-tractability: polysize solutions of systems of linear inequalities
→ polysize r_i suffice

Generic Complexity Bounds

Thm. Provability in the logics E , M , K , graded modal logic, probabilistic modal logic, coalition logic (and many others!) is decidable in polynomial space.

Remarks.

- ▶ all results just **instances** of the same, general theory
- ▶ results by just **inspecting** corresponding rule sets
- ▶ generation of rule-sets semi-automatic

Extensions.

- ▶ accommodate non-iterative rules: more logics (not covered)
- ▶ accommodate other logical primitives: fixpoints, nominals, etc.

Interlude: Parametric Implementation

Example Language: Haskell

Parametric Formulas.

```
data L a
  = F | T | Atom Int
    | Neg (L a) | And (L a) (L a) | M a (L a)
```

Example. The logic K and graded modal logic

```
data K = K deriving (Eq,Show)
data G = G Int
```

Logic. Type-class that supports matching

```
class (Eq a, Show a) => Logic a where
  match :: Clause a -> [[L a]]
```

(double lists as rule premises are generally in cnf)

Matching and Provability

Example. Syntax of K (again)

```
data K = K
```

Matching: representation of resolution closed rule sets

```
instance Logic K where
  match (Clause (pl,nl)) =
    let (nls,pls) = (map neg (stripmany nl), stripmany pl)
    in map disjlst (map (\x -> x:nls) pls)
```

Generic Provability Predicate.

```
provable :: (Logic a) => L a -> Bool
provable phi = all (\c -> any (all provable) ( match c))
               (cnf phi)
```

(lazyness of Haskell guarantees polynomial space)