Synthesis, Refinements and Search Strategies for Semantic Tableaux with Blocking

Renate A. Schmidt

School of Computer Science
The University of Manchester

1 July 2012

Tableau-based deduction

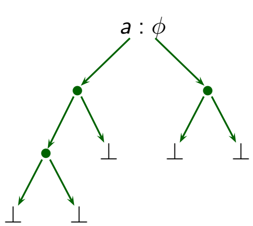
- Has a long tradition and is a well established method in AR
  Approach can be successfully used for a large number of logics
  Many implemented systems
  Multitude of different approaches

First-order logic	Smullyan ground sentence tableau, free-variable tableau, connection tableau, disconnection tableau, hypertableau, ...
Modal, description, hybrid, intuitionistic logics, ...	ground semantic tableau, tableau avoiding reference to semantics ...

- Our focus Ground semantic tableau calculi with blocking for mainly non-classical logics

The essence of tableau-based deduction

- Refutation approach, testing satisfiability (constructing a model)
- Goal-directed
- Rules break down formulae
- Rules for each logical operator
- Branching rules $\rightsquigarrow$ derivations are trees



$$\frac{s : \psi_1 \wedge \psi_2}{s : \psi_1,\ s : \psi_2} \qquad \frac{s : \psi_1 \vee \psi_2}{s : \psi_1 \mid s : \psi_2}$$

$$\frac{s : \Box\psi,\ (s,t) : R}{t : \psi}$$

$$\frac{s : \neg\Box\psi}{(s,u) : R,\ u : \neg\psi}\ (u \text{ new})$$

etc

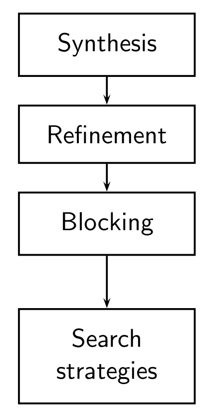
Process of developing a tableau prover for some logic

Define a sound and complete calculus
  Not difficult for semantically defined logics
  Calculi can be synthesised

Making tableau calculi effective
  Refining the rules

Ensure termination for decidable logics
  Devise blocking technique
  Various possibilities and challenges

Decide how to perform search
  Issues of turning the non-deterministic calculus it into a deterministic procedure
  Search strategies



Modal logic S4

Propositional modal logic $=$ propositional logic plus $\Box$

Formulae: $\phi, \psi \longrightarrow p_i \mid \bot \mid \neg\phi \mid \phi \wedge \psi \mid \Box\phi$

Semantics: Kripke model $\mathcal{M} = (W, R, v)$

$$
\begin{array}{lll}
\mathcal{M}, x \models p_i & \text{iff} & x \in v(p_i) \\
\mathcal{M}, x \not\models \bot & & \\
\mathcal{M}, x \models \neg\phi & \text{iff} & \mathcal{M}, x \not\models \phi \\
\mathcal{M}, x \models \phi \wedge \psi & \text{iff} & \mathcal{M}, x \models \phi \text{ and } \mathcal{M}, x \models \psi \\
\mathcal{M}, x \models \Box\phi & \text{iff} & \text{for all } R\text{-successors } y \text{ of } x \;\; \mathcal{M}, y \models \phi
\end{array}
$$

R is a pre-order, i.e., reflexive and transitive

Synthesis – Step 1: Specification in first-order logic [LMCS11]

- Specification of the semantics of the operators

$$
\begin{array}{c}
\forall x \left[\, \nu(\bot, x) \equiv \bot \,\right] \\
\forall x \left[\, \nu(\neg\phi, x) \equiv \neg\nu(\phi, x) \,\right] \\
\forall x \left[\, \nu(\phi_1 \wedge \phi_2, x) \equiv \nu(\phi_1, x) \wedge \nu(\phi_2, x) \,\right] \\
\forall x \left[\, \nu(\Box\phi, x) \equiv \forall y \left(R(x, y) \rightarrow \nu(\phi, y)\right) \,\right]
\end{array}
$$

- Specification of properties on R

$$
\begin{array}{c}
\forall x \, R(x, x) \\
\forall x \forall y \forall z \left(R(x, y) \wedge R(y, z) \rightarrow R(x, z)\right)
\end{array}
$$

Step 2: Extracting tableau rules

- Conversion for left-to-right definition of $\Box$:

$$
\begin{array}{l}
\forall x \left[\, \nu(\Box\phi, x) \rightarrow \forall y \left(R(x, y) \rightarrow \nu(\phi, y)\right) \,\right] \\
\nu(\Box\phi, x) \rightarrow .\neg R(x, y) \vee \nu(\phi, y) \\
\dfrac{\nu(\Box\phi, x)}{\neg R(x, y) \mid \nu(\phi, y)}
\end{array}
$$

- Conversion for right-to-left definition of $\Box$:

$$
\begin{array}{l}
\forall x \left[\, \nu(\Box\phi, x) \leftarrow \forall y \left(R(x, y) \rightarrow \nu(\phi, y)\right) \,\right] \\
\forall x \left[\, \neg\nu(\Box\phi, x) \rightarrow \neg\forall y \left(R(x, y) \rightarrow \nu(\phi, y)\right) \,\right] \\
\neg\nu(\Box\phi, x) \rightarrow .R(x, f(\phi, x)) \wedge \neg\nu(\phi, f(\phi, x)) \\
\dfrac{\neg\nu(\Box\phi, x)}{R(x, f(\phi, x)),\; \neg\nu(\phi, f(\phi, x))}
\end{array}
$$

$f(\phi, x) =$ Skolem term uniquely associated with $\neg\nu(\Box\phi, x)$

A synthesised tableau calculus for S4

- Decomposition rules

$$
\dfrac{\nu(\bot, x)}{\bot} \qquad \dfrac{\neg\nu(\bot, x)}{\neg\bot} \qquad \dfrac{\nu(\neg\phi, x)}{\neg\nu(\phi, x)} \qquad \dfrac{\neg\nu(\neg\phi, x)}{\nu(\phi, x)}
$$

$$
\dfrac{\nu(\phi_1 \wedge \phi_2, x)}{\nu(\phi_1, x),\; \nu(\phi_2, x)} \qquad \dfrac{\neg\nu(\phi_1 \wedge \phi_2, x)}{\neg\nu(\phi_1, x) \mid \neg\nu(\phi_2, x)}
$$

$$
\dfrac{\nu(\Box\phi, x)}{\neg R(x, y) \mid \nu(\phi, y)} \qquad \dfrac{\neg\nu(\Box\phi, x)}{R(x, f(\phi, x)),\; \nu(\phi, f(\phi, x))}
$$

- Closure rules

$$
\dfrac{\nu(\phi, x),\; \neg\nu(\phi, x)}{\bot} \qquad \dfrac{R(x, y),\; \neg R(x, y)}{\bot}
$$

- Theory rules

$$
\dfrac{}{R(x, x)} \qquad \dfrac{}{\neg R(x, y) \mid \neg R(y, z) \mid R(x, z)}
$$

Definition of rule application is so that all rules are grounding

Refinement 1: How to get rid of ν symbols?

- Define $:$ as a logical operator in the semantic specification

$$\forall x \; [\; \nu(s : \phi, x) \equiv \nu(\phi, \nu_0(s)) \;]$$

Transformation of rules:

$$\frac{\nu(\phi_1 \wedge \phi_2, x)}{\nu(\phi_1, x),\; \nu(\phi_2, x)} \quad \rightsquigarrow \quad \frac{s : \phi_1 \wedge \phi_2}{s : \phi_1,\; s : \phi_2}$$

$$\frac{\neg\nu(\phi_1 \wedge \phi_2, x)}{\neg\nu(\phi_1, x) \;\mid\; \neg\nu(\phi_2, x)} \quad \rightsquigarrow \quad \frac{s : \neg(\phi_1 \wedge \phi_2)}{s : \neg\phi_1,\; s : \neg\phi_2}$$

$:$ is not needed if the logic includes the @ operator of hybrid logic

- Gives transformation to labelled prefix tableau calculus

A labelled prefix tableau calculus for S4

- Decomposition rules

$$\frac{s : \bot}{\bot} \qquad \cancel{\frac{s : \neg\bot}{\neg\bot}} \qquad \cancel{\frac{s : \neg\phi}{s : \neg\phi}} \qquad \frac{s : \neg\neg\phi}{s : \phi}$$

$$\frac{s : \phi \wedge \psi}{s : \phi,\; s : \psi} \qquad \frac{s : \neg(\phi \wedge \psi)}{s : \neg\phi \;\mid\; s : \neg\psi}$$

$$\frac{s : \Box\phi}{\neg R(s, t) \;\mid\; t : \phi} \qquad \frac{s : \neg\Box\phi}{R(s, f(\phi, s)),\; f(\phi, s) : \phi}$$

- Closure rules

$$\frac{s : \phi,\; s : \neg\phi}{\bot} \qquad \frac{R(s, t),\; \neg R(s, t)}{\bot}$$

- Theory rules

$$\frac{}{R(s, s)} \qquad \frac{}{\neg R(s, t) \;\mid\; \neg R(t, u) \;\mid\; R(s, u)}$$

Standard rules for S4

- Standard ML tableau calculi include

$$\frac{s : \Box\phi,\; R(s, t)}{t : \phi} \quad \text{instead of} \quad \frac{s : \Box\phi}{\neg R(s, t) \;\mid\; t : \phi}$$

$$\frac{R(s, t),\; R(t, u)}{R(s, u)} \quad \text{instead of} \quad \frac{}{\neg R(s, t) \;\mid\; \neg R(t, u) \;\mid\; R(s, u)}$$

Less branching reduces the search space

- These examples suggest a general refinement principle

Rule refinement

Suppose *Tab* includes this rule, where $X_1 = \{F_1, \ldots, F_k\}$

$$\rho = \frac{X_0}{X_1 \;\mid\; \cdots \;\mid\; X_m}$$

- Refinement Tab' of Tab = Tab with ρ replaced by $\{\rho_1, \ldots, \rho_k\}$

$$\rho_j = \frac{X_0 \cup \{\sim F_j\}}{X_2 \;\mid\; \cdots \;\mid\; X_m} \qquad (j = 1, \ldots, k)$$

$\sim$ denotes complement

- Some properties
  - Each ρ_j is sound, if ρ is sound
  - Each ρ_j is derivable in Tab
  - In general, ρ is not derivable in Tab'

Soundness and completeness of refined tableau calculi

- Dagger condition – sufficient condition for completeness of Tab':
  In any Tab'-tableau derivation and every open branch $\mathcal{B}$,
  if $E_1, \ldots, E_k$ belong to $\mathcal{B}$ and $X_0\sigma = \{E_1, \ldots, E_k\}$ and
  each E_i holds in $\mathcal{I}(\mathcal{B})$, then

$$\mathcal{I}(\mathcal{B}) \models X_i\sigma \quad \text{for some } i = 1, \ldots, m$$

- Theorem (Refinement)
  1. Tab' is sound whenever Tab is sound
  2. If Tab is complete and the dagger condition holds in any Tab'-tableau derivation then Tab' is complete

Refining rules for S4

Dagger condition is true for

$$\frac{s : \Box\phi}{\neg R(s,t) \mid t : \phi} \quad \rightsquigarrow \quad \frac{s : \Box\phi,\ R(s,t)}{t : \phi}$$

$$\frac{}{\neg R(s,t) \mid \neg R(t,u) \mid R(s,u)} \quad \rightsquigarrow \quad \frac{R(s,t),\ R(t,u)}{R(s,u)}$$

Dagger condition is not true for

$$\frac{s : \neg(\phi \wedge \psi)}{s : \neg\phi \mid s : \neg\psi} \quad \rightsquigarrow \quad \frac{s : \neg(\phi \wedge \psi),\ s : \phi}{s : \neg\psi}$$

But it is true for

$$\frac{s : \neg(p \wedge \psi)}{s : \neg p \mid s : \neg\psi} \quad \rightsquigarrow \quad \frac{s : \neg(p \wedge \psi),\ s : p}{s : \neg\psi}$$

A refined labelled prefix tableau calculus for S4

- Decomposition rules

$$\frac{s : \bot}{\bot} \qquad \frac{s : \neg\neg\phi}{s : \phi}$$

$$\frac{s : \phi \wedge \psi}{s : \phi,\ s : \psi} \qquad \frac{s : \neg(p \wedge \psi),\ s : p}{s : \neg\psi} \qquad \frac{s : \neg(\phi \wedge \psi)}{s : \neg\phi \mid s : \neg\psi}$$

$$\frac{s : \Box\phi,\ R(s,t)}{t : \phi} \qquad \frac{s : \neg\Box\phi}{R(s, f(\phi, s)),\ f(\phi, s) : \phi}$$

- Closure rules

$$\frac{s : \phi,\ s : \neg\phi}{\bot}$$

$$\cancel{\frac{R(s,t),\ \neg R(s,t)}{\bot}}$$

- Theory rules

$$\frac{}{R(s,s)}$$

$$\frac{R(s,t),\ R(t,u)}{R(s,u)}$$

Only positive R-literals derived $\Rightarrow$ *S4* has a kind of tree model property

Synthesised calculus for $K_{(m)}(\neg)$

- Decomposition rules:

$$\frac{s : \neg\neg\phi}{s : \phi} \qquad \frac{s : \phi \vee \psi}{s : \phi \mid s : \psi} \qquad \frac{s : \neg(\phi \vee \psi)}{s : \neg\phi,\ s : \neg\psi}$$

$$\frac{s : [\alpha]\phi}{(s,t) : \neg\alpha \mid t : \phi} \qquad \frac{s : \neg[\alpha]\phi}{(s, f(\alpha, \phi, s)) : \alpha,\ f(\alpha, \phi, s) : \neg\phi}$$

$$\frac{(s,t) : \neg\neg\alpha}{(s,t) : \alpha}$$

- Closure rules:

$$\frac{s : \phi,\ s : \neg\phi}{\bot} \qquad \frac{(s,t) : \alpha,\ (s,t) : \neg\alpha}{\bot}$$

Question: Can the $[\cdot]$-rule be replaced by the refined non-branching version?

Rules synthesised from an alternative spec. for $K_{(m)}(\neg)$

- Decomposition rules:

$$\frac{s:\neg\neg\phi}{s:\phi} \qquad \frac{s:\phi\vee\psi}{s:\phi \mid s:\psi} \qquad \frac{s:\neg(\phi\vee\psi)}{s:\neg\phi,\ s:\neg\psi}$$

$$\frac{s:[\alpha]\phi}{(s,t):\neg\alpha \mid t:\phi} \qquad \frac{s:\neg[\alpha]\phi}{(s,f(\alpha,\phi,s)):\alpha,\ f(\alpha,\phi,s):\neg\phi}$$

$$\frac{(s,t):\neg\neg\alpha}{(t,s):\alpha} \qquad \frac{s:[\neg\alpha]\phi}{(s,t):\alpha \mid t:\phi}$$

- Closure rules:

$$\frac{s:\phi,\ s:\neg\phi}{\bot} \qquad \frac{(s,t):\alpha,\ (s,t):\neg\alpha}{\bot}$$

Rules synthesised from an alternative spec. for $K_{(m)}(\neg)$

- Decomposition rules:

$$\frac{s:\neg\neg\phi}{s:\phi} \qquad \frac{s:\phi\vee\psi}{s:\phi \mid s:\psi} \qquad \frac{s:\neg(\phi\vee\psi)}{s:\neg\phi,\ s:\neg\psi}$$

$$\frac{s:[\alpha]\phi,\ (s,t):\alpha}{(s,t):\neg\alpha \mid t:\phi} \qquad \frac{s:\neg[\alpha]\phi}{(s,f(\alpha,\phi,s)):\alpha,\ f(\alpha,\phi,s):\neg\phi}$$

$$\frac{(s,t):\neg\neg\alpha}{(t,s):\alpha} \qquad \frac{s:[\neg\alpha]\phi}{(s,t):\alpha \mid t:\phi}$$

- Closure rules:

$$\frac{s:\phi,\ s:\neg\phi}{\bot} \qquad \frac{(s,t):\alpha,\ (s,t):\neg\alpha}{\bot}$$

Now, the dagger condition holds for the $[\cdot]$ rule

Termination via blocking

General idea of blocking Use the tableau procedure to find finite models through reusing terms or identifying terms

- Reusing terms

Standard loop-checking mechanisms	Subset or equality blocking Ancestor or anywhere blocking Static or dynamic blocking
Many other techniques	Pairwise blocking Core blocking Pattern-based blocking δ^*-rule

- Identifying terms and equality reasoning

	Unrestricted blocking Sound restricted blocking

Unrestricted blocking mechanism [ISWC07]

- Add the following

(ub) rule $\qquad \dfrac{}{s\approx t \mid s\not\approx t} \quad (s\neq t)$

Ordered rewriting:

$s\approx t$ is a trigger for rewriting $s\longrightarrow t$, if $s\succ t$

- Termination condition

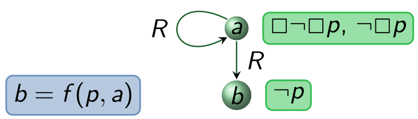
Apply (ub) rule eagerly from some point onwards for all pairs of terms

Example: Unrestricted blocking

(ub) $\dfrac{}{s \approx t \mid s \not\approx t}$ $(s \neq t)$

Example: $\Box\neg\Box p$ is *S4*-satisfiable

$a : \Box\neg\Box p$
$R(a,a)$
$a : \neg\Box p$
$b : \neg p$
$a \approx b$

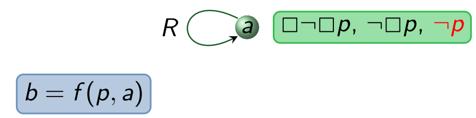


Assume $a \approx b$
Rewrite $b \rightarrow a$

Example: Unrestricted blocking

(ub) $\dfrac{}{s \approx t \mid s \not\approx t}$ $(s \neq t)$

Example: $\Box\neg\Box p$ is *S4*-satisfiable



$a \approx b$

$a : \Box\neg\Box p$
$R(a,a)$
$a : \neg\Box p$
$b : \neg p$
$a \approx b$

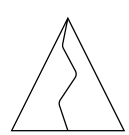
$\downarrow$

$a : \Box\neg\Box p$
$R(a,a)$
$a : \neg\Box p$
$a : \neg p$
$a \approx b$

Soundness, completeness and termination

- Theorem
  Let *Tab* be a sound and complete ground semantic tableau calculus for logic L. Then
  1. $Tab + (ub)$ is a sound and complete.
  2. $Tab + (ub)$ is terminating if L has the finite model property.

- A calculus *Tab* is *terminating* if for any *finite* set N, every closed tableau $Tab(N)$ is finite and every open tableau $Tab(N)$ has a finite open branch.



- Gr. semantic tableaux can decide numerous logics
  Numerous modal, description and hybrid logics, incl. $\mathcal{ALBO}$, $\mathcal{ALBO}^{id}$
  Two-variable fragment of first-order logic

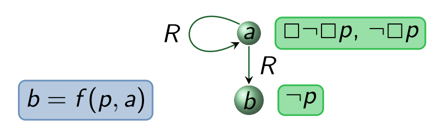
Sound equality ancestor blocking

Restrict application of (ub)

(ub-=) $\dfrac{}{s \approx t \mid s \not\approx t}$ (s is an ancestor of t, $L(s) = L(t)$, $s \neq t$)

where $L(s) = \{\phi \mid s : \phi \text{ in current branch } \mathcal{B}\}$

Example from before: $\Box\neg\Box p$ is *S4*-satisfiable



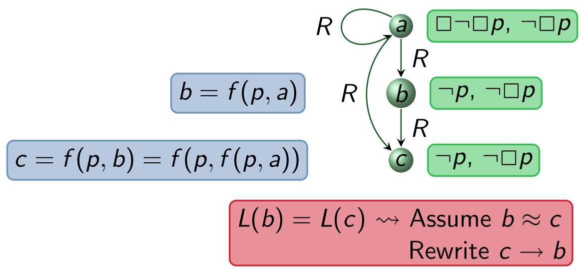
Sound equality ancestor blocking

Restrict application of (ub)

$$(\text{ub-}{=})\quad \frac{}{s \approx t \mid s \not\approx t} \quad (s \text{ is an ancestor of } t,\ L(s) = L(t),\ s \neq t)$$

where $L(s) = \{\phi \mid s : \phi \text{ in current branch } \mathcal{B}\}$

Example from before: $\Box\neg\Box p$ is $S4$-satisfiable



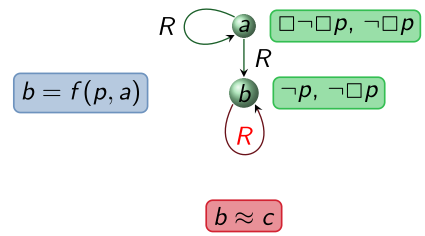
Sound equality ancestor blocking

Restrict application of (ub)

$$(\text{ub-}{=})\quad \frac{}{s \approx t \mid s \not\approx t} \quad (s \text{ is an ancestor of } t,\ L(s) = L(t),\ s \neq t)$$

where $L(s) = \{\phi \mid s : \phi \text{ in current branch } \mathcal{B}\}$

Example from before: $\Box\neg\Box p$ is $S4$-satisfiable



Sound equality ancestor blocking (cont'd)

- Properties of sound equality ancestor blocking
  Produces larger models
  Creates fewer decision points
  Emulates standard equality ancestor blocking
  Sound and logic-independent
- Properties of standard equality ancestor blocking
  Not generally sound
  Soundness can be ensured for certain MLs/DLs with a tree-model property via a certain rule application strategy + ($\Diamond\Box$)-rule
  Gives strong termination results for many MLs/DLs

Sound equality ancestor blocking (cont'd)

- Properties of sound equality ancestor blocking
  Produces larger models
  Creates fewer decision points
  Emulates standard equality ancestor blocking
  Sound and logic-independent
- Properties of standard equality ancestor blocking
  Not generally sound
  Soundness can be ensured for certain MLs/DLs with a tree-model property via a certain rule application strategy + ($\Diamond\Box$)-rule
  Gives strong termination results for many MLs/DLs

Turning calculi into deterministic procedures

Tableau calculi provide *non-deterministic* procedures

At any point there is complete flexibility in choosing:

- which branch to select and expand next
- which rule to apply next
- which formula to select

How to turn the developed tableau calculi into deterministic procedures?

Without loosing soundness, completeness and termination?
That a calculus is sound, complete and terminating does not automatically imply that its implementation is sound, complete and terminating.

Techniques and strategies are needed

Ensuring soundness

Soundness: If N is satisfiable, then in any non-deterministically constructed tableau derivation $Tab(N)$ is open.

Problem: Does this imply that every *deterministic* procedure starting constructs an open tableau for N?

- Problematic are techniques not generally sound and logic-dependent
  E.g., standard blocking techniques for MLs/DLs
- Solution: Apply the rules in a certain order –
  first Boolean rules, then $(\Diamond\Box)$-rule

Ensuring soundness

Soundness: If N is satisfiable, then in any non-deterministically constructed tableau derivation $Tab(N)$ is open.

Problem: Does this imply that every *deterministic* procedure starting constructs an open tableau for N?

- Problematic are techniques not generally sound and logic-dependent
  E.g., standard blocking techniques for MLs/DLs
- Solution: Apply the rules in a certain order –
  first Boolean rules, then $(\Diamond\Box)$-rule

Ensuring completeness

Completeness: If N is unsatisfiable, then any tableau derivation $Tab(N)$ constructed non-deterministically for it is closed

Problem: Does this imply that every *deterministic* procedure starting from N constructs a closed tableau?

- Problematic are rules of universal quantifier extent

$$\frac{s : \Box\phi,\ R(s,t)}{t : \phi}$$ Applicable to the same $s : \Box\phi$ on a branch for each $R(s,t)$ occuring on that branch

- Fairness ensures completeness for deterministic tableau procedure
  A tableau procedure is fair if: When a rule is applicable to a formula then the rule is eventually applied to this formula on every branch on which it occurs (unless the branch is closed and an open, fully expanded branch has already been found)

Solution: Give γ-rules and γ-formulae equal priority

Ensuring completeness

Completeness: If N is unsatisfiable, then any tableau derivation $Tab(N)$ constructed non-deterministically for it is closed

Problem: Does this imply that every *deterministic* procedure starting from N constructs a closed tableau?

- Problematic are rules of universal quantifier extent

$$\frac{s : \Box\phi,\ R(s,t)}{t : \phi}$$ Applicable to the same $s : \Box\phi$ on a branch for each $R(s,t)$ occuring on that branch

- Fairness ensures completeness for deterministic tableau procedure
  A tableau procedure is fair if: When a rule is applicable to a formula then the rule is eventually applied to this formula on every branch on which it occurs (unless the branch is closed and an open, fully expanded branch has already been found)
  Solution: Give γ-rules and γ-formulae equal priority

Ensuring termination

Strong termination: If N is a finite, satisf. set, then every fully expanded or closed branch of any non-determ. constructed tableau $Tab(N)$ is finite

Problem: Does this imply that every *deterministic* procedure starting from N constructs a finite open tableau?

- Yes, for every fair derivation
- Any fair tableau procedure provides a decision procedure; this means depth-first search strategies and arbitrary branch selection strategies may be used.

Ensuring termination

Strong termination: If N is a finite, satisf. set, then every fully expanded or closed branch of any non-determ. constructed tableau $Tab(N)$ is finite

Problem: Does this imply that every *deterministic* procedure starting from N constructs a finite open tableau?

- Yes, for every fair derivation
- Any fair tableau procedure provides a decision procedure; this means depth-first search strategies and arbitrary branch selection strategies may be used.

R. A. Schmidt (Manchester) Semantic Tableaux with Blocking Manchester, 1 July 2012 27 / 32

Ensuring termination

Weak termination: If N is a finite, satisf. set, then any non-determinist. constructed $Tab(N)$ has a finite open, fully expanded branch

Problem: Does this imply that every *deterministic* procedure starting from N constructs a finite open, fully expanded branch?

- Problematic are sound blocking rules

$$\frac{}{s \approx t \mid s \not\approx t}\quad (s \neq t)$$

  Always choosing the right branch at (ub) branching points is like not using blocking at all – if depth-first search is used
  Always choosing the left branch is also not a solution, if depth-first search is used – counterexample for FO^2 due to Reker (2011)

- Fairness of branch selection ensures termination for deterministic proc.
  Solution: Use depth-first iterative deepening or depth-first up-to-maximal-bound or breadth-first search

Ensuring termination

Weak termination: If N is a finite, satisf. set, then any non-determinist. constructed $Tab(N)$ has a finite open, fully expanded branch

Problem: Does this imply that every *deterministic* procedure starting from N constructs a finite open, fully expanded branch?

- Problematic are sound blocking rules

$$\frac{}{s \approx t \mid s \not\approx t} \quad (s \neq t)$$

Always choosing the right branch at (ub) branching points is like not using blocking at all – if depth-first search is used

Always choosing the left branch is also not a solution, if depth-first search is used – counterexample for FO^2 due to Reker (2011)

- Fairness of branch selection ensures termination for deterministic proc.
  Solution: Use depth-first iterative deepening or depth-first up-to-maximal-bound or breadth-first search

Ensuring termination

Weak termination: If N is a finite, satisf. set, then any non-determinist. constructed $Tab(N)$ has a finite open, fully expanded branch

Problem: Does this imply that every *deterministic* procedure starting from N constructs a finite open, fully expanded branch?

- Problematic are sound blocking rules

$$\frac{}{s \approx t \mid s \not\approx t} \quad (s \neq t)$$

Always choosing the right branch at (ub) branching points is like not using blocking at all – if depth-first search is used

Always choosing the left branch is also not a solution, if depth-first search is used – counterexample for FO^2 due to Reker (2011)

- Fairness of branch selection ensures termination for deterministic proc.
  Solution: Use depth-first iterative deepening or depth-first up-to-maximal-bound or breadth-first search

R. A. Schmidt (Manchester) Semantic Tableaux with Blocking Manchester, 1 July 2012 28 / 32

Ensuring termination

Weak termination: If N is a finite, satisf. set, then any non-determinist. constructed $Tab(N)$ has a finite open, fully expanded branch

Problem: Does this imply that every *deterministic* procedure starting from N constructs a finite open, fully expanded branch?

- Problematic are sound blocking rules

$$\frac{}{s \approx t \mid s \not\approx t} \quad (s \neq t)$$

Always choosing the right branch at (ub) branching points is like not using blocking at all – if depth-first search is used

Always choosing the left branch is also not a solution, if depth-first search is used – counterexample for FO^2 due to Reker (2011)

- Fairness of branch selection ensures termination for deterministic proc.
  Solution: Use depth-first iterative deepening or depth-first up-to-maximal-bound or breadth-first search

R. A. Schmidt (Manchester) Semantic Tableaux with Blocking Manchester, 1 July 2012 28 / 32

Concluding remarks and outlook

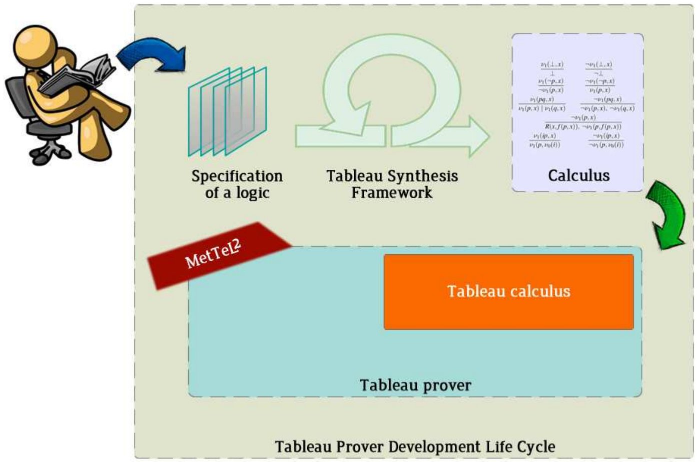
The possibilities in designing tableau calculi/provers are endless

Focus in this talk

- Synthesis
- Refinement
- Blocking
- Determinisation & search strategies

Much remains to be done

Prover generation: `http://www.mettel-prover.org/`



Picture by Mohammad Khodadadi

Thanks

- Dmitry Tishkovsky
- Mohammad Khodadadi
- Hilverd Reker
- Fabio Papacchini
- Users:
  Michal Zawidzki, Stefan Minica, Clare Dixon, Boris Konev, ...

Some references

- Tableau synthesis
  Schmidt, R. A. (2009), A New Methodology for Developing Deduction Methods. *Ann. Math. and Artificial Intelligence* **55** (1–2), 155–187.
  Schmidt, R. A. & Tishkovsky, D. (2011), Automated Synthesis of Tableau Calculi. *Logical Methods in Comput. Sci.* **7** (2), 1–32. Short version in *Proc. TABLEAUX'09*.
  Tishkovsky, D., Schmidt, R. A. & Khodadadi, M. (2012), MetTeL2: Towards a Tableau Prover Generation Platform. In *Proc. PAAR'12*.
- Applications
  Babenyshev, S., Rybakov, V., Schmidt, R. A., & Tishkovsky, D. (2010), A Tableau Method for Checking Rule Admissibility in S4. *ENTCS* **262**, 17–32.
  Minica, S., Khodadadi, M., Tishkovsky, D. & Schmidt, R. A. (2012), Synthesising and Implementing Tableau Calculi for Interrogative Epistemic Logics. In Proc. PAAR'12.
- Unrestricted blocking
  Schmidt, R. A., & Tishkovsky, D. (2007), Using Tableau to Decide Expressive Description Logics with Role Negation. In *Proc. ISWC'07*.
  Schmidt, R. A., & Tishkovsky, D. (2008), A General Tableau Method for Deciding Description Logics, Modal Logics and Related First-Order Fragments. In *Proc. IJCAR'08*.