Coinduction in Algorithmic Proof Theory: Modal Logics

Dirk Pattinson, Imperial College London

(Joint work with Clemens Kupke and Rajeev Gorè)

Swansea (Wessex), Nov 2009

Modal Logics

Modal Logics

- Extensions of classical propositional logics with *modal operators*
- idea: modal operatorsoperators qualify the truth value of a judgement

Examples.

- Kripke Modal Logic: $\Box A$ reads as "A holds *necessarily*"
- Conditional Logic: $A \Rightarrow B$ reads as "B holds *under condition* A"
- Probabilistic Modal Logic: $\langle p \rangle \phi$ reads as "ϕ holds *with probability* $\geq p$"

Applications.

- Automated Reasoning in general: "good" decidability
- Knowledge Representation in particular: "good" expressive power

Automated Reasoning in Modal Logics

Semantical Methods

- typically developed for logics on an individual basis
- often via isolating a corresponding notion of automata
- (by and large) not efficiently implementable

Syntactical Methods

- via sequent calculi or tableau systems
- decidability via subformula property
- vast array of optimisation techniques available

Real-World Applications

- Semantic Web: modal reasoners underly semantical classification
- (medical) ontologies: e.g. SNOMED CT or Galen

Tableau Calculi: The Good, The Bad and The Ugly

The Good. Loop-free calculi such as the one for Kripke Modal Logic:

$$(\wedge)\frac{\Gamma, A \wedge B}{\Gamma, A, B} \qquad (\vee)\frac{\Gamma, A \vee B}{\Gamma, A \qquad \Gamma, B} \qquad (K)\frac{\Diamond A_0, \Box A_1, \ldots, \Box A_n}{A_0, A_1, \ldots, A_n}$$

Loops never arise in proof search

The Bad. Additional Axioms such as reflexivity $\Box A \to A$

$$(T)\frac{\Gamma, \Box A}{\Gamma, A, \Box A}$$

Loops can be handled via contraction

The Ugly. Global Assumptions Δ (e.g. $\Delta = \Diamond\top$) that force infinite paths

$$(K_\Delta)\frac{\Diamond A_0, \Box A_1, \ldots, \Box A_n}{A_0, A_1, \ldots, A_n, \Delta}$$

Need clever (blocking) techniques to avoid being trapped in infinite paths

Example: Modal Logic K with global assumptions Δ

Syntax.

$$A ::= p \mid \bar{p} \mid A \wedge B \mid A \vee B \mid \Box A \mid \Diamond A \quad (p \in V)$$

Semantics. Kripke Models (W, R, π) where $R \subseteq W \times W$ and $\pi : V \to \mathcal{P}(W)$

$$w \models \Box A \quad \text{iff} \quad \forall w'.(wRw' \Rightarrow w' \models A)$$

$$w \models \Diamond A \quad \text{iff} \quad \exists w'.(wRw' \;\&\; w' \models A)$$

Tableau Rules.

$$\frac{\Diamond A_0, \Box A_1, \ldots, \Box A_n}{A_0, A_1, \ldots, A_n} \qquad \frac{\text{to satisfy } \Diamond A_0, \Box A_1, \ldots, \Box A_n}{\text{need successor that satisfies } A_0, A_1, \ldots, A_n}$$

In presence of global assumptions Δ

$$\frac{\Diamond A_0, \Box A_1, \ldots, \Box A_n}{A_0, A_1, \ldots, A_n, \Delta} \qquad \frac{\text{to satisfy } \Diamond A_0, \Box A_1, \ldots, \Box A_n}{\text{need successor that additionally satisfies } \Delta}$$

Soundness, Completeness and Closed Tableaux

General Setup. Tableau System (S, T) where

- S is a set of *sequents*, i.e. conjunctive (multi)sets of formulas $A_1, \ldots, A_n$
- T is a set of *rules* of the form Γ/Ψ where $\Gamma \in S$ and $\Psi \subseteq S$ is finite

(Axioms $p, \bar{p}, \Gamma$ are rules $p, \bar{p}, \Gamma/\emptyset$ without conclusions)

Closed Tableaux. Well-founded trees with $\emptyset$ at the leaves.

Intuition. A closed tableau with root node Γ is a witness of unsatisfiability of Γ.

Obvious Lemma. $\Gamma \in S$ *has a closed tableau* if $\Gamma \in \mu M$ where

$$M : \mathcal{P}(S) \to \mathcal{P}(S), M(X) = \{\Gamma \in S \mid \exists \Psi \subseteq X \, (\Gamma/\Psi \in T)\}$$

and μM is M's least fixpoint.

(Blanket assumption: soundness and completeness, i.e. $\Gamma \in \mu M$ iff Γ unsat.)

Algorithmic Satisfiability Checking

General Assumption. The set S of sequents in a tableau system is *finite*.

Algorithm 1. Check whether $\Gamma \in S$ has a closed tableau

1. compute the least fixpoint μM of M – this is possible since S is finite
2. check whether $\Gamma \in \mu M$

Pro: obviously correct. **Con:** practically infeasible due to large "footprint"

Practical Algorithms.

- start with the sequent Γ we're interested in – no unneccessary "junk"
- imitate backward proof search – but cut off potentially infinite branches

Example. Ancestor Equality Blocking

"Discard Branches on which the same sequent appears twice"

Blocking, Coinductively

Recall. Γ has a closed tableau if $\Gamma \in \mu M$, where

$$M : \mathcal{P}(S) \to \mathcal{P}(S), M(X) = \{\Gamma \in S \mid \exists \Psi \subseteq X \, (\Gamma/\Psi \in T)\}$$

Obvious Lemma. $\Gamma \in S$ *doesn't* have a closed tableau iff $\Gamma \in \nu W$ where

$$W : \mathcal{P}(S) \to \mathcal{P}(S), W(X) = \{\Gamma \in S \mid \forall \Gamma/\Psi \in T \, (\Psi \cap X \neq \emptyset)\}$$

and νW is W's *greatest* fixpoint.

Dual Problem. Decide whether $\Gamma \in \nu W$ – but why should this be easier?

Enter Coinduction. $\Gamma \in \nu W$ iff $\Gamma \in F$ and F is *co-closed*:

$$\Gamma \in \nu W \iff \exists F \subseteq S \, (\Gamma \in F \text{ and } (F \subseteq W(F))$$

Algorithmically. Try and expand Γ to a co-closed set – no unneccessary “junk”

Proof Search, Coinductively

Question. How do we expand $\{\Gamma\}$ into a coclosed set?

Observation. If $\mathcal{B} \subseteq S$ is *not* co-closed, then there's a "culprit", i.e.

$$\exists \Gamma \in \mathcal{B} \text{ and } \Gamma/\Psi \in T \; (\Psi \cap \mathcal{B} = \emptyset)$$

and $\mathcal{B}$ will become "more" coclosed if we add an element of Ψ.

Expansion. Define a relation $\longrightarrow$ on "branches" $\mathcal{B} \subseteq S$ by

$$\mathcal{B} \longrightarrow \mathcal{B}' \iff \exists \Gamma \in \mathcal{B}, \Gamma/\Psi \in T, \Psi \cap \mathcal{B} = \emptyset, \mathcal{B} \subseteq \mathcal{B}' \subseteq \mathcal{B} \cup \Psi$$

and $\mathcal{B} \not\longrightarrow$ if $\{\mathcal{B}' \mid \mathcal{B} \longrightarrow \mathcal{B}'\} = \emptyset$.

Theorem. Γ does *not* have a closed tableau if and only if $\{\Gamma\} \longrightarrow^* \mathcal{B} \not\longrightarrow$.

Proof. Easy exercise in coinduction.

Algorithmic Digression

Algorithm 2. Check satisfiability of $\Gamma \in S$ (Initialise $\mathcal{B} = \{\Gamma\}$).

1. if $\mathcal{B}$ is coclosed, then say "satisfiable"
2. choose $\mathcal{B}'$ so that $\mathcal{B} \to \mathcal{B}'$. If $\mathcal{B} \subsetneq \mathcal{B}'$ let $\mathcal{B} = \mathcal{B}'$ and goto 1

Correctness. Γ is satisfiable iff there exists a run of Algorithm 2 that says so.

Practicality. minimal "footprint" and in fact implemented in lots of reasoners

Complexity. For logics "in the wild":

- S is exponential in the size of Γ as S contains all sets of subformulas of Γ
- which gives a worst-case complexity of NEXPTIME ;-(

BUT: naive fixpoint computation can be performed in EXPTIME – can we do better?

Global Caching

Caching. Keep track of satisfiable sequents (E), unsatisfiable sequents (A) and undecided sequents (U) – compute least/greatest fixpoints *locally*

Expansion and Propagation. Define two relations $\rightarrow_E$ ("expand") and $\rightarrow_P$ ("propagate") on triples(!) (A, U, E) of sets of sequents by

$(A, U, E) \rightarrow_E (A', U', E')$ iff $\quad U' = (U \cup \Psi) \setminus (A \cup E)$ for some $\Gamma \in U$ with $\Gamma/\Psi \in T$

and $A' = A$, $E' = E$ for expansion, and

$$(A, U, E) \rightarrow_P (A', U', E') \text{ iff } A' = A \cup \mu M \quad E' = E \cup \nu W$$

and $U' = U$ where $M, W : \mathcal{P}(U) \rightarrow \mathcal{P}(U)$ are given by

$$M(X) = \{\Gamma \in U \mid \exists \Psi \subseteq X \cup A \, (\Gamma/\Psi \in T)$$
$$W(X) = \{\Gamma \in U \mid \forall \Gamma/\Psi \in T \, (\Psi \cap (X \cup E) \neq \emptyset\}$$

for propagation. Write $(A, U, E) \circlearrowleft$ if $(A, U, E) \rightarrow_E (A', U', E')$ implies that $(A, U, E) = (A', U', E')$.

Correctness

Proposition. Suppose that

$$(\emptyset, \{\Gamma\}, \emptyset) \to^*_{PE} (A, U, E) \to_P (A', U', E') \circlearrowleft$$

Then Γ has a closed tableau iff $\Gamma \in A'$.

Algorithm 3 Decide whether $\Gamma \in S$ has a closed tableau.

1. Let $(A, U, E) = (\emptyset, \{\Gamma\}, \emptyset)$
2. while $(A, U, E) \not\circlearrowleft$ do
   (a) let $(A, U, E) = (A', U', E')$ where $(A, U, E) \to_E (A', U', E')$
   (b) (opt.) let $(A, U, E) = (A', U', E')$ where $(A, U, E) \to_P (A', U', E')$
3. let $(A, U, E) = (A', U', E')$ where $(A, U, E) \to_P (A', U', E')$
4. say "yes" if $\Gamma \in A$ and "no" otherwise

Note. This algorithm is nondeterministic, but we only need to check *one* execution.

Analysis

Correctness. Γ has a closed tableau iff Algorithm 3 says "yes"

Practicality. "footprint" still minimal and encouraging experimental results

Complexity. For logics "in the wild":

- S is (still) exponential in the size of Γ
- BUT: if rules are well-behaved (exponentially decidable), every run of Algorithm 3 takes at most exponential time

Well-behaved rules. Practically all logics one encounters (probabilistic, graded, conditional etc.) – but non-examples can be constructed.

Best of Both Worlds? Optimal Complexity plus small "footprint" – experiments!

Implementation and Applications

Fixpoint Computation. Scanning through *all* rules at every step is impractical:

- enrich triples (A, U, E) with information on rule conclusions $\leadsto$ bipartite graph
- graph structure links sequents to conclusions: $\Gamma \longrightarrow_L \Psi$ if $\Gamma/\Psi \in T$,
- and conclusions to their elements: $\Psi \longrightarrow_L \Gamma$ if $\Gamma \in \Psi$

Motivating Application. Global Consequence Reasoning for *Coalgebraic Logics*

- instantiates to lots of special cases: relational, probabilistic, graded
- medium term goal: implement a practical reasoner based on coalgebraic logics
- driving force: applications in knowledge representation