Decision Procedures in Verification

First-Order Logic (4)

12.12.2016

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Exam

Until now:

General Resolution

Soundness, refutational completeness

Refinements: Ordered resolution with selection

Consequences:

Herbrand's theorem

The Theorem of Löwenheim-Skolem

Compactness of first-order logic

Craig Interpolation

Resolution Calculus $Res_S^{\succ}$

Let $\succ$ be a total and well-founded ordering on ground atoms and S a selection function.

Ordered resolution with selection

$$\frac{C \vee A \qquad \neg B \vee D}{(C \vee D)\sigma} \qquad \text{[ordered resolution with selection]}$$

if $\sigma = \mathrm{mgu}(A, B)$ and

(i) $A\sigma$ strictly maximal wrt. $C\sigma$;

(ii) nothing is selected in C by S;

(iii) either $\neg B$ is selected,
or else nothing is selected in $\neg B \vee D$ and $\neg B\sigma$ is maximal in $D\sigma$.

Ordered factoring

$$\frac{C \vee A \vee B}{(C \vee A)\sigma} \qquad \text{[ordered factoring]}$$

if $\sigma = \mathrm{mgu}(A, B)$ and $A\sigma$ is maximal in $C\sigma$ and nothing is selected in C.

Craig Interpolation

Theorem: $Res_S^{\succ}$ is sound and refutationally complete.

A theoretical application of ordered resolution is Craig- Interpolation:

Theorem (Craig 57)

Let F and G be two propositional formulas such that $F \models G$.

Then there exists a formula H (called the interpolant for $F \models G$), such that H contains only propostional variables occurring both in F and in G, and such that $F \models H$ and $H \models G$.

Craig Interpolation

Proof:

Translate F and $\neg G$ into CNF.

Let N and M, resp., denote the resulting clause set.

Choose an atom ordering $\succ$ for which the propositional variables that occur in F but not in G are maximal.

Saturate N into N^* wrt. $\mathrm{Res}_S^{\succ}$ with an empty selection function S.

Then saturate $N^* \cup M$ wrt. $\mathrm{Res}\succ_S$ to derive $\bot$.

As N^* is already saturated, due to the ordering restrictions only inferences need to be considered where premises, if they are from N^*, only contain symbols that also occur in G.

The conjunction of these premises is an interpolant H.

The theorem also holds for first-order formulas. For universal formulas the above proof can be easily extended. In the general case, a proof based on resolution technology is more complicated because of Skolemization.

Applications of Craig Interpolation

Modular databases

Given: Two databases (different but possibly overlapping languages)

Task: Is the union of the two databases consistent? If not: locate error

Applications of Craig Interpolation

Modular databases

Given: Two databases (different but possibly overlapping languages)

Logical modeling: $F_1 \wedge F_2$

Task: Is the union of the two databases consistent? If not: locate error

$F_1 \wedge F_2 \models \perp$

Applications of Craig Interpolation

Modular databases

Given: Two databases (different but possibly overlapping languages)

Logical modeling: $F_1 \wedge F_2$

Task: Is the union of the two databases consistent? If not: locate error

$F_1 \wedge F_2 \models \perp$

$F_1 \models \neg F_2$ (assume we are in prop. logic)

Applications of Craig Interpolation

Modular databases

Given: Two databases (different but possibly overlapping languages)

Logical modeling: $F_1 \wedge F_2$

Task: Is the union of the two databases consistent? If not: locate error

$F_1 \wedge F_2 \models \perp$

$F_1 \models \neg F_2$ (assume we are in prop. logic)

Craig Interpolation (propositional case)

There exists I containing only propositional variables occurring in F_1 and F_2 such that:

$F_1 \models I$ and $I \models \neg F_2$

Applications of Craig Interpolation

Reasoning in combinations of theories

Given: Two theories (different but possibly overlapping languages)
s.t. decision procedures for component theories for certain fragments exist

Task: Reason in the combination of the two theories

Question: Which information needs to be exchanged between provers?

Answer: Craig Interpolation

The case of two disjoint theories will be discussed later in this lecture

Applications of Craig Interpolation

Verification (programs or hardware)

Model programs as transition systems.

- Sets of states expressed as formulae
- Transitions expressed as formulae T

Question:

Can a state in a certain set of states E (error)
be reached from some state in a set I (initial) in k steps?

$\phi_I \wedge T_1 \wedge T_2 \wedge \cdots \wedge T_k \wedge \phi_E$

Applications of Craig Interpolation

Verification (programs or hardware)

Model programs as transition systems.

- Sets of states expressed as formulae
- Transitions expressed as formulae T

Question:
Can a state in a certain set of states E (error)
be reached from some state in a set I (initial) in k steps?

$\underbrace{(\phi_I \wedge T_1)}_{F_1} \wedge \underbrace{(T_2 \wedge \cdots \wedge T_k \wedge \phi_E)}_{F_2}$ Not reachable: $F_1 \wedge F_2 \models \perp$

Applications of Craig Interpolation

Verification (programs or hardware)

Model programs as transition systems.

- Sets of states expressed as formulae
- Transitions expressed as formulae T

Question:

Can a state in a certain set of states E (error)
be reached from some state in a set I (initial) in k steps?

$$\underbrace{(\phi_I \wedge T_1)}_{F_1} \wedge \underbrace{(T_2 \wedge \cdots \wedge T_k \wedge \phi_E)}_{F_2}$$

Not reachable: $F_1 \wedge F_2 \models \perp$

Interpolant: I overapproximates the set of successors of ϕ_I.

Goal

Goal: Make resolution efficient

Identify clauses which are not needed and can be discarded

Redundancy

So far: local restrictions of the resolution inference rules using orderings and selection functions.

Is it also possible to delete clauses altogether?
Under which circumstances are clauses unnecessary?
(Conjecture: e. g., if they are tautologies or if they are subsumed by other clauses.)

Intuition: If a clause is guaranteed to be neither a minimal counterexample nor productive, then we do not need it.

Recall

Construction of I for the extended clause set:

	clauses C	I_C	Δ_C	Remarks
1	$\neg P_0$	$\emptyset$	$\emptyset$	
2	$P_0 \vee P_1$	$\emptyset$	$\{P_1\}$	
3	$P_1 \vee P_2$	$\{P_1\}$	$\emptyset$	
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	
9	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	$\{P_3\}$	
8	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_3 \vee P_0$	$\{P_1, P_2, P_3\}$	$\emptyset$	true in $\mathcal{A}_C$
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$	$\{P_1, P_2, P_3\}$	$\emptyset$	
6	$\neg P_1 \vee \neg P_4 \vee P_3$	$\{P_1, P_2, P_3\}$	$\emptyset$	true in $\mathcal{A}_C$
7	$\neg P_3 \vee P_5$	$\{P_1, P_2, P_3\}$	$\{P_5\}$	

The resulting $I = \{P_1, P_2, P_3, P_5\}$ is a model of the clause set.

A Formal Notion of Redundancy

Let N be a set of ground clauses and C a ground clause (not necessarily in N). C is called redundant w. r. t. N, if there exist $C_1, \ldots, C_n \in N$, $n \geq 0$, such that $C_i \prec C$ and $C_1, \ldots, C_n \models C$.

Redundancy for general clauses:
C is called redundant w. r. t. N, if all ground instances $C\sigma$ of C are redundant w. r. t. $G_\Sigma(N)$.

Intuition: Redundant clauses are neither minimal counterexamples nor productive.

Note: The same ordering $\succ$ is used for ordering restrictions and for redundancy (and for the completeness proof).

Examples of Redundancy

Proposition 2.40:

- C tautology (i.e., $\models C$) $\;\Rightarrow\;$ C redundant w. r. t. any set N.
- $C\sigma \subset D \;\Rightarrow\; D$ redundant w. r. t. $N \cup \{C\}$
- $C\sigma \subseteq D \;\Rightarrow\; D \vee \overline{L}\sigma$ redundant w. r. t. $N \cup \{C \vee L,\ D\}$

(Under certain conditions one may also use non-strict subsumption, but this requires a slightly more complicated definition of redundancy.)

Saturation up to Redundancy

N is called saturated up to redundancy (wrt. $Res_S^{\succ}$)

$$:\Leftrightarrow Res_S^{\succ}(N \setminus Red(N)) \subseteq N \cup Red(N)$$

Theorem 2.41:

Let N be saturated up to redundancy. Then

$$N \models \bot \Leftrightarrow \bot \in N$$

Saturation up to Redundancy

Proof (Sketch):

(i) Ground case:

- consider the construction of the candidate model $I_N^{\succ}$ for $Res_S^{\succ}$
- redundant clauses are not productive
- redundant clauses in N are not minimal counterexamples for $I_N^{\succ}$

The premises of "essential" inferences are either minimal counterexamples or productive.

(ii) Lifting: no additional problems over the proof of Theorem 2.39.

Monotonicity Properties of Redundancy

Theorem 2.42:

(i) $N \subseteq M \Rightarrow \mathit{Red}(N) \subseteq \mathit{Red}(M)$

(ii) $M \subseteq \mathit{Red}(N) \Rightarrow \mathit{Red}(N) \subseteq \mathit{Red}(N \setminus M)$

Proof:

(i) Let $C \in \mathit{Red}(N)$. Then there exist $C_1, \ldots, C_n \in N, n \geq 0$ such that $C_i \prec C$ for all $i = 1, \ldots, n$ and $C_1, \ldots, C_n \models C$.

We assumed that $N \subseteq M$, so we know that $C_1, \ldots, C_n \in M$. Thus: there exist $C_1, \ldots, C_n \in M, n \geq 0$ such that $C_i \prec C$ for all $i = 1, \ldots, n$ and $C_1, \ldots, C_n \models C$. Therefore, $C \in \mathit{Red}(M)$.

Monotonicity Properties of Redundancy

Theorem 2.42:

(i) $N \subseteq M \Rightarrow \mathit{Red}(N) \subseteq \mathit{Red}(M)$

(ii) $M \subseteq \mathit{Red}(N) \Rightarrow \mathit{Red}(N) \subseteq \mathit{Red}(N \setminus M)$

Proof (Idea):

(ii) Let $C \in \mathit{Red}(N)$. Then there exist $C_1, \dots, C_n \in N$, $n \geq 0$ such that $C_i \prec C$ for all $i = 1, \dots, n$ and $C_1, \dots, C_n \models C$.

Case 1: For all i, $C_i \notin M$. Then $C \in \mathit{Red}(N \setminus M)$.

Case 2: For some i, $C_i \in M \subseteq \mathit{Red}(N)$. Then for every such index i there exist $C^i_1, \dots, C^i_{n_i} \in N$ such that $C^i_j \prec C_i$ and $C^i_1, \dots, C^i_{n_i} \models C_i$. We can replace C_i above with $C^i_1, \dots, C^i_{n_i}$. We can iterate the procedure until none of the C_i's are in M (termination guaranteed by the fact that $\succ$ is well-founded).

Some theorem provers for first-order logic

• SPASS	`http://www.spass-prover.org/`
• E	`http://www4.informatik.tu-muenchen.de/~schulz/E/E.html`
• Vampire	`http://www.vprover.org/`

Decidable subclasses of first-order logic

Applications

Use ordered resolution with selection to give a decision procedure for the Ackermann class.

The Ackermann class

$\Sigma = (\Omega, \Pi)$, Ω is a finite set of constants

The Ackermann class consists of all sentences of the form

$$\exists x_1 \ldots \exists x_n \forall x \exists y_1 \ldots \exists y_m F(x_1, \ldots, x_n, x, y_1, \ldots, y_m)$$

Idea: CNF translation:

$$\begin{aligned}
&\exists x_1 \ldots \exists x_n \forall x \exists y_1 \ldots \exists y_m F(x_1, \ldots, x_n, x, y_1, \ldots, y_m) \\
&\qquad \Rightarrow_S \forall x F(\overline{c}_1, \ldots, \overline{c}_n, x, f_1(x), \ldots, f_m(x)) \\
&\qquad \Rightarrow_K \forall x \bigwedge \bigvee L_i(c_1, \ldots, c_n, x, f_1(x), \ldots, f_m(x))
\end{aligned}$$

$c_1, \ldots, c_n$ are Skolem constants

$f_1, \ldots, f_m$ are unary Skolem functions

The Ackermann class

$\Sigma = (\Omega, \Pi)$, Ω is a finite set of constants

The Ackermann class consists of all sentences of the form

$$\exists x_1 \dots \exists x_n \forall x \exists y_1 \dots \exists y_m F(x_1, \dots, x_n, x, y_1, \dots, y_m)$$

Idea: CNF translation:

$$\begin{aligned} &\exists x_1 \dots \exists x_n \forall x \exists y_1 \dots \exists y_m F(x_1, \dots, x_n, x, y_1, \dots, y_m) \\ &\qquad \Rightarrow^* \forall x \bigwedge \bigvee L_i(c_1, \dots, c_n, x, f_1(x), \dots, f_m(x)) \end{aligned}$$

The clauses are in the following classes:

$G = G(c_1, \dots, c_n)$ ground clauses without function symbols
$V = V(x, c_1, \dots, c_n)$ clauses with one variable and without function symbols
$G_f = G(c_1, \dots, c_n, f_1, \dots, f_n)$ ground clauses with function symbols
$V_f = V(x, c_1, \dots, c_n, f_1(x), \dots, f_n(x))$ clauses with a variable & function symbols

The Ackermann class

$G = G(c_1, \dots, c_n)$ ground clauses without function symbols
$V = V(x, c_1, \dots, c_n)$ clauses with one variable and without function symbols
$G_f = G(c_1, \dots, c_n, f_1, \dots, f_n)$ ground clauses with function symbols
$V_f = V(x, c_1, \dots, c_n, f_1(x), \dots, f_n(x))$ clauses with a variable & function symbols

Term ordering

$f(t) \succ t$; terms containing function symbols larger than those who do not.

$B \succ A$ iff exists argument u of B such that every argument t of A: $u \succ t$

Ordered resolution: $G \cup V \cup G_f \cup V_f$ is closed under ordered resolution.

$G, G \mapsto G$; $G, V \mapsto G$; $G, G_f \mapsto$ nothing; $G, V_f \mapsto$ nothing

$V, V \mapsto V \cup G$; $V, G_f \mapsto G \cup G_f$; $V, V_f \mapsto G \cup V \cup G_f \cup V_f$

$G_f, G_f \mapsto G_f$; $G_f, V_f \mapsto G_f \cup G$; $V_f, V_f \mapsto G \cup V \cup V_f \cup G_f$

Observation 1: $G \cup V \cup G_f \cup V_f$ finite set of clauses (up to renaming of variables).

The Ackermann class

$G = G(c_1, \ldots, c_n)$ ground clauses without function symbols
$V = V(x, c_1, \ldots, c_n)$ clauses with one variable and without function symbols
$G_f = G(c_1, \ldots, c_n, f_i)$ ground clauses with function symbols
$V_f = V(x, c_1, \ldots, c_n, f_1(x), \ldots, f_n(x))$ clauses with a variable & function symbols

Term ordering

$f(t) \succ t$; terms containing function symbols larger than those who do not.

$B \succ A$ iff exists argument u of B such that every argument t of A: $u \succ t$

Ordered resolution: $G \cup V \cup G_f \cup V_f$ is closed under ordered resolution.

$G, G \mapsto G$; $G, V \mapsto G$; $G, G_f \mapsto$ nothing; $G, V_f \mapsto$ nothing

$V, V \mapsto V \cup G$; $V, G_f \mapsto G \cup G_f$; $V, V_f \mapsto G \cup V \cup G_f \cup V_f$

$G_f, G_f \mapsto G_f$; $G_f, V_f \mapsto G_f \cup G$; $V_f, V_f \mapsto G \cup V \cup V_f \cup G_f$

Observation 2: No clauses with nested function symbols can be generated.

The Ackermann Class

Conclusion:

Resolution (with implicit factorization) will always terminate if the input clauses are in the class defined before.

Resolution can be used as a decision procedure to check the satisfiability of formulae in the Ackermann class.

The Monadic Class

Monadic first-order logic (MFO) is FOL (without equality) over purely relational signatures $\Sigma = (\Omega, \Pi)$, where $\Omega = \emptyset$, and every $p \in \Pi$ has arity 1.

Abstract syntax:

$$\Phi := \top \mid P(x) \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \forall x \Phi$$

Idea. Let Φ be a MFO formula with k predicate symbols.

Let $\mathcal{A} = (U_{\mathcal{A}}, \{p_{\mathcal{A}}\}_{p \in \Pi})$ be a Σ-algebra. The only way to distinguish the elements of $U_{\mathcal{A}}$ is by the atomic formulae $p(x)$, $p \in \Pi$.

- the elements which $a \in U_{\mathcal{A}}$ which belong to the same $p_{\mathcal{A}}$'s, $p \in \Pi$ can be collapsed into one single element.
- if $\Pi = \{p^1, \ldots, p^k\}$ then what remains is a *finite structure* with at most 2^k elements.
- the truth value of a formula: computed by evaluating all subformulae.

The Monadic Class

MFO Abstract syntax: $\Phi := \top \mid P(x) \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \forall x \Phi$

Theorem (Finite model theorem for MFO). If Φ is a satisfiable MFO formula with k predicate symbols then Φ has a model where the domain is a subset of $\{0,1\}^k$.

Proof: Let $\mathcal{B} = (\{0,1\}^k, \{p^1_{\mathcal{B}}, \ldots, p^k_{\mathcal{B}}\})$, where $p^i_{\mathcal{B}}{=}\{(b_1, \ldots, b_k) \mid b_i{=}1\}$.

Let $\mathcal{A} = (U_{\mathcal{A}}, \{p^1_{\mathcal{A}}, \ldots, p^k_{\mathcal{A}}\})$, $\beta : X \to U_{\mathcal{A}}$ be such that $(\mathcal{A}, \beta) \models \Phi$.
We construct a model for Φ with cardinality at most 2^k as follows:

- Let $h : \mathcal{A} \to \mathcal{B}$ be defined for all $a \in U_{\mathcal{A}}$ by:

  $$h(a) = (b_1, \ldots, b_k) \text{ where } b_i = 1 \text{ if } a \in p^i_{\mathcal{A}} \text{ and } 0 \text{ otherwise.}$$

  Then $a \in p^i_{\mathcal{A}}$ iff $h(a) \in p^i_{\mathcal{B}}$ for all $a \in U_{\mathcal{A}}$ and all $i = 1, \ldots, k$.
- Let $\mathcal{B}' = (\{0,1\}^k \cap h(U_{\mathcal{A}}), \{p^1_{\mathcal{B}} \cap h(U_{\mathcal{A}}), \ldots, p^k_{\mathcal{B}} \cap h(U_{\mathcal{A}})\})$.
- We show that $(\mathcal{B}', \beta \circ h) \models \Phi$.

The Monadic Class

Let $\mathcal{B} = (\{0,1\}^k, \{p^1_{\mathcal{B}}, \dots, p^k_{\mathcal{B}}\})$, where $p^i_{\mathcal{B}}{=}\{(b_1, \dots, b_k) \mid b_i{=}1\}$.

Let $\mathcal{A} = (U_{\mathcal{A}}, \{p^1_{\mathcal{A}}, \dots, p^k_{\mathcal{A}}\})$, $\beta : X \to U_{\mathcal{A}}$ be such that $(\mathcal{A}, \beta) \models \Phi$.
We construct a model for Φ with cardinality at most 2^k as follows:

- Let $h : \mathcal{A} \to \mathcal{B}$ be defined for all $a \in U_{\mathcal{A}}$ by:

  $$h(a) = (b_1, \dots, b_k) \text{ where } b_i = 1 \text{ if } a \in p^i_{\mathcal{A}} \text{ and 0 otherwise.}$$

  Then $a \in p^i_{\mathcal{A}}$ iff $h(a) \in p^i_{\mathcal{B}}$ for all $a \in U_{\mathcal{A}}$ and all $i = 1, \dots, k$.
- Let $\mathcal{B}' = (\{0,1\}^k \cap h(U_{\mathcal{A}}), \{p^1_{\mathcal{B}} \cap h(U_{\mathcal{A}}), \dots, p^k_{\mathcal{B}} \cap h(U_{\mathcal{A}})\})$.
- We show that $(\mathcal{B}', \beta \circ h) \models \Phi$.

Induction on the structure of Φ

- $\Phi = \top$ OK
- $\Phi = p^i(x)$. Then $(\mathcal{A}, \beta) \models \Phi$ iff $\beta(x) \in p^i_{\mathcal{A}}$ iff $h(\beta(x)) \in p^i_{\mathcal{B}}$ iff $(\mathcal{B}', \beta \circ h) \models \Phi$.

The Monadic Class

Let $\mathcal{B} = (\{0,1\}^k, \{p^1_{\mathcal{B}}, \dots, p^k_{\mathcal{B}}\})$, where $p^i_{\mathcal{B}}=\{(b_1, \dots, b_k) \mid b_i=1\}$.

Let $\mathcal{A} = (U_{\mathcal{A}}, \{p^1_{\mathcal{A}}, \dots, p^k_{\mathcal{A}}\})$, $\beta : X \rightarrow U_{\mathcal{A}}$ be such that $(\mathcal{A}, \beta) \models \Phi$.
We construct a model for Φ with cardinality at most 2^k as follows:

- Let $h : \mathcal{A} \rightarrow \mathcal{B}$ be defined for all $a \in U_{\mathcal{A}}$ by:

  $$h(a) = (b_1, \dots, b_k) \text{ where } b_i = 1 \text{ if } a \in p^i_{\mathcal{A}} \text{ and } 0 \text{ otherwise.}$$

  Then $a \in p^i_{\mathcal{A}}$ iff $h(a) \in p^i_{\mathcal{B}}$ for all $a \in U_{\mathcal{A}}$ and all $i = 1, \dots, k$.
- Let $\mathcal{B}' = (\{0,1\}^k \cap h(U_{\mathcal{A}}), \{p^1_{\mathcal{B}} \cap h(U_{\mathcal{A}}), \dots, p^k_{\mathcal{B}} \cap h(U_{\mathcal{A}})\})$.
- We show that $(\mathcal{B}', \beta \circ h) \models \Phi$.

Induction on the structure of Φ

- $\Phi = \Phi_1 \wedge \Phi_2$: standard
- $\Phi = \neg\Phi_1$: standard

The Monadic Class

Let $\mathcal{B} = (\{0,1\}^k, \{p^1_\mathcal{B}, \dots, p^k_\mathcal{B}\})$, where $p^i_\mathcal{B} = \{(b_1, \dots, b_k) \mid b_i = 1\}$.

Let $\mathcal{A} = (U_\mathcal{A}, \{p^1_\mathcal{A}, \dots, p^k_\mathcal{A}\})$, $\beta : X \to U_\mathcal{A}$ be such that $(\mathcal{A}, \beta) \models \Phi$.
We construct a model for Φ with cardinality at most 2^k as follows:

- Let $h : \mathcal{A} \to \mathcal{B}$ be defined for all $a \in U_\mathcal{A}$ by:

  $$h(a) = (b_1, \dots, b_k) \text{ where } b_i = 1 \text{ if } a \in p^i_\mathcal{A} \text{ and } 0 \text{ otherwise.}$$

  Then $a \in p^i_\mathcal{A}$ iff $h(a) \in p^i_\mathcal{B}$ for all $a \in U_\mathcal{A}$ and all $i = 1, \dots, k$.
- Let $\mathcal{B}' = (\{0,1\}^k \cap h(U_\mathcal{A}), \{p^1_\mathcal{B} \cap h(U_\mathcal{A}), \dots, p^k_\mathcal{B} \cap h(U_\mathcal{A})\})$.
- We show that $(\mathcal{B}', \beta \circ h) \models \Phi$.

Induction on the structure of Φ

- $\Phi = \forall x \Phi_1(x)$. Then the following are equivalent:
  - $(\mathcal{A}, \beta) \models \Phi$ (i.e. $(\mathcal{A}, \beta[x \mapsto a]) \models \Phi_1$ for all $a \in U_\mathcal{A}$)
  - $(\mathcal{B}', \beta[x \mapsto a] \circ h) \models \Phi_1$ for all $a \in U_\mathcal{A}$ (ind. hyp)
  - $(\mathcal{B}', \beta \circ h[x \mapsto b]) \models \Phi_1$ for all $b \in \{0,1\}^k \cap h(A)$ (i.e. $(\mathcal{B}', \beta \circ h) \models \Phi$)

The Monadic Class

Resolution-based decision procedure for the Monadic Class (and for several other classes):

William H. Joyner Jr.
Resolution Strategies as Decision Procedures.
J. ACM 23(3): 398-417 (1976)

Idea:

- Use orderings to restrict the possible inferences
- Identify a class of clauses (with terms of bounded depth) which contains the type of clauses generated from the respective fragment and is closed under ordered resolution (+ red. elim. criteria)
- Show that a saturation of the clauses can be obtained in finite time

The Monadic Class

Resolution-based decision procedure for the Monadic Class:

$$\Phi: \quad \forall \overline{x}_1 \exists \overline{y}_1 \ldots \forall \overline{x}_k \exists \overline{y}_k (\ldots.p^s(x_i)\ldots\ldots p^l(y_i)\ldots)$$

$$\mapsto \quad \forall \overline{x}_1 \ldots \forall \overline{x}_k (\ldots p^s(x_i)\ldots p^l(f_{\mathsf{sk}}(\overline{x}_1, \ldots, \overline{x}_i)\ldots)$$

Consider the class MON of clauses with the following properties:

- no literal of heigth greater than 2 appears
- each variable-disjoint partition has at most $n = \sum_{i=1} |\overline{x}_i|$ variables (can order the variables as $x_1, \ldots, x_n$)
- the variables of each non-ground block can occur either in atoms $p(x_i)$ or in atoms $P(f_{\mathsf{sk}}(x_1, \ldots, x_t))$, $0 \leq t \leq n$

It can be shown that this class contains all CNF's of formulae in the monadic class and is closed under ordered resolution.

3.2 Deduction problems

Satisfiability w.r.t. a theory

Satisfiability w.r.t. a theory

Example

Let $\Sigma = (\{e/0, */2, i/1\}, \emptyset)$

Let $\mathcal{F}$ consist of all (universally quantified) group axioms:

$$
\begin{array}{ll}
\forall x, y, z & x * (y * z) \approx (x * y) * z \\
\forall x & x * i(x) \approx e \quad \wedge \quad i(x) * x \approx e \\
\forall x & x * e \approx x \quad \wedge \quad e * x \approx x
\end{array}
$$

Question: Is $\forall x, y(x * y = y * x)$ entailed by $\mathcal{F}$?

Satisfiability w.r.t. a theory

Example

Let $\Sigma = (\{e/0, */2, i/1\}, \emptyset)$

Let $\mathcal{F}$ consist of all (universally quantified) group axioms:

$$
\begin{array}{llcl}
\forall x, y, z & x * (y * z) \approx (x * y) * z & & \\
\forall x & x * i(x) \approx e & \wedge & i(x) * x \approx e \\
\forall x & x * e \approx x & \wedge & e * x \approx x
\end{array}
$$

Question: Is $\forall x, y(x * y = y * x)$ entailed by $\mathcal{F}$?

Alternative question:

Is $\forall x, y(x * y = y * x)$ true in the class of all groups?

Logical theories

Syntactic view

first-order theory: given by a set $\mathcal{F}$ of (closed) first-order Σ-formulae.

the models of $\mathcal{F}$: $\quad \mathsf{Mod}(\mathcal{F}) = \{\mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$

Semantic view

given a class $\mathcal{M}$ of Σ-algebras

the first-order theory of $\mathcal{M}$: $\mathsf{Th}(\mathcal{M}) = \{G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G\}$

Decidable theories

Let $\Sigma = (\Omega, \Pi)$ be a signature.

$\mathcal{M}$: class of Σ-algebras. $\mathcal{T} = \mathrm{Th}(\mathcal{M})$ is decidable

iff

there is an algorithm which, for every closed first-order formula ϕ, can decide (after a finite number of steps) whether ϕ is in $\mathcal{T}$ or not.

$\mathcal{F}$: class of (closed) first-order formulae.

The theory $\mathcal{T} = \mathrm{Th}(\mathrm{Mod}(\mathcal{F}))$ is decidable

iff

there is an algorithm which, for every closed first-order formula ϕ, can decide (in finite time) whether $\mathcal{F} \models \phi$ or not.

Examples

Undecidable theories

- Th$((\mathbb{Z}, \{0, 1, +, *\}, \{\leq\}))$
- Peano arithmetic

- Th(Σ-alg)

Peano arithmetic

Peano axioms:

$$\forall x \neg(x + 1 \approx 0) \quad \text{(zero)}$$

$$\forall x \forall y\, (x + 1 \approx y + 1 \rightarrow x \approx y \quad \text{(successor)}$$

$$F[0] \wedge (\forall x\, (F[x] \rightarrow F[x + 1]) \rightarrow \forall x F[x]) \quad \text{(induction)}$$

$$\forall x\, (x + 0 \approx x) \quad \text{(plus zero)}$$

$$\forall x, y\, (x + (y + 1) \approx (x + y) + 1) \quad \text{(plus successor)}$$

$$\forall x, y\, (x * 0 \approx 0) \quad \text{(times 0)}$$

$$\forall x, y\, (x * (y + 1) \approx x * y + x) \quad \text{(times successor)}$$

$3 * y + 5 > 2 * y$ expressed as $\exists z(z \neq 0 \wedge 3 * y + 5 \approx 2 * y + z)$

Intended interpretation: $(\mathbb{N}, \{0, 1, +, *\}, \{\approx, \leq\})$

(does not capture true arithmetic by Goedel's incompleteness theorem)

Examples

Undecidable theories

$\bullet$Th$((\mathbb{Z}, \{0, 1, +, *\}, \{\leq\}))$
$\bullet$ Peano arithmetic

$\bullet$Th(Σ-alg)

Idea of undecidability proof: Suppose there is an algorithm P that, given a formula in one of the theories above decides whether that formula is valid.

We use P to give a decision algorithm for the language

$\{(G(M), w) | G(M)$ is the Gödelisation of a TM M that accepts the string w $\}$

As the latter problem is undecidable, this will show that P cannot exist.

Examples

Undecidable theories

- $\mathsf{Th}((\mathbb{Z}, \{0, 1, +, *\}, \{\leq\}))$
- Peano arithmetic
- $\mathsf{Th}(\Sigma\text{-alg})$

Idea of undecidability proof: (ctd)

(1) For $\mathsf{Th}((\mathbb{Z}, \{0, 1, +, *\}, \{\leq\}))$ and Peano arithmetic:

multiplication can be used for modeling Gödelisation

(2) For $\mathsf{Th}(\Sigma\text{-alg})$:

Given M and w, we create a FOL signature and a set of formulae over this signature encoding the way M functions, and a formula which is valid iff M accepts w.

Examples

In order to obtain decidability results:

- Restrict the signature
- Enrich axioms
- Look at certain fragments

Examples

In order to obtain decidability results:

- Restrict the signature
- Enrich axioms
- Look at certain fragments

Decidable theories

- Presburger arithmetic decidable in 3EXPTIME [Presburger'29]
  Signature: $(\{0, 1, +\}, \{\approx, \leq\})$ (no $*$)

  Axioms { (zero), (successor), (induction), (plus zero), (plus successor) }

- $\mathsf{Th}(\mathbb{Z}_+)$ $\quad\quad$ $\mathbb{Z}_+ = (\mathbb{Z}, 0, s, +, \leq)$ the standard interpretation of integers.

Examples

In order to obtain decidability results:

- Restrict the signature
- Enrich axioms
- Look at certain fragments

Decidable theories

- The theory of real numbers (with addition and multiplication) is decidable in 2EXPTIME [Tarski'30]

Examples

In order to obtain decidability results:

- Restrict the signature
- Enrich axioms
- Look at certain fragments

Problems

$\mathcal{T}$: first-order theory in signature Σ; $\mathcal{L}$ class of (closed) Σ-formulae

Given ϕ in $\mathcal{L}$, is it the case that $\mathcal{T} \models \phi$?

Common restrictions on $\mathcal{L}$

	Pred $= \emptyset$	$\{\phi \in \mathcal{L} \mid \mathcal{T} \models \phi\}$
$\mathcal{L}=\{\forall x A(x) \mid A \text{ atomic}\}$	word problem	
$\mathcal{L}=\{\forall x (A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B \text{ atomic}\}$	uniform word problem	$Th_{\forall \text{Horn}}$
$\mathcal{L}=\{\forall x C(x) \mid C(x) \text{ clause}\}$	clausal validity problem	$Th_{\forall,\text{cl}}$
$\mathcal{L}=\{\forall x \phi(x) \mid \phi(x) \text{ unquantified}\}$	universal validity problem	$Th_{\forall}$
$\mathcal{L}=\{\exists x A_1 \wedge \ldots \wedge A_n \mid A_i \text{ atomic}\}$	unification problem	$Th_{\exists}$
$\mathcal{L}=\{\forall x \exists x A_1 \wedge \ldots \wedge A_n \mid A_i \text{ atomic}\}$	unification with constants	$Th_{\forall\exists}$

$\mathcal{T}$-validity vs. $\mathcal{T}$-satisfiability

$\mathcal{T}$-validity: Let $\mathcal{T}$ be a first-order theory in signature Σ
Let $\mathcal{L}$ be a class of (closed) Σ-formulae

Given ϕ in $\mathcal{L}$, is it the case that $\mathcal{T} \models \phi$?

Remark: $\mathcal{T} \models \phi$ iff $\mathcal{T} \cup \neg\phi$ unsatisfiable

Every $\mathcal{T}$-validity problem has a dual $\mathcal{T}$-satisfiability problem:

$\mathcal{T}$-satisfiability: Let $\mathcal{T}$ be a first-order theory in signature Σ
Let $\mathcal{L}$ be a class of (closed) Σ-formulae

$$\neg\mathcal{L} = \{\neg\phi \mid \phi \in \mathcal{L}\}$$

Given ψ in $\neg\mathcal{L}$, is it the case that $\mathcal{T} \cup \psi$ is satisfiable?

$\mathcal{T}$-validity vs. $\mathcal{T}$-satisfiability

Common restrictions on $\mathcal{L}$ / $\neg\mathcal{L}$

$\mathcal{L}$	$\neg\mathcal{L}$
$\{\forall x A(x) \mid A \text{ atomic}\}$	$\{\exists x \neg A(x) \mid A \text{ atomic}\}$
$\{\forall x (A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B \text{ atomic}\}$	$\{\exists x (A_1 \wedge \ldots \wedge A_n \wedge \neg B) \mid A_i, B \text{ atomic}\}$
$\{\forall x \bigvee L_i \mid L_i \text{ literals}\}$	$\{\exists x \bigwedge L'_i \mid L'_i \text{ literals}\}$
$\{\forall x \phi(x) \mid \phi(x) \text{ unquantified}\}$	$\{\exists x \phi'(x) \mid \phi'(x) \text{ unquantified}\}$

validity problem for universal formulae   ground satisfiability problem

$\mathcal{T}$-validity vs. $\mathcal{T}$-satisfiability

Common restrictions on $\mathcal{L}$ / $\neg\mathcal{L}$

$\mathcal{L}$	$\neg\mathcal{L}$
$\{\forall x A(x) \mid A \text{ atomic}\}$	$\{\exists x \neg A(x) \mid A \text{ atomic}\}$
$\{\forall x (A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B \text{ atomic}\}$	$\{\exists x (A_1 \wedge \ldots \wedge A_n \wedge \neg B) \mid A_i, B \text{ atomic}\}$
$\{\forall x \bigvee L_i \mid L_i \text{ literals}\}$	$\{\exists x \bigwedge L'_i \mid L'_i \text{ literals}\}$
$\{\forall x \phi(x) \mid \phi(x) \text{ unquantified}\}$	$\{\exists x \phi'(x) \mid \phi'(x) \text{ unquantified}\}$

validity problem for universal formulae

ground satisfiability problem

In what follows we will focus on the problem of checking the satisfiability of conjunctions of ground literals

$\mathcal{T}$-validity vs. $\mathcal{T}$-satisfiability

$\mathcal{T} \models \forall x A(x)$	iff	$\mathcal{T} \cup \exists x \neg A(x)$ unsatisfiable
$\mathcal{T} \models \forall x (A_1 \wedge \cdots \wedge A_n \rightarrow B)$	iff	$\mathcal{T} \cup \exists x (A_1 \wedge \cdots \wedge A_n \wedge \neg B)$ unsatisfiable
$\mathcal{T} \models \forall x (\bigvee_{i=1}^n A_i \vee \bigvee_{j=1}^m \neg B_j)$	iff	$\mathcal{T} \cup \exists x (\neg A_1 \wedge \cdots \wedge \neg A_n \wedge B_1 \wedge \cdots \wedge B_m)$ unsatisfiable

$\mathcal{T}$-satisfiability vs. Constraint Solving

The field of Constraint Solving also deals with satisfiability problems

But be careful:

- in Constraint Solving one is interested if a formula is satisfiable in a given, fixed model of $\mathcal{T}$.
- in $\mathcal{T}$-satisfiability one is interested if a formula is satisfiable in any model of $\mathcal{T}$ at all.