Formalizing Classical Modal Logic in Constructive Logic

Christian Doczkal   Gert Smolka

Programming Systems Lab, Saarland University

Coq-3 Workshop, Nijmegen, August 26, 2011

Outline

How to *faithfully* represent *classical* modal logic in the constructive meta theory of Coq and prove *decidability* of satisfiability?

- Quick Review: Decidability in Coq
- Representation of classical modal logic in Coq
- Formalization of the decidability proof

Decidability in Coq

- Coq term normalization defines a model of computation
- Any term of type

  forall x:X, { P x } + { ~ P x }

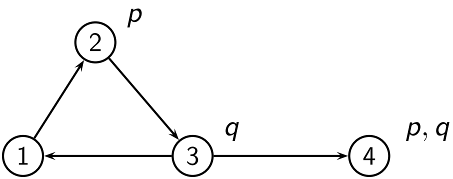
  is a decision procedure for the predicate $P : X \rightarrow \mathrm{Prop}$
- Equivalently one can show

  forall x, P x $\leftrightarrow$ p x = true

  for some $p : X \rightarrow \mathrm{bool}$
- To employ this simple notion of decidability we are confined to an *axiom free* setting

Modal Logic K*

Models: Graphs, Nodes labeled with predicates (p, q, . . .)



Formulas: $s ::= p \mid \neg p \mid s \vee s \mid s \wedge s \mid \Diamond s \mid \Box s \mid \Diamond^* s \mid \Box^* s$

Modal Logic K*

- Formulas are evaluated at a particular state of a model

$$\begin{aligned}
\mathcal{M}, a \models \Diamond s &\quad\approx\quad \text{some successor of } a \text{ satisfies } s\\
\mathcal{M}, a \models \Box s &\quad\approx\quad \text{all successors of } a \text{ satisfy } s\\
\mathcal{M}, a \models \Diamond^* s &\quad\approx\quad \text{some node reachable from } a \text{ satisfies } s\\
\mathcal{M}, a \models \Box^* s &\quad\approx\quad \text{all nodes reachable from } a \text{ satisfy } s
\end{aligned}$$

- A formula is *satisfiable* if it holds at some state in some model
- Interpreted classically: Every state of every model satisfies $s \vee \neg s$

Related work

- $K^* \approx$ basic modal logic $+$ eventualities $(\Diamond^*) \approx$ stripped down PDL
- Eventualities cause non-compactness
- K^* has the small model property [Fischer Ladner '79]
- EXPTIME decision procedure for satisfiability [Pratt '79]

- This work: based on recent account of Pratt-style decision procedures for extensions of PDL [Kaminski, Schneider, Smolka 2011]

Representation in Coq

- A faithful representation consists of:
  - Syntax (trivial)
  - Models
  - Evaluation relation
- Defines a satisfiability relation
- Faithful if equivalent to external (set threoretic) satisfiability relation

Models and Evaluation of Formulas

- Naive representation:

```
Record model := Model {
  state :> Type ;
  trans  :  state → state → Prop ;
  label  :  var → state → Prop }
```

- Direct evaluation into Prop does not capture classical logic
- Design decision: evaluate formulas to bool :

```
eval : forall M : model , form → pred M
```

pred M ≈ boolean predicates on (states of) M

Formulas as Boolean Predicates

- Formulas: $s ::= p \mid \neg p \mid s \vee s \mid s \wedge s \mid \Diamond s \mid \Box s \mid \Diamond^* s \mid \Box^* s$
- Need: boolean logical operators:
  $\wedge, \vee$: **forall** M, pred M $\longrightarrow$ pred M $\longrightarrow$ pred M
  $\neg, \Diamond, \Box, \Diamond^*, \Box^*$: **forall** M, pred M $\longrightarrow$ pred M
- Use boolean labeling function:

```
Record model := Model {
  state :> Type
  ...
  label : var → pred state }
```

- Propositional connectives are definable
- Modal operators do not preserve decidability of predicates.

Interpreting Modalities

- Simple specification of modalities (in Prop)

$$DIA\ trans\ p\ w \equiv \exists v.\ trans\ w\ v \land p\ v$$
$$DSTAR\ trans\ p\ w \equiv \exists v.\ trans^{*}\ w\ v \land p\ v$$

- Neither $\exists$ nor * preserve decidability

Interpreting Modalities

- Simple specification of modalities (in Prop)

$$DIA\ trans\ p\ w \equiv \exists v.\ trans\ w\ v \wedge p\ v$$
$$DSTAR\ trans\ p\ w \equiv \exists v.\ trans^*\ w\ v \wedge p\ v$$

- Neither $\exists$ nor * preserve decidability
- Require models to provide boolean modal operators

```
Record model := Model {
  ...
  DIAb : pred state → pred state;
  DIAbP (p:pred state) w : (DIA trans p w) ↔ (DIAb p w = true);

  DSTARb : pred state → pred state;
  DSTARbP (p:pred state) w : (DSTAR trans p w) ↔ (DSTARb p w = true)
}.
```

- Boolean modal operators for $\Box$ and $\Box^*$ are definable

Faithful Representation in Coq

- Allows the definition of a boolean evaluation function

```
Fixpoint eval (M:model) (s:form) : (pred M) :=
  match s with
    Var v => label v |  ...  | Box s => BOXb (eval M s) | ...
  end.
Notation "M , w |= s" := (eval M s w).
```

- Evaluation satisfies the usual classical equivalences:

$$
\begin{aligned}
p \vee \neg p &\equiv \top \\
\Diamond^* s &\equiv s \vee \Diamond\Diamond^* s \\
\Box^* s &\equiv s \wedge \Box\Box^* s
\end{aligned}
$$

Localized Classical Assumptions

- If we were to assume

  ```
  Axiom IXM : forall P, { P } + { ~ P }
  ```

  DIAb and DSTARb would be definable
- Boolean logical operators regarded as *localized* classical assumptions
- Here: Assume what is needed to obtain a boolen evaluation

Satisfiability and Demos

Theorem

Satisfiability of formulas is decidable

- We define syntactic models called *demos* such that:
  1. The states of a demo are sets of formulas
  2. Every state of a demo satisfies all formulas it contains

Satisfiability and Demos

Theorem

Satisfiability of formulas is decidable

- We define syntactic models called *demos* such that:
  1. The states of a demo are sets of formulas
  2. Every state of a demo satisfies all formulas it contains
- A formula is satisfiable iff it is contained in demo built from its subformulas

Satisfiability and Demos

Theorem

Satisfiability of formulas is decidable

- We define syntactic models called *demos* such that:
  1. The states of a demo are sets of formulas
  2. Every state of a demo satisfies all formulas it contains
- A formula is satisfiable iff it is contained in demo built from its subformulas
- For every formula there are only finitely many demos to consider
- Yields decidability of satisfiability

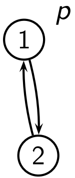
Example Demo

Demos are sets of sets of formulas

$\boxed{\Diamond\Diamond p, \Box\neg p, p}$

$\boxed{\Diamond p, \neg p}$

$\rightsquigarrow$



Every demo $\mathcal{D}$ can be seen as a model $M_{\mathcal{D}}$

- states: elements of $\mathcal{D}$
- transitions: $H \rightarrow_{\mathcal{D}} H'$ iff $\{s \mid \Box s \in H\} \subseteq H'$
- labels: H is labeled with p iff $p \in H$

Consistency Conditions

Need conditions that ensure:

Lemma (Model Existence)

If $\mathcal{D}$ is a demo and $t \in H \in \mathcal{D}$, then $M_{\mathcal{D}}, H \models t$.

Consistency Conditions

Need conditions that ensure:

Lemma (Model Existence)

If $\mathcal{D}$ is a demo and $t \in H \in \mathcal{D}$, then $M_{\mathcal{D}}, H \models t$.

- Local consistency - The states of a demo are Hintikka sets:
  1. If $\neg p \in H$, then $p \notin H$.
  2. If $s \wedge t \in H$, then $s \in H$ and $t \in H$.
  3. If $s \vee t \in H$, then $s \in H$ or $t \in H$.
  4. If $\Box^* s \in H$, then $s \in H$ and $\Box\Box^* s \in H$.
  5. If $\Diamond^* s \in H$, then $s \in H$ or $\Diamond\Diamond^* s \in H$.

Consistency Conditions

Need conditions that ensure:

Lemma (Model Existence)

If $\mathcal{D}$ is a demo and $t \in H \in \mathcal{D}$, then $M_{\mathcal{D}}, H \models t$.

- Local consistency - The states of a demo are Hintikka sets:
  1. If $\neg p \in H$, then $p \notin H$.
  2. If $s \wedge t \in H$, then $s \in H$ and $t \in H$.
  3. If $s \vee t \in H$, then $s \in H$ or $t \in H$.
  4. If $\Box^* s \in H$, then $s \in H$ and $\Box\Box^* s \in H$.
  5. If $\Diamond^* s \in H$, then $s \in H$ or $\Diamond\Diamond^* s \in H$.
- Global consistency - All diamonds are realized:

(D$\Diamond$) If $\Diamond s \in H \in \mathcal{D}$, then $H \to_{\mathcal{D}} H'$ and $s \in H'$ for some $H' \in \mathcal{D}$.

(D$\Diamond^*$) If $\Diamond^* s \in H \in \mathcal{D}$, then $H \to^*_{\mathcal{D}} H'$ and $s \in H'$ for some $H' \in \mathcal{D}$.

Decidability of Satisfiability

- Fix some formula s_0 and let $\mathcal{F}$ denote the syntactic closure of s_0
- Solve the satisfiability problem for formulas in $\mathcal{F}$

Lemma (Model Existence)

If $\mathcal{D} \in 2^{2^{\mathcal{F}}}$ is a demo and $t \in H \in \mathcal{D}$, then $M_{\mathcal{D}}, H \models t$.

Theorem (Small Model Theorem)

Let $s \in \mathcal{F}$ and $M, w \models s$.
There exists a demo $\mathcal{D} \in 2^{2^{\mathcal{F}}}$ and $H \in \mathcal{D}$ such that $s \in H$

- Satisfiability for all formulas follows from $s_0 \in \mathcal{F}$

Formalization Setup

- Can fix a formula s_0 throughout the proof.
- We only require Hintikka sets $H \subseteq \mathcal{F}$ ($\mathcal{F}$ is finite)

Formalization Setup

- Can fix a formula s_0 throughout the proof.
- We only require Hintikka sets $H \subseteq \mathcal{F}$ ($\mathcal{F}$ is finite)
- Required data-structures:
  - Models: boolean functions reflecting predicates, ...
  - Decidability proof: finite syntactic closure, finite sets, sets of finite sets, boolean quantifiers, ...
- Little support for these structures in the Coq Standard Library

Formalization Setup

- Can fix a formula s_0 throughout the proof.
- We only require Hintikka sets $H \subseteq \mathcal{F}$ ($\mathcal{F}$ is finite)
- Required data-structures:
  - Models: boolean functions reflecting predicates, ...
  - Decidability proof: finite syntactic closure, finite sets, sets of finite sets, boolean quantifiers, ...
- Little support for these structures in the Coq Standard Library
- The Ssreflect extension provides all this (and much more)

Formalization

Representation:

fixed formula s_0	$\rightsquigarrow$	Section variable
syntactic closure of s_0	$\rightsquigarrow$	finite type F
Hintikka sets over $\mathcal{F}$	$\rightsquigarrow$	boolean predicate on {set F}
Demos over $\mathcal{F}$	$\rightsquigarrow$	boolean predicate on {set {set F}}

Formalization

Representation:

fixed formula s_0	$\rightsquigarrow$	Section variable
syntactic closure of s_0	$\rightsquigarrow$	finite type F
Hintikka sets over $\mathcal{F}$	$\rightsquigarrow$	boolean predicate on {set F}
Demos over $\mathcal{F}$	$\rightsquigarrow$	boolean predicate on {set {set F}}

Lemma (Model Existence)

If $\mathcal{D} \in 2^{2^{\mathcal{F}}}$ is a demo and $t \in H \in \mathcal{D}$, then $M_{\mathcal{D}}, H \models t$.

- Requires the construction of a finite model
- Interpretations for modalities (DIAb, ...) definable for finite carriers

Demo construction

Theorem (Small Model Theorem)

Let $s \in \mathcal{F}$ and $M, w \models s$.
There exists a demo $\mathcal{D} \in 2^{2^{\mathcal{F}}}$ and $H \in \mathcal{D}$ such that $s \in H$

- Construct largest demo with pruning algorithm [Pratt '79]

Pruning

Pruning Algorithm:

- $S := \{H \subseteq \mathcal{F} \mid \text{H is a Hintikka set}\}$
- while S is not a demo,
  remove some H violating $(D\Diamond)$ or $(D\Diamond^*)$

Lemma

1. *All pruned sets are unsatisfiable*
2. *Pruning terminates with demo containing exactly the satisfiable Hintikka sets*

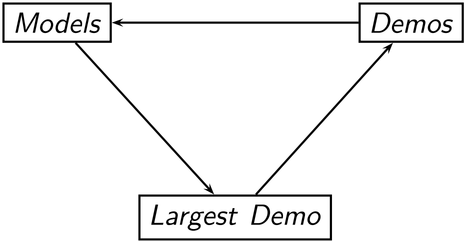
Definition largest_demo := prune [H | hintikka H]
Theorem decidability (s:F) :

$$\text{sat s} \leftrightarrow \underbrace{\textbf{existsb}\ \text{H : \{set F\}, H \textbackslash in largest\_demo \&\& s \textbackslash in H}}_{\text{boolean predicate over } s} = \text{true}$$

Corresponds to worst-case optimal exponential decision procedure

Models and Satisfiability

- Every class of models defines a satisfiability relation
- We have seen three variants:



- All three are constructively equivalent

Summary

- Constructive formalization of classical modal logic
  - Syntax
  - Models (boolean logical operations)
  - Boolean evaluation of formulas
  - Formalized small model theorem
  - Formal proof of decidability

```
forall s : form , { sat s } + { ~ sat s }
```

- Design space for the representation of models:
  - Allows definition of two-valued evaluation relation
  - Finite models need to be constructible

$\Rightarrow$ Many other possibilities

- Future Work:
  - Scale to richer logics like PDL/CTL
  - Consider other logics with the small model property

The Model Based Proof

The classical proof of the small model theorem is model based:

Theorem (Small Model Theorem)

Let $s \in \mathcal{F}$ and $M, w \models s$.
There exists a demo $\mathcal{D} \in 2^{2^{\mathcal{F}}}$ and $H \in \mathcal{D}$ such that $s \in H$

- Proof Idea:
  - Define $H_w := \{t \in \mathcal{F} \mid M, w \models t\}$
  - The set $\{H_w \mid w \in |M|\}$ is a demo containing s
- This expands to: $\{H \mid \underbrace{\exists w \in |M|.\ H_w = H}_{\text{not a boolean statement}}\}$
- finite sets $\approx$ extensional *boolean* predicates over finite domain
- Cannot define the set $\{H_w \mid w \in |M|\}$ as a finite set in Coq

The Model Based Proof

Extend the model with a boolean existential quantifier:

```
Record model := Model {
  ...
  exb : (pred state) → bool ;
  exbP (p:pred state) : (exists x , p x) ↔ (exb p = true) }.
```

- $\{H_w \mid w \in |M|\}$ definable as $\{H \mid exb\ w : M, H == H_w\}$

```
Theorem decidability (s : F) :
sat s ↔ (existsb D : {set {set F}},
         demo D && existsb H, H \in D && s \in H) = true
```

$\underbrace{\text{demo D \&\& existsb H, H \textbackslash in D \&\& s \textbackslash in H)}}_{\text{boolean statement}}$

Corresponds to the naive double exponential decision procedure