Non-classical logics

Lecture 13: Modal logics (Part 6)

Viorica Sofronie-Stokkermans

sofronie@uni-koblenz.de

Until now

- Modal logic

  Syntax/Semantics

  Correspondence theory

  First-order definability

  Theorem proving in modal logics

  Decidability

- Description logics
- Dynamic logics

Dynamic Logic

Motivation (last time)

Dynamic Logic

Dynamic logic is a language for specifying programming languages.

The original work on dynamic logic is by Vaughan Pratt (1976) and by David Harel (1979).

Propositional Dynamic Logic

Propositional dynamic logic (PDL) is a multi-modal logic with structured modalities.

For each program α, there is:

- a box-modality $[\alpha]$ and
- a diamond modality $\langle\alpha\rangle$.

PDL was developed from first-order dynamic logic by Fischer-Ladner (1979) and has become popular recently.

Here we consider regular PDL.

Propositional Dynamic Logic

Syntax

Prog set of programs

$\mathrm{Prog}_0 \subseteq \mathrm{Prog}$: set of atomic programs

Π: set of propositional variables

The set of formulae $\mathrm{Fma}^{PDL}_{\mathrm{Prog},\Pi}$ of (regular) propositional dynamic logic and the set of programs P_0 are defined by simultaneous induction as follows:

PDL: Syntax

Formulae:

$$
\begin{array}{lll}
F, G, H & ::= & \bot \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(falsum)} \\
& \mid & \top \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(verum)} \\
& \mid & p \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad p \in \Pi_0 \text{ (atomic formula)} \\
& \mid & \neg F \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(negation)} \\
& \mid & (F \wedge G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(conjunction)} \\
& \mid & (F \vee G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(disjunction)} \\
& \mid & (F \rightarrow G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(implication)} \\
& \mid & (F \leftrightarrow G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(equivalence)} \\
& \mid & [\alpha]F \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } \alpha \in \text{Prog} \\
& \mid & \langle\alpha\rangle F \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{if } \alpha \in \text{Prog}
\end{array}
$$

Programs:

$$
\begin{array}{lll}
\alpha, \beta, \gamma & ::= & \alpha_0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \alpha_0 \in \text{Prog}_0 \text{ (atomic program)} \\
& \mid & F? \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad F \text{ formula (test)} \\
& \mid & \alpha; \beta \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(sequential composition)} \\
& \mid & \alpha \cup \beta \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(non-deterministic choice)} \\
& \mid & \alpha^* \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(non-deterministic repetition)}
\end{array}
$$

Semantics

A PDL structure $\mathcal{K} = (S, R(), I)$ is a multimodal Kripke structure with an accessibility relation for each atomic program. That is it consists of:

- a non-empty set S of states
- an interpretation $R() : \text{Prog}_0 \to \mathcal{P}(S \times S)$ of atomic programs that assigns a transition relation $R(\alpha) \subseteq S \times S$ to each atomic program α
- an interpretation $I : \Pi \times S \to \{0, 1\}$

PDL: Semantics

The interpretation of PDL relative to a PDL structure $\mathcal{K} = (S, R(), I)$ is defined by extending $R()$ to Prog and extensing I to $\mathrm{Fma}^{PDL}_{\mathrm{Prop}_0}$ by the following simultaneously inductive definition:

Interpretation of formulae/programs

$$
\begin{array}{lll}
val_{\mathcal{K}}(p, s) & = & I(p, s) \\
val_{\mathcal{K}}(\neg F, s) & = & \neg_{\text{Bool}}\, val_{\mathcal{K}}(F, s) \\
val_{\mathcal{K}}(F \wedge G, s) & = & val_{\mathcal{K}}(F, s) \wedge_{\text{Bool}} val_{\mathcal{K}}(G, s) \\
val_{\mathcal{K}}(F \vee G, s) & = & val_{\mathcal{K}}(F, s) \vee_{\text{Bool}} val_{\mathcal{K}}(G, s) \\
val_{\mathcal{K}}([\alpha]F, s) = 1 & \textit{iff} & \text{for all } t \in S \text{ with } (s, t) \in R(\alpha),\ val_{\mathcal{K}}(F, t) = 1 \\
val_{\mathcal{K}}(\langle\alpha\rangle\, F, s) = 1 & \textit{iff} & \text{for some } t \in S \text{ with } (s, t) \in R(\alpha),\ val_{\mathcal{K}}(F, t) = 1 \\
R([F?]) & = & \{(s, s) \mid val_{\mathcal{K}}(F, s) = 1\} \\
 & & (F? \text{ has the same meaning as: if } F \text{ then skip else do not termina} \\
R(\alpha \cup \beta) & = & R(\alpha) \cup R(\beta) \\
R(\alpha; \beta) & = & \{(s, t) \mid \text{ there exists } u \in S \ s.t. (s, u) \in R(\alpha) \text{ and } (u, t) \in R(\beta) \\
R(\alpha^*) & = & \{(s, t) \mid \text{ there exists } n \geq 0 \text{ and there exist } u_0, \ldots, u_n \in S \text{ with} \\
 & & s = u_0, y = u_n, (u_0, u_1), \ldots, (u_{n-1}, u_n) \in R(\alpha)\}
\end{array}
$$

Interpretation of formulae/programs

- $(\mathcal{K}, s)$ satisfies F (notation $(\mathcal{K}, s) \models F$) iff $val_{\mathcal{K}}(F, s) = 1$.
- F is valid in $\mathcal{K}$ (notation $\mathcal{K} \models F$) iff $(\mathcal{K}, s) \models F$ for all $s \in S$.
- F is valid (notation $\models F$) iff $\mathcal{K} \models F$ for all PDL-structures $\mathcal{K}$.

Axiom system for PDL

Comp :	$[\alpha;\beta]A \leftrightarrow [\alpha][\beta]A$,
Alt :	$[\alpha \cup \beta]A \leftrightarrow [\alpha]A \wedge [\beta]A$,
Mix :	$[\alpha^*]A \rightarrow A \wedge [\alpha][\alpha^*]A$,
Ind :	$[\alpha^*](A \rightarrow [\alpha]A) \rightarrow (A \rightarrow [\alpha^*]A)$,
Test :	$[A?]B \leftrightarrow (A \rightarrow B)$.

We will show that PDL is determined by PDL structures, and has the finite model property.

Soundness and Completeness of PDL

Proof similar to the proof in the case of the modal system K (with small differences)

Theorem. If the formula F is provable in the inference system for PDL then F is valid in all PDL structures.

Proof: The axioms are valid in every PDL structure. Easy computation (examples on the blackboard).

The inference rules *MP* and *Gen* preserve validity.

(proof as for the modal logic K)

Soundness and Completeness of PDL

Theorem. If the formula F is is valid in all PDL structures then F is provable in the inference system for PDL.

Proof

Idea:

Assume that F is not provable in the inference system for PDL.

We show that:

(1) $\neg F$ is consistent with the set L of all theorems of PDL

(2) We can construct a "canonical" PDL structure $\mathcal{K}_L$ and a state w in this PDL structure such that $(\mathcal{K}, w) \models \neg F$.

Contradiction!

Consistent sets of formulae

Let L be a set of PDL formulae which:
(1) contains all propositional tautologies
(2) contains axiom PDL
(3) is closed under modus ponens and generalization
(4) is closed under instantiation

Definition. A subset $F \subseteq L$ is called L-inconsistent iff there exist formulae $A_1, \ldots, A_n \in F$ such that

$$(\neg A_1 \vee \cdots \vee \neg A_n) \in L$$

F is called L-consistent iff it is not L-inconsistent.

Definition. A consistent set F of PDL formulae is called maximal L-consistent if for every formula A wither $A \in F$ or $\neg A \in F$.

Consistent sets of formulae

Let L be as before. In what follows we assume that L is consistent.

Theorem. Let F be a maximal L-consistent set of formulae. Then:

(1) For every formula A, either $A \in F$ or $\neg A \in F$, but not both.

(2) $A \vee B \in F$ iff $A \in F$ or $B \in F$

(3) $A \wedge B \in F$ iff $A \in F$ and $B \in F$

(4) $L \subseteq F$

(5) F is closed under Modus Ponens

Theorem. Every consistent set F of formulae is contained in a maximally consistent set of formulae.

Proofs: As for modal logic.

Canonical models

Goal: Assume F is not a theorem. Construct a PDL structure $\mathcal{K}$ and a state w of $\mathcal{K}$ such that $(\mathcal{K}, w) \models \neg F$.

States:

State of $\mathcal{K}$: maximal consistent set of formulae.

Intuition: $(\mathcal{K}, W) \models F$ iff $F \in W$.

Accessibility relation:

Intuition:
$(\mathcal{K}, W) \models [\alpha]F$ iff for all W', $((W, W') \in R(\alpha) \rightarrow (\mathcal{K}, W') \models F)$

Canonical models

Goal: Assume F is not a PDL theorem. Construct a PDL structure $\mathcal{K}$ and a state w of $\mathcal{K}$ such that $(\mathcal{K}, w) \models \neg F$.

States:

State of $\mathcal{K}$: maximal consistent set of formulae.

Intuition: $(\mathcal{K}, W) \models F$ iff $F \in W$.

Accessibility relation:

Intuition:
$(\mathcal{K}, W) \models [\alpha]F$ iff for all W', $((W, W') \in R(\alpha) \rightarrow (\mathcal{K}, W') \models F$
$[\alpha]F \in W$ iff for all W', $((W, W') \in R(\alpha) \rightarrow F \in W')$

$(W, W') \in R(\alpha)$ iff $W' \supseteq \{F \mid [\alpha]F \in W\}$

Canonical models

Theorem. $\mathcal{K}$ satisfies all PDL structure conditions except $R(\alpha^*) \subseteq (R(\alpha))^*$.

Proof: By direct checking.

Example: $R(\alpha;\beta) \subseteq R(\alpha) \circ R(\beta)$
Assume $(W, W') \in R(\alpha;\beta)$. Then $W' \subseteq \{F \mid [\alpha;\beta]F \in W\}$.

We want to show that there exists W_0 with $(W, W_0) \in R(\alpha)$ and $(W_0, W') \in R(\beta)$.

It is not difficult to show that $W_0 = \{B \mid [\alpha]B \in W\} \cup \{\neg[\beta]D \mid D \notin W'\}$ is PDL-consistent. For this, the PDL-theorem $[\alpha;\beta]A \leftrightarrow [\alpha][\beta]A$ is used.

Canonical models

Theorem. Assume F is not a PDL theorem. We can construct a PDL structure $\mathcal{K}'$ and a state w of $\mathcal{K}'$ such that $(\mathcal{K}', w) \models \neg F$.

Proof. To obtain a PDL structure that falsifies F we will collapse $\mathcal{K}$ by a suitable Γ that contains F. The closure rules for Γ that will be needed are:

- Γ is closed under subformulae;
- $[B?]D \in \Gamma$ implies $B \in \Gamma$;
- $[\alpha;\beta]B \in \Gamma$ implies $[\alpha][\beta]B \in \Gamma$;
- $[\alpha \cup \beta]B \in \Gamma$ implies $[\alpha]B, [\beta]B \in \Gamma$;
- $[\alpha^*]B \in \Gamma$ implies $[\alpha][\alpha^*]B \in \Gamma$

A set Γ satisfying these conditions will be called closed.

Completeness/Decidability of PDL

Theorem. If Γ is the smallest closed set containing a given formula F, then Γ is finite.

Proof. The point is to show that closing Subformulae(F) under the above rules produces only finitely many new formulae.

Define a formula to be boxed if it is prefixed by a modal connective, i.e. is of the form $[\alpha]B$ for some α and B. Each time we apply a closure rule, new boxed formulae appear on the right side of the rule, and further rules may apply to these new formulae.

But observe that the programs α indexing prefixes $[\alpha]$ on the right side are in all cases shorter in length than those indexing the prefix on the left of the rule in question. Hence we will eventually produce only atomic prefixes on the right, and run out of rules to apply.

Completeness/Decidability of PDL

Having determined that Γ, the smallest closed set containing F, is finite, we perform a Γ- filtration of $\mathcal{K}$.

Completeness/Decidability of PDL

Reminder (modal logic K):
Fix a model $\mathcal{K} = (S, R, I)$ and a set $\Gamma \subseteq Fma_\Sigma$ that is closed under subformulae, i.e. $B \in \Gamma$ implies Subformulae$(B) \subseteq \Gamma$.

For each $s \in S$, define

$$\Gamma_s = \{B \in \Gamma \mid (\mathcal{K}, s) \models B\}$$

and put $s \sim_\Gamma t$ iff $\Gamma_s = \Gamma_t$,

Then $s \sim_\Gamma t$ iff for all $B \in \Gamma$, $(\mathcal{K}, s) \models B$ iff $(\mathcal{K}, t) \models B$.

Fact: $\sim_\Gamma$ is an equivalence relation on S.

Let $[s] = \{t \mid s \sim_\Gamma t\}$ be the $\sim_\Gamma$-equivalence class of s.

Let $S_\Gamma := \{[s] \mid s \in S\}$ be the set of all such equivalence classes.

Decidability/Completeness

Goal: $(\mathcal{K}, s) \models A \quad \mapsto \quad (\mathcal{K}', s') \models A$, $\mathcal{K}' = (S', R', I')$.

Step 1: $S' := S_\Gamma$, where $\Gamma = \text{Subformulae}(S)$

Step 2: $I' : (\Pi \cap \Gamma) \times S' \to \{0, 1\}$ def. by $I'(P, [s]) = I(P, s)$

Step 3: R' def. e.g. by: $([s], [t]) \in R'$ iff $\exists s' \in [s], \exists t' \in [t]$: $(s', t') \in R$

Same construction for PDL (only we need to define a relation for each program).

Theorem: $\mathcal{K}'$ is a PDL structure (and a filtration of $\mathcal{K}$.

If $(\mathcal{K}, s) \not\models F$ then $(\mathcal{K}', [s]) \models \neg F$.
$\mapsto$ completeness.

Lemma. If Γ is finite, then S_Γ is finite and has at most 2^n elements, where n is the number of elements of Γ.
$\mapsto$ decidability

Conclusions

Although PDL appears to be more expressive than modal logic, it is still decidable (it has the finite model propety).

Proof calculi for PDL exist.

For really reasoning about programs, often *first order dynamic logic* is needed (undecidable)

First-Order Modal Logics

First-order modal logic

We introduce first-order modal logic and consider its relationship to classical first-order logic.

First-order modal logic

Syntax

Semantics

Syntax

Given:

A signature $\Sigma = (\Omega, \Pi)$,

A set X of variables

Syntax

Given:

A signature $\Sigma = (\Omega, \Pi)$,

A set X of variables

Terms are defined as for classical logic

Atomic formulae are defined as for classical logic

General first-order formulae

$$
\begin{array}{lll}
F, G, H & ::= & \bot \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(falsum)} \\
& \mid & \top \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(verum)} \\
& \mid & A \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(atomic formula)} \\
& \mid & \neg F \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(negation)} \\
& \mid & (F \wedge G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(conjunction)} \\
& \mid & (F \vee G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(disjunction)} \\
& \mid & (F \to G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(implication)} \\
& \mid & (F \leftrightarrow G) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(equivalence)} \\
& \mid & \Box F \\
& \mid & \Diamond F \\
& \mid & \forall x\ F \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(universal quantification)} \\
& \mid & \exists x\ F \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(existential quantification)}
\end{array}
$$

Semantics

A Kripke structure $K = (S, R, I)$ consists of Kripke frame $F = (S, R)$ and a mapping I that assigns to each world $s \in S$ a first-order structure

$$I(s) = (U_{I(s)}, \{f_{I(s)}\}_{f \in \Omega}, \{p_{I(s)}\}_{p \in \Pi})$$

such that, for each $s, t \in S$ with sRt, $I(s)$ is a substructure of $I(t)$, i.e.:

- the universe of $I(s)$ is a subset of the universe of $I(t)$ (monotonicity), and
- the structures $I(s)$ and $I(t)$ agree on the interpretation of all function symbols on the (smaller) universe of $I(s)$.

A Kripke structure (S, R, I) is called Kripke structure with constant domain if all the models $\{I(s) \mid s \in S\}$ are required to have the same universe.

Semantics

Interpretation of quantified modal formulae

$(\mathcal{K}, s) \models p(t_1, \ldots, t_k)$	iff	$I(s) \models p(t_1, \ldots, t_k)$ for atomic formulae $p(t_1, \ldots,$
$(\mathcal{K}, s) \models F \wedge G$	iff	$(\mathcal{K}, s) \models F$ and $(\mathcal{K}, s) \models G$
$(\mathcal{K}, s) \models F \vee G$	iff	$(\mathcal{K}, s) \models F$ or $(\mathcal{K}, s) \models G$
$(\mathcal{K}, s) \models F \rightarrow G$	iff	$(\mathcal{K}, s) \not\models F$ or $(\mathcal{K}, s) \models G$
$(\mathcal{K}, s) \models \neg F$	iff	$(\mathcal{K}, s) \not\models F$
$(\mathcal{K}, s) \models \Box F$	iff	for all t with $R(s, t)$, $(\mathcal{K}, s) \models F$
$(\mathcal{K}, s) \models \Diamond F$	iff	there exists t with $R(s, t)$ and $(\mathcal{K}, s) \models F$
$(\mathcal{K}, s) \models \forall x F(x)$	iff	for all $d \in U_{I(s)}$, $(\mathcal{K}, s) \models F(d)$
$(\mathcal{K}, s) \models \exists x F(x)$	iff	there exists $d \in U_{I(s)}$, $(\mathcal{K}, s) \models F(d)$

Semantics

Interpretation of quantified modal formulae

$(\mathcal{K}, s) \models p(t_1, \ldots, t_k)$	iff	$I(s) \models p(t_1, \ldots, t_k)$ for atomic formulae $p(t_1, \ldots,$
$(\mathcal{K}, s) \models F \wedge G$	iff	$(\mathcal{K}, s) \models F$ and $(\mathcal{K}, s) \models G$
$(\mathcal{K}, s) \models F \vee G$	iff	$(\mathcal{K}, s) \models F$ or $(\mathcal{K}, s) \models G$
$(\mathcal{K}, s) \models F \rightarrow G$	iff	$(\mathcal{K}, s) \not\models F$ or $(\mathcal{K}, s) \models G$
$(\mathcal{K}, s) \models \neg F$	iff	$(\mathcal{K}, s) \not\models F$
$(\mathcal{K}, s) \models \Box F$	iff	for all t with $R(s, t)$, $(\mathcal{K}, s) \models F$
$(\mathcal{K}, s) \models \Diamond F$	iff	there exists t with $R(s, t)$ and $(\mathcal{K}, s) \models F$
$(\mathcal{K}, s) \models \forall x F(x)$	iff	for all $d \in U_{I(s)}$, $(\mathcal{K}, s) \models F(d)$
$(\mathcal{K}, s) \models \exists x F(x)$	iff	there exists $d \in U_{I(s)}$, $(\mathcal{K}, s) \models F(d)$

Semantics

Goal: formalize the last statements using variable assignments, as for first-order logic

A (variable) assignment, also called a valuation (over a given Σ-algebra $\mathcal{A}$), is a map $\beta : X \to \mathcal{A}$.

Semantics

Goal: formalize the last statements using variable assignments, as for first-order logic

A (variable) assignment, also called a valuation (over a given Σ-algebra $\mathcal{A}$), is a map $\beta : X \to \mathcal{A}$.

Difficulty: the domains of the structures are different from world to world.

Semantics

In varying domain semantics, quantifiers may possibly refer to a different set of objects, depending on the world.

In constant domain semantics, quantifiers refer to the same set of objects (same universe U) in all worlds.

Variable assignment: $\beta : X \to U$.

Evaluation of quantified formulae as in classical first-order logic.

Proof calculi

Tableau calculi

Translation to first-order logic and resolution

Proof calculi

Tableau calculi

need to take into account γ and δ rules
(as for classical first-order logic)

Translation to first-order logic and resolution

out of the scope of this lecture

First-order Dynamic Logic

First-order dynamic logic (DL) extends PDL (although DL had been developed first) to a first-order logic and gives concrete atomic programs with specific effects, as opposed to abstract atomic programs with unknown effects as in PDL.

First-order Dynamic Logic

Definition. Let X be a set of variables. The set Fma^{DL} of formulas of dynamic logic and the set of programs Prog are defined by simultaneous induction as:

- $\bot, \top \in \text{Fma}^{DL}$ (propositional constants)
- All instances of formulas of first-order logic are in Fma^{DL}
- If $F, G \in \text{Fma}^{DL}$ then $\neg F, (F \wedge G), (F \vee G) \in \text{Fma}^{DL}$
- If $F \in \text{Fma}^{DL}$ and $x \in X$ is a variable then $\forall x F, \exists x F \in \text{Fma}^{DL}$
- If $F \in \text{Fma}^{DL}$, $\alpha \in \text{Prog}$ then $[\alpha]F, \langle\alpha\rangle F \in \text{Fma}^{DL}$
- $(x := t) \in \text{Prog}$ are atomic programs for variables $x \in X$ and terms t.
- If $F \in \text{Fma}^{DL}$ then $F? \in \text{Prog}$.
- If $\alpha, \beta \in \text{Prog}$ then $\alpha; \beta, \alpha \cup \beta, \alpha^* \in \text{Prog}$.

First-order Dynamic Logic

The semantics of DL is extended from that of PDL in the obvious way where the set of states is chosen to be $W := D^X$, i.e., the set of assignments $s : X \to D$ of elements of the domain D (of the first-order structure) to the variables X.

Predicate symbols, function symbols, and terms are interpreted as usual in first-order logic.

Because there are no other atomic programs, we only need to specify the accessibility relation belonging to an assignment $x := t$:

$$R(x := t) = \{(s, s') \mid s'(x) = s(t) \text{ and } s'(z) = s(z) \text{ for all } z \neq x\}$$

First-order Dynamic Logic

DL can inherit the axioms of first-order logic and of PDL. One typical axiom that DL needs in addition is an axiom that relates assignment to substitution:

$$(D15) \qquad \langle x := t \rangle F \leftrightarrow F_x^t$$

It says that formula F is true after assigning t to x if and only if F is true after substituting the new value t for x.

First-order Dynamic Logic

DL cannot be decidable because it includes first-order logic, where validity is only semidecidable.

But DL does not have a sound and complete effective calculus. Note here that DL formulas can state the halting problem for Turing machines. Nevertheless, there are proofs showing that DL has a relatively complete or arithmetically complete proof calculus.

Overview

- Many valued logics (and applications)
- Modal logics

  applications: description logics, dynamic logic

Overview

- Many valued logics (and applications)
- Modal logics

  applications: description logics, dynamic logic

Other non-classical logics

- Intuitionistic logic

  $\models A$ means "There exists a proof for A"
  can be modeled using the modal logic $S4$

- Temporal logic

  will be presented in the lecture "Formal Specification and Verification"

Perspectives

- Next semester: Seminar "Decision Procedures and Applications"

- Other Lectures:
  Decision procedures for verification
  Formal Specification and Verification

- This summer (end of August):
  Summer school "Verification Technology, Systems & Applications"

Forschungspraktika

BSc/MSc Theses in the area