The Model Evolution Calculus with Built-in Theories

Peter Baumgartner

MPI Informatik, Saarbrücken

www.mpi-sb.mpg.de/~baumgart/

Problem

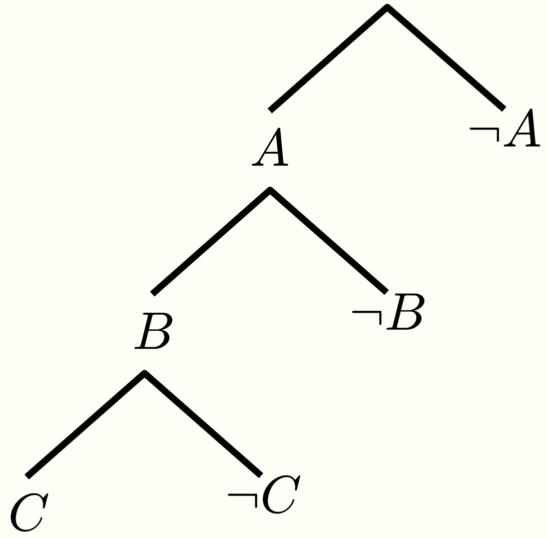
- The Model Evolution Calculus is a sound and refutationally complete calculus for first-order clause logic
- **Can we extend it with built-in theory handling?**
  That is, „plug in“ an (efficient) reasoner for a special domain
- Examples for interesting theories
  - Equality
  - Real arithmetic
  - Theories axiomatized by logic programs
- Can existing theory reasoners be plugged in (to Darwin)?
  - Equality: Waldmeister
  - Real arithmetic: quantifier elimination
  - Logic programs: logic program interpreter

Model Evolution – Idea (1)

DPLL: Davis-Putnam-Logemann-Loveland Procedure (1960-63)
Basis of some of the SAT solvers (Chaff, …)

Input: Propositional clause set
Output: Model or „unsatisfiable"

Algorithm components:
- Simplification
- Split
- Backtracking



$$\begin{tikzcd} \{A, B\} \overset{?}{\models} \{\neg A \vee \neg B \vee C \vee D, \ldots\} \arrow[d, "\text{No, split on } C"] \\ \{A, B, C\} \models \{\neg A \vee \neg B \vee C \vee D, \ldots\} \end{tikzcd}$$

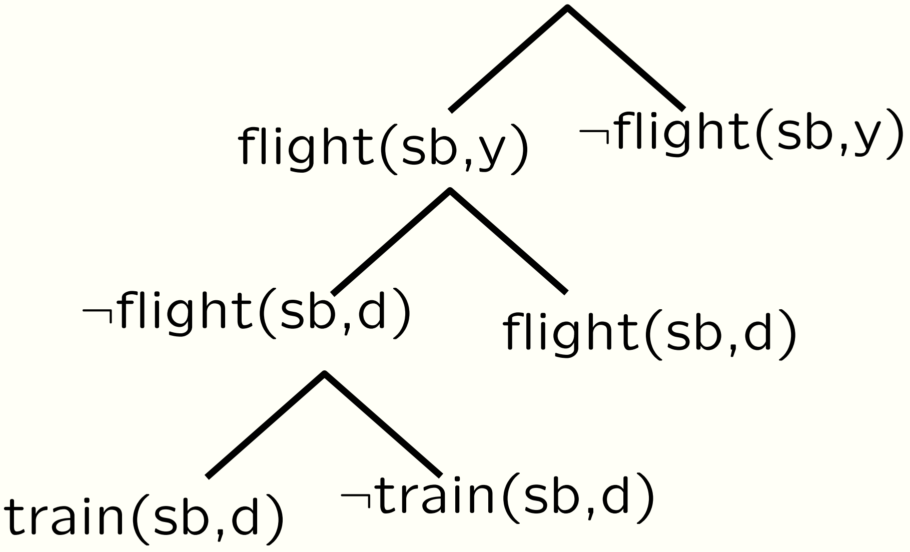
Model Evolution – Idea (2)

≈**First Order DPLL** [Joint Work with Cesare Tinelli]

Input: First-order clause set
Output: Model or „unsatisfiable"
if termination

Procedure components:

- Simplification
- Split
- Backtracking



flight(sb,y)

¬flight(sb,y)

¬flight(sb,d)

flight(sb,d)

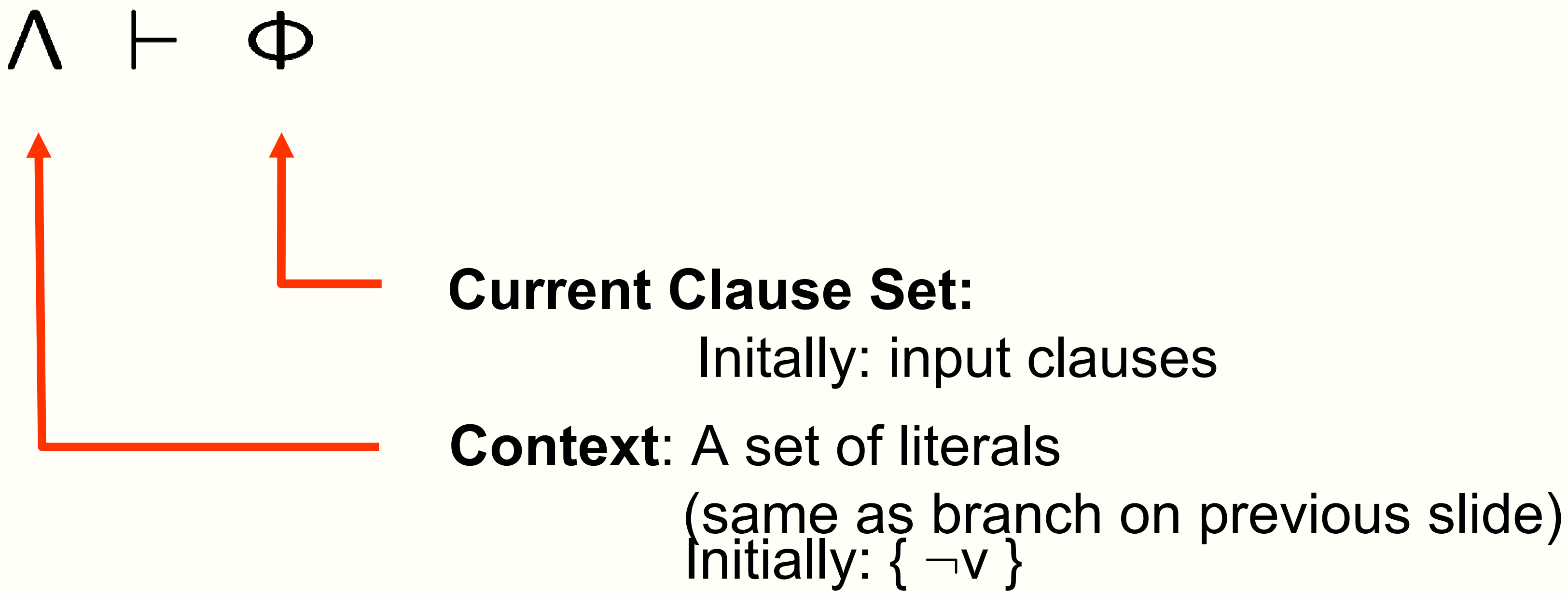
train(sb,d)

¬train(sb,d)

$$\begin{tikzcd}
\{\mathrm{flight}(sb,y), \neg\mathrm{flight}(sb,d)\} \overset{?}{\models} \{\mathrm{flight}(x,y) \vee \mathrm{train}(x,y), \ldots\} \arrow[d, "\text{No, split on train(sb,d)}"] \\
\{\mathrm{flight}(sb,y), \neg\mathrm{flight}(sb,d), \mathrm{train}(sb,d)\} \models \{\mathrm{flight}(x,y) \vee \mathrm{train}(x,y), \ldots\}
\end{tikzcd}$$

Calculus

- **Sequent Style Calculus**



$$\Lambda \vdash \Phi$$

Current Clause Set:

Initally: input clauses

Context: A set of literals

(same as branch on previous slide)

Initially: { ¬v }

- **Simplified Calculus (for the purpose of talk)**
  - No simplification inference rules to modify Φ
  - No simplification inference rules to modify Λ
  - No „universal“ variables, only „parametric“ ones

Derivation Rules (1)

Split $$\frac{\Lambda \vdash \Phi, C \vee L}{\Lambda, L\sigma \vdash \Phi, C \vee L \qquad\qquad \Lambda, \overline{L}\sigma \vdash \Phi, C \vee L}$$

if

(1) σ is a context unifier of $C \vee L$ against Λ

(2) neither $L\sigma$ nor $\neg\, L\sigma$ is contraditory with Λ

- σ is a **context unifier**: σ is a most general simultaneous unifier of the clause literals and context literals with opposite sign (pairwise)
- $L\sigma$ is **contradictory** with Λ : Λ contains a variant of $\neg L\sigma$

Context: $\underline{\mathrm{P(u,u)}}$ $\underline{\mathrm{Q(v,b)}}$

Clause: ¬P(x,y) ∨ ¬Q(a,z)

σ = { x→u, y→u, v→a, z→b }

Clause σ: $\underline{\neg\mathrm{P(x,x)}}$ ∨ $\underline{\neg\mathrm{Q(a,b)}}$

¬Q(a,b) is admissible for Split

contradictory not contradictory

Derivation Rules (2)

Close $$\frac{\Lambda \vdash \Phi, C}{\Lambda \vdash \bot}$$

if (1) $\Phi \neq \emptyset$ or $C \neq \bot$

(2) there is a context unifier σ of C against Λ such that each literal of $C\sigma$ is contradictory with Λ

- σ is a **context unifier**: σ is a most general simultaneous unifier of the clause literals and context literals with opposite sign (pairwise)
- $L\sigma$ is **contradictory** with Λ : Λ contains a variant of $\neg L\sigma$

Context: $\underline{P(u,u)}$ $\underline{Q(a,b)}$

Clause: $\neg P(x,y) \vee \neg Q(a,z)$ $\sigma = \{ x \rightarrow u,\ y \rightarrow u,\ z \rightarrow b \}$

Clause σ: $\underline{\neg P(x,x)} \vee \underline{\neg Q(a,b)}$ Close is applicable

contradictory contradictory

Model Evolution – Further Ingredients

- **Derivation**
  - Start with sequent $\neg v \vdash$ „Input Clause Set“
  - Apply Split and Close derivation rules (gives tree over sequents)
- **Refutation**: Every branch ends in sequent of the form $\Lambda \vdash \bot$
- **Fairness**
  - Consider a derivation with limit context $\Lambda_\infty = \cup_{i>0} \Lambda_i$
  - Close is not applicable to any Λ_i
  - Roughly: if some ground instance $C\gamma$ of an input clause is falsified by Λ_i then there is a $j>i$ such that Λ_j satisfies $C\gamma$ (this can always be achieved by applying the split rule)
- **Completeness**
  - Assume a fair derivation with limit context
  - Show that Λ_∞ constitutes a model for the input clause set

Theories – Basic Definitions

- A **Theory** $\mathcal{T}$ is a consistent set of sentences
- Consider here **universal** theories

  (no existential quantifier in prenex normal form)
- **Def:** Clause set Φ is $\mathcal{T}$-unsatisfiable iff

  $\Phi \cup \mathcal{T}$ is unsatisfiable
- **Def:** Let $\mathcal{K}$ be a set of literals and L be a literal

  $\mathcal{K} \models_{\mathcal{T}} L$

  iff $\mathcal{K} \cup \mathcal{T} \models L$

  iff for every structure $\mathcal{A}$ and every valuation v:

  $\mathcal{A},v \models \mathcal{K} \cup \mathcal{T}$ implies $\mathcal{A},v \models L$

Examples

$\{ \mathsf{P(u,a)},\ \mathsf{u{=}f(u)},\ \mathsf{a{=}f(a)} \} \models_{\mathsf{E}} \mathsf{P(f(u),f(a))}$ holds

$\{ \mathsf{P(u,a)},\ \mathsf{u{=}f(u)},\ \mathsf{v{=}f(v)} \} \models_{\mathsf{E}} \mathsf{P(f(u),f(a))}$ does not hold

ME($\mathcal{T}$) – Derivation Rules (1)

$\mathcal{T}$-Split
$$\frac{\Lambda \vdash \Phi, C \vee L}{\Lambda, K\sigma \vdash \Phi, C \vee L \qquad\qquad \Lambda, \overline{K}\sigma \vdash \Phi, C \vee L}$$

if

(1) σ is a $\mathcal{T}$-context unifier of $C \vee L$ against Λ with key set $\mathcal{K} \cup \{ L \}$

(2) $K \in \neg\mathcal{K}$

(3) neither $K\sigma$ nor $\neg K\sigma$ is $\mathcal{T}$-contradictory with Λ

σ is a $\mathcal{T}$-**context unifier** of clause $L_1 \vee \ldots \vee L_\nu$

iff there are sets $\mathcal{K}_1,\ldots,\mathcal{K}_n$ of variants of literals from Λ s.th. $\mathcal{K}_i\sigma \models_{\mathcal{T}} \neg L_i\sigma$

Each set $\mathcal{K}_i \cup \{ L_i \}$ is called a **key set**

Context: P(a,b) u=f(u)

Clause: ¬P(f(a),f(x))

Key set:

{ P(a,b), u=f(u), v=f(v), ¬P(f(a),f(x)) }

σ = { u→a, v→b, x→b }

$\mathcal{T}$-Split on ¬(a=f(a))

ME($\mathcal{T}$) – Derivation Rules (1)

$\mathcal{T}$-Split $$\frac{\Lambda \vdash \Phi, C \vee L}{\Lambda, K\sigma \vdash \Phi, C \vee L \qquad\qquad \Lambda, \overline{K}\sigma \vdash \Phi, C \vee L}$$

if

(1) σ is a $\mathcal{T}$-context unifier of $C \vee L$ against Λ with key set $\mathcal{K} \cup \{ L \}$

(2) $K \in \neg\mathcal{K}$

(3) neither $K\sigma$ nor $\neg K\sigma$ is $\mathcal{T}$-contraditory with Λ

$K\sigma$ is $\mathcal{T}$–**contradictory with** Λ

iff there is a set $\mathcal{K}$ of variants of literals from Λ s.th. $\mathcal{K} \models_{\mathcal{T}} \neg K_i\sigma$

Example for $\mathcal{T}$–contradictory:

Context: P(u,v) u=f(u)

$K\sigma$: ¬P(f(u),f(v))

$\mathcal{K}$ = { P(u,v), u=f(u), v=f(v) }

ME($\mathcal{T}$) – Derivation Rules (2)

$\mathcal{T}$-Repair $$\frac{\Lambda \vdash \Phi, C \vee L}{\Lambda, K\sigma \vdash \Phi, C \vee L}$$

if

(1) σ is a $\mathcal{T}$-context unifier of $C \vee L$ against Λ with key set $\mathcal{K} \cup \{ L \}$

(2) $K \in \neg\mathcal{K}$

(3) $K\sigma$ is not $\mathcal{T}$–contradictory with Λ, but
$\neg K\sigma$ is $\mathcal{T}$-contraditory with Λ

(4) Λ does not contain a variant of $K\sigma$

- $\mathcal{T}$–Repair is the one-armed, disjoint variant of $\mathcal{T}$–Split
- $\mathcal{T}$–Repair is not applicable if $\mathcal{T}$ is the „empty“ theory

Context: ¬(f(a)=b)   a=b   P(a)   f(u)=u

Clause: ¬P(f(a))

$\mathcal{T}$-Repair with ¬(a=f(a))

ME($\mathcal{T}$) – Derivation Rules (3)

$\mathcal{T}$-Close $$\frac{\Lambda \vdash \Phi, C}{\Lambda \vdash \bot}$$

if (1) $\Phi \neq \emptyset$ or $C \neq \bot$

(2) there is a $\mathcal{T}$-context unifier σ of C against Λ such that each literal of $C\sigma$ is $\mathcal{T}$-contradictory with Λ

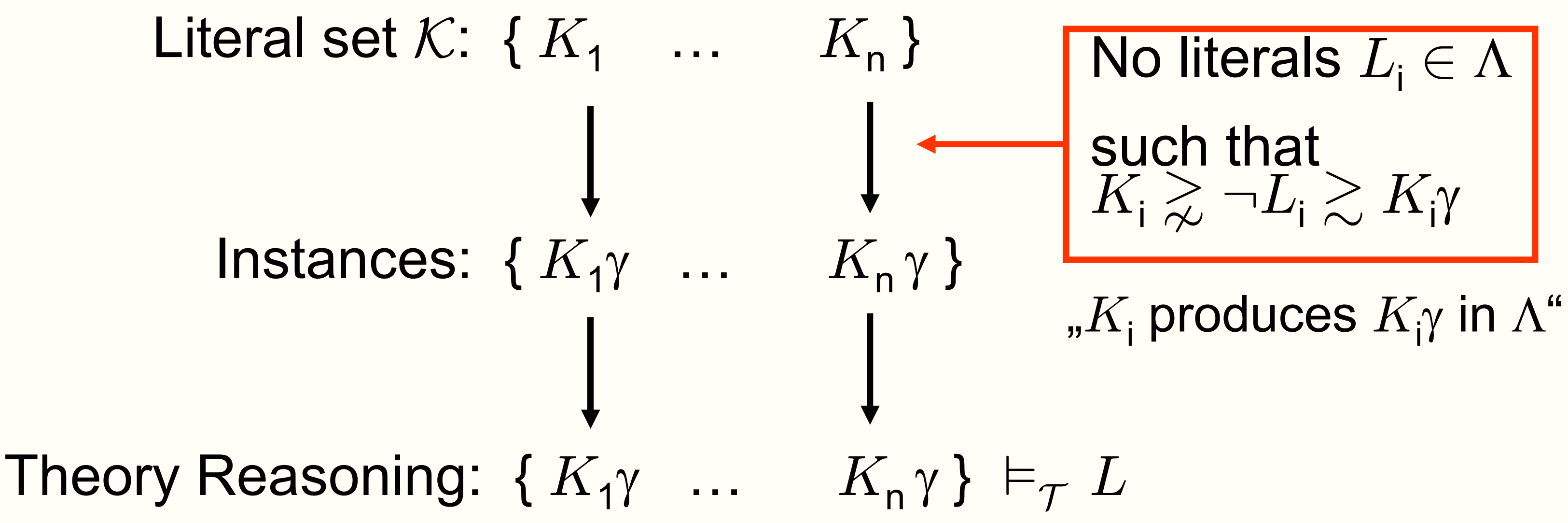
Note: Condition (2) must be decidable!

Interpretation Associated to a Context

- Crucial to understand the working of the calculus
- Basis of the completeness proof
- Basis of feasible instantiation with theory reasoners
  E.g. Waldmeister for the theory of equality

Interpretation Associated to a Context

Literal set $\mathcal{K}\mathcal{T}$–produces a literal L in Λ



Literal set $\mathcal{K}$: $\{ K_1 \quad \dots \quad K_n \}$

Instances: $\{ K_1\gamma \quad \dots \quad K_n\gamma \}$

Theory Reasoning: $\{ K_1\gamma \quad \dots \quad K_n\gamma \} \models_{\mathcal{T}} L$

No literals $L_i \in \Lambda$ such that $K_i \not\gtrsim \neg L_i \gtrsim K_i\gamma$

„K_i produces $K_i\gamma$ in Λ“

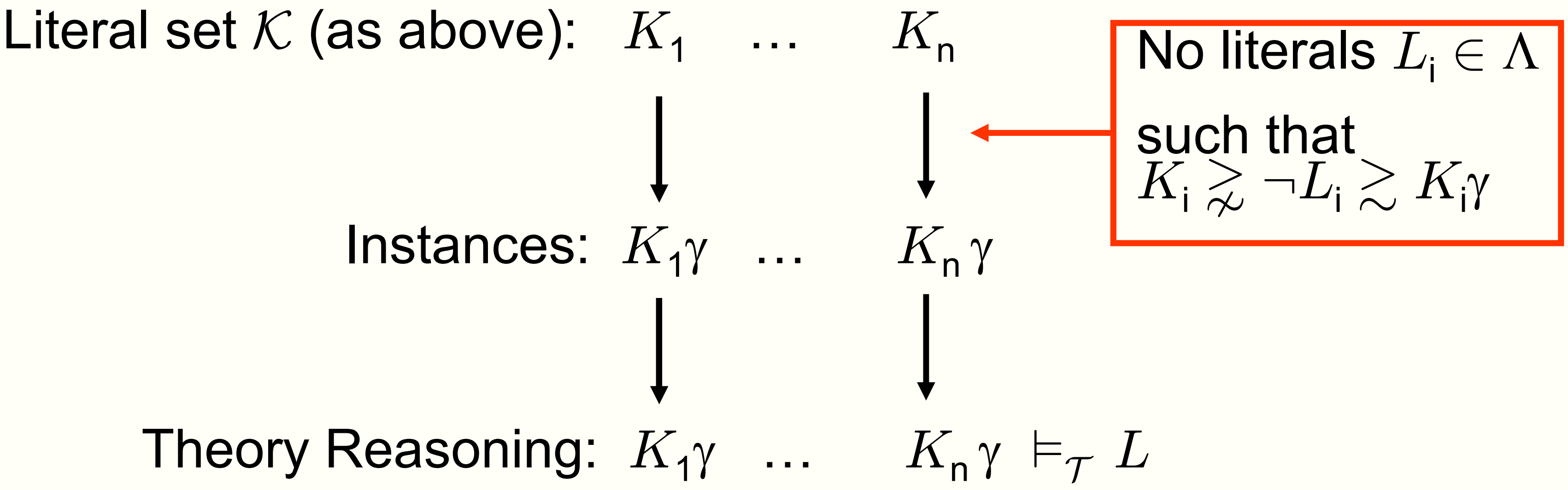
Interpretation Associated to Λ

A ground atom A **is assigned true in Λ via $\mathcal{K}$**

iff some set $\mathcal{K}$ of variants of literals from Λ $\mathcal{T}$–produces A

Interpretation Associated to a Context

Context Λ $\mathcal{T}$–produces a literal L



Literal set $\mathcal{K}$ (as above): K_1 … K_n

Instances: $K_1\gamma$ … $K_n\gamma$

Theory Reasoning: $K_1\gamma$ … $K_n\gamma \models_{\mathcal{T}} L$

No literals $L_i \in \Lambda$ such that $K_i \not\gtrsim \neg L_i \gtrsim K_i\gamma$

Examples

{ P(a), f(x)=x, ¬(f(a)=a) } does not E-produce P(f(a))

=> P(f(a)) is assigned false in associated E-interpretation

{ P(a), f(x)=x, ¬P(f(a)) } E-produces P(f(a)) and ¬P(f(a))

=> P(f(a)) is assigned true in associated E-interpretation

ME($\mathcal{T}$) Calculus – Theory Reasoner $R_\mathcal{T}$

- A lifting lemma cannot be proven „once and for all“, replace it by **admissibility condition** of theory reasoner $R_\mathcal{T}$
- **Theory reasoner** $R_\mathcal{T}$
  - **Input**: a context Λ and a clause C = L_1 Ç … Ç L_n
  - **Output**: a n+1 -tuple ($\mathcal{K}_1$,…,$\mathcal{K}_n$,σ) or undefined where $\mathcal{K}_i$ is a set of variants of literals from Λ and σ is a substitution
- $R_\mathcal{T}$ is **sound** iff $\mathcal{K}_i\sigma \models_\mathcal{T} \neg L_i\sigma$ (i.e. σ is a $\mathcal{T}$–context unifier)
- $R_\mathcal{T}$ is **complete** iff the following holds:

  For every ground instance $C\gamma$ and all sets $\mathcal{K}_1$,...,$\mathcal{K}_n$ (as above):

  If $C\gamma$ is assigned false in Λ via $\mathcal{K}_1$,...,$\mathcal{K}_n$

  then $R_T(\Lambda,C)$ = ($\mathcal{K}_1$,…,$\mathcal{K}_n$,σ) for some substitution $\sigma \gtrsim \gamma$
- $R_\mathcal{T}$ is **admissible** iff it is sound and complete

Consequences and Properties

- Associated interpretation should be **total:** easy, context contains ¬v
- Associated interpretation should be a $\mathcal{T}$**–interpretation**

  Need further restrictions on allowed theories to guarantee this:

  - Non-negative theories: not $\models \exists(A_1 \wedge \cdots \wedge A_n)$
  - $\mathcal{T}$ = { ¬A } is not allowed
  - Theory must be ground convex:
    $\models_{\mathcal{T}} B \rightarrow A_1 \vee \ldots \vee A_n$ implies $\models_{\mathcal{T}} B \rightarrow A_i$ for some i

    (B conjunction of ground atoms, A ground atom)
    $\mathcal{T}$ = { A ∨B } is not allowed

- **Property**
  If limit context Λ_∞ assigns false to a (ground) clause $C\gamma$ via $\mathcal{K}_1,\ldots,\mathcal{K}_n$

  then
  there is an i such that for all j > i Λ_j assigns false to $C\gamma$ via $\mathcal{K}_1,\ldots,\mathcal{K}_n$

- **Completeness**
  Fairness + admissible theory reasoner will detect this situation eventually and invalidate it

Equality and Waldmeister

- **Problem**

  Waldmeister is a theorem prover for unit clauses
  $\{ s_1 = t_1, \ldots, s_n = t_n, \ \neg(s = t) \}$

  How to match it to **contexts** and **arbitrary clauses**?
  $\neg(s_1 = t_1) \ \vee \ldots \vee \ \neg(s_m = t_m) \ \vee \ s_{m+1} = t_{m+1} \vee \ldots \vee \ s_n = t_n$

- **Context Problem**

  $\Lambda = \{ a = f(a), \ P(u), \ \neg P(a), \ \neg P(f(a)), \ \neg P(f(f(a))) \}$

  Clause $\neg P(a)$

  Waldmeister has to discover instances $P(f(f(f(a)))), \ldots$

  Solution (?)

  Convert context to equivalent set of atoms

  E.g. for signature $\{a/0, \ b/0, \ f/1\}$ obtain

  $\Lambda = \{ a = f(a), \ P(b), \ P(f(b)), \ P(f(f(b))), \ P(f(f(f(x)))) \}$

  Resulting set can be infinite in case of non-linear literals!

Equality and Waldmeister

- **Problem**

  Waldmeister is a theorem prover for unit clauses
  $\{ s_1 = t_1, \ldots, s_n = t_n,\ \neg(s = t) \}$

  How to match it to **contexts** and **arbitrary clauses**
  $\neg(s_1 = t_1) \vee \ldots \vee \neg(s_m = t_m) \vee s_{m+1} = t_{m+1} \vee \ldots \vee s_n = t_n$

- **Arbitrary Clauses Problem**

  From definition of associated interpretation it follows:

  Context Λ falsifies a positive literal A
  iff some negative literal $\neg B \in \Lambda$ produces $\neg A$ in Λ

  Consequently:

  Can resolve away positive clause literals against context literals
  Leaves only rest clause $(\neg(s_1 = t_1) \vee \ldots \vee \neg(s_m = t_m))\sigma$

Equality and Waldmeister

- **Problem**
  Waldmeister is a theorem prover for unit clauses
  $\{ s_1{=}t_1,\ldots,s_n{=}t_n,\ \neg(s{=}t) \}$

  How to match it to **contexts** and **arbitrary clauses**
  $\neg(s_1{=}t_1) \vee\ldots\vee \neg(s_m{=}t_m) \vee s_{m+1}{=}t_{m+1} \vee\ldots\vee s_n{=}t_n$

- **Arbitrary Clauses Problem**
  How to treat rest clause $(\neg(s_1{=}t_1) \vee\ldots\vee \neg(s_m{=}t_m))\sigma$?

  Solution

  Code it as a negative unit clause (due to Thomas Hillenbrand):
  $\neg(\mathrm{clause}(s_1,t_1,\ldots,s_m,t_m) = \mathrm{true})$

  $\mathrm{clause}(x_1,x_1,\ldots,x_m,x_m) = \mathrm{true}$

  Can easily query Waldmeister with many clauses simultaneously

- **Thus have transformation for Waldmeister now**

  But Waldmeister still has to be modifed to compute „all“ solutions!

Conclusion

- Presented simplified calculus, without universal variables e.g. $\forall$ x P(x,u)
  - Universal variables crucial for performance
    - calculus instantiates to postive hyper-resolution for Horn case
    - One call to Waldmeister for unit theories
  - Should work out without greater difficulties
- Is this all feasible?
- Difference to Ganzinger/Korovin Calculus wrt.\ theory reasoning
  - Works for arbitrary universal non-negative convex theories
  - Does not need a term ordering
    But using term orderings might be advantageous…