Contents

CHAPTER 1

Introduction

1.1 Preliminaries

The language of a first-order formula A is denoted by $\mathrm{L}(A)$ and contains all predicate, constant, function and free variable symbols that occur in A. These are also referred to as the *non-logical symbols* of A.

An occurrence of Φ-term is called *maximal* if it does not occur as subterm of another Φ-term.

We denote $x_1, \dots, x_n$ by $\bar{x}$.

A substitution is a mapping of variables to terms. It is denoted by $\phi[x/t]$, where ϕ is a formula or term where each occurrence of the variable x is replaced by the term t. A substitution σ is called trivial on x if $x\sigma = x$. Otherwise it is called non-trivial.

A term replacement on the other hand is a mapping of terms to terms. It is denoted by $\phi\{s/t\}$, where ϕ is a formula or term where each occurrence of the term t is replaced by the term s.

is term replacement apt?

1.2 Craig Interpolation

Theorem 1.1 (Interpolation). *Let Γ and Δ be sets of first-order formulas such that $\Gamma \cup \Delta$ is unsatisfiable. Then there exists a first-order formula I, called interpolant, such that*

1. $\Gamma \models I$
2. $\Delta \models \neg I$
3. $\mathrm{L}(I) \subseteq \mathrm{L}(\Gamma) \cap \mathrm{L}(\Delta)$. $\Box$

In the context of interpolation, every non-logical symbol is assigned a color which indicates the its origin(s). A non-logical symbol is said to be Γ *(Δ)-colored* if it only occurs in Γ (Δ) and *grey* in case it occurs in both Γ and Δ.

CHAPTER 2

The Resolution Calculus

2.1 Resolution

Resolution calculus, in the formulation as given here, is a sound and complete calculus for first-order logic with equality. Due to the simplicity of its rules, it is widely used in the area of automated deduction.

Definition 2.1. A *clause* is a finite set of literals. The empty clause will be denoted by $\Box$. A *resolution refutation* of a set of clauses Γ is a derivation of $\Box$ consisting of applications of resolution rules (cf. figure 2.1) starting from clauses in Γ. $\triangle$

Theorem 2.2. *A clause set Γ is unsatisfiable if and only if there is resolution refutation of Γ.*

Proof. See [Rob65]. $\Box$

Clauses will usually be denoted by C or D, literals by l.

$$\textit{Resolution:}\quad \frac{C \vee l \qquad D \vee \neg l'}{(C \vee D)\sigma} \quad \sigma = \mathrm{mgu}(l, l')$$

$$\textit{Factorisation:}\quad \frac{C \vee l \vee l'}{(C \vee l)\sigma} \quad \sigma = \mathrm{mgu}(l, l')$$

$$\textit{Paramodulation:}\quad \frac{C \vee s = t \qquad D[r]}{(C \vee D[t])\sigma} \quad \sigma = \mathrm{mgu}(s, r)$$

Figure 2.1: The rules of resolution calculus

2.2 Resolution and Interpolation

In order to apply resolution to arbitrary first-order formulas, they have to be converted to clauses first. This usually makes use of intermediate normal forms which are defined as follows:

Definition 2.3. A formula is in *Negation Normal Form (NNF)* if negations only occur directly befor of atoms. A formula is in *Conjunctive Normal Form (CNF)* if it is a conjunction of disjunctions of literals. $\triangle$

In this context, the conjuncts of a CNF-formula are interpreted as clauses. A well-established procedure for the translation to CNF is comprised of the following steps:

1. NNF-Transformation
2. Skolemisation
3. CNF-Transformation

Step 1 can be achieved by solely pushing the negation inwards. As this transformation yields an equivalent formula, it clearly has no effect on the interpolants. Step 2 and 3 on the other hand do not produce equivalent formulas since they introduce new symbols. In this section, we will show that they nonetheless do preserve the set of interpolants. This fact is vital for the use of resolution-based methods for interpolant computation of arbitrary formulas.

2.2.1 Interpolation and Skolemisation

Skolemisation is a procedure for replacing existential quantifiers with Skolem terms:

Definition 2.4. Let $V_{\exists x}$ be the set of universally bound variables in the scope of the occurrence of $\exists x$ in a formula. The skolemisation of a formula A in NNF, denoted by $\mathrm{sk}(A)$, is the result of replacing every occurrence of an existential quantifier $\exists x$ in A by a term $f(y_1, \dots, y_n)$ where f is a new Skolem function symbol and $V_{\exists x} = \{y_1, \dots, y_n\}$. In case $V_{\exists x}$ is empty, the occurrence of $\exists x$ is replaced by a new Skolem constant symbol c.

The skolemisation of a set of formulas Φ is defined to be $\mathrm{sk}(\Phi) = \{\mathrm{sk}(A) \mid A \in \Phi\}$. $\triangle$

Proposition 2.5. *Let* $\Gamma \cup \Delta$ *be unsatisfiable. Then* I *is an interpolant for* $\Gamma \cup \Delta$ *if and only if it is an interpolant for* $\mathrm{sk}(\Gamma) \cup \mathrm{sk}(\Delta)$.

Proof. Since $\mathrm{sk}(\cdot)$ adds fresh symbols to both Γ and Δ individually, none of them are containd in $\mathrm{L}(\mathrm{sk}(\Gamma)) \cap \mathrm{L}(\mathrm{sk}(\Delta))$. Therefore condition 3 of theorem 1.1 is satisfied in both directions.

As for any set of formulas Φ, each model of Φ can be extended to a model of $\mathrm{sk}(\Phi)$ and every model of $\mathrm{sk}(\Phi)$ is a witness for the satisfiability of Φ, $\Phi \models I$ iff $\mathrm{sk}(\Phi) \models I$. Hence conditions 1 and 2 of theorem 1.1 remain satisfied for I as well. $\square$

2.2.2 Interpolation and structure-preserving Normal Form Transformation

A common method for transforming a skolemised formula A into CNF while preserving their structure is defined as follows:

Definition 2.6. For every occurrence of a subformula B of A, introduce a new atom L_B which acts as a label for the subformula. For each of them, create a defining clause D_B:

does it suffice to not treat universal quantifiers specifically here? (subterms have free variables; possibly need to mention to just pull universal quantifiers outwards to get prenex form and drop quantifiers)

If B is atomic:

$$D_B \equiv (\neg B \vee L_B) \wedge (B \vee \neg L_B)$$

If B is $\neg G$:

$$D_B \equiv (L_B \vee L_G) \wedge (\neg L_B \vee \neg L_G)$$

If B is $G \wedge H$:

$$D_B \equiv (\neg L_B \vee L_G) \wedge (\neg L_B \vee L_H) \wedge (L_B \vee \neg L_G \vee \neg L_H)$$

If B is $G \vee H$:

$$D_B \equiv (L_B \vee \neg L_G) \wedge (L_B \vee \neg L_H) \wedge (\neg L_B \vee L_G \vee L_H)$$

If B is $G \supset H$:

$$D_B \equiv (L_B \vee L_G) \wedge (L_B \vee \neg L_H) \wedge (\neg L_B \vee \neg L_G \vee L_H)$$

If B is $\forall x G$:

$$D_B \equiv \forall x(\neg L_B \vee L_G) \wedge \forall x(L_B \vee \neg L_G)$$

Let $\delta(A)$ be defined as $\bigwedge_{B \in \Sigma(A)} D_B \wedge L_A$, where $\Sigma(A)$ denotes the set of occurrences of subformulas of A. △

Proposition 2.7. *Let A be a formula. Then $\mathrm{sk}(A)$ is unsatisfiable if and only if $\delta(\mathrm{sk}(A))$ is unsatisfiable.*

Proposition 2.8. *Let $\mathrm{sk}(\Gamma) \cup \mathrm{sk}(\Delta)$ be unsatisfiable. Then I is an interpolant for $\mathrm{sk}(\Gamma) \cup \mathrm{sk}(\Delta)$ if and only if I is an interpolant for $\delta(\mathrm{sk}(\Gamma)) \cup \delta(\mathrm{sk}(\Delta))$.*

Proof. As δ introduces fresh symbols for each $\mathrm{sk}(\Gamma)$ and $\mathrm{sk}(\Delta)$, they must not occur in any interpolant of $\mathrm{sk}(\Gamma)$ and $\mathrm{sk}(\Delta)$. This establishes condition 3 of theorem 1.1 in both directions.

Using proposition 2.7, condition 1 and 2 of theorem 1.1 are immediate. □

CHAPTER 3

Proof by Reduction to First-Order Logic without Equality

A common theme of proofs in theoretical computer science is to instead of proving the result from first principles to reduce the problem to another one, which then is easier to solve. In this instance, we are able to give a reduction for finding interpolants for first-order logic *with* equality to first-order logic *without* equality, where it is simpler to give an appropriate algorithm.

The general layout of this approach is the following: From two sets Γ and Δ, where $\Gamma \cup \Delta$ is unsatisfiable, we compute Γ' and Δ' which do not make use of equality but simulate equality it via axioms. In the process of this transformation, also function symbols are replaced by predicate symbols with appropriate axioms to make sure that their behaviour is compatible to the one of functions. Now an interpolant of Γ' and Δ' can be derived using an algorithm that is only capable of handling predicate symbols, as all other non-logical symbols have been removed. Since the additional axioms ensure that the newly added predicate symbols mimic equality and functions respectively, we will see that the occurrences of these predicates in the interpolant can be translated back to occurrences of equality and function symbols in first-order logic with equality in the language of Γ and Δ, thereby yielding the originally desired interpolant.

3.1 Reduction to first-order logic without equality

As we shall see in this section, first-order formulas with equality can be transformed into first-order formulas without equality in a way that is satisfiability-preserving, which is sufficient for our purposes.

In order to simplify notation, we shall consider constant symbols to be function symbols of arity 0 in this section.

First, we define the axioms which allow for simulation of equality and functions in first order logic without equality and function symbols:

Definition 3.1. For a first-order language $\mathcal{L}$ and fresh predicate symbols E and F_f for $f \in \mathrm{FS}(\mathcal{L})$, we define:

$$\begin{aligned}
\mathrm{F}_{\mathrm{Ax}}(\mathcal{L}) &\overset{\text{def}}{=} \bigwedge_{f \in \mathrm{FS}(\mathcal{L})} \forall \bar{x} \exists y (F_f(\bar{x}, y) \wedge (\forall z (F_f(\bar{x}, z) \supset E(y, z)))) \\
\mathrm{Refl}(P) &\overset{\text{def}}{=} \forall x P(x, x) \\
\mathrm{Congr}(P) &\overset{\text{def}}{=} \forall x_1 \forall y_1 \ldots \forall x_{\mathrm{ar}(P)} \forall y_{\mathrm{ar}(P)} ((E(x_1, y_1) \wedge \ldots \wedge E(x_{\mathrm{ar}(P)}, y_{\mathrm{ar}(P)})) \supset \\
&\qquad (P(x_1, \ldots, x_{\mathrm{ar}(P)}) \supset P(y_1, \ldots, y_{\mathrm{ar}(P)}))) \\
\mathrm{E}_{\mathrm{Ax}}(\mathcal{L}) &\overset{\text{def}}{=} \mathrm{Refl}(E) \wedge \bigwedge_{\substack{P \in \mathrm{PS}(\mathcal{L}) \cup \{E\} \cup \\ \{F_f | f \in \mathrm{FS}(\mathcal{L})\}}} \mathrm{Congr}(P) \qquad \triangle
\end{aligned}$$

$\mathrm{Refl}(P)$ will be referred to as reflexivity axiom of P, $\mathrm{Congr}(P)$ as congruence axiom of P.

Definition 3.2. Let A be a first-order formula and E and F_f for $f \in \mathrm{FS}(A)$ be fresh predicate symbols. Then $\mathrm{T}(\mathrm{L}(A))$ denotes $(\mathrm{L}(A) \cup \{E\} \cup \{F_f \mid f \in \mathrm{FS}(A)\}) \setminus (\{=\} \cup \mathrm{FS}(A))$.

Moreover, $\mathrm{T}(A)$ is the result of applying the following algorithm to A:

1. Replace every occurrence of $s = t$ in A by $E(s, t)$
2. As long as there is an occurrence of a function symbol f in A:
   Let B be the atom in which f as outermost symbol of a term occurs. Then B is of the form $P(s_1, \ldots, s_{j-1}, f(\bar{t}), s_{j+1}, \ldots s_m)$. Replace B in A by $\exists y (F_f(\bar{t}, y) \wedge P(s_1, \ldots, s_{j-1}, y, s_{j+1}, \ldots s_m))$ for a variable y which does not occur free in B.

Furthermore, let the inverse operation $\mathrm{T}^{-1}(B)$ for formulas in the language $\mathrm{T}(L(A))$ be defined as the result of applying the following algorithm to B:

1. Replace every occurrence of $E(s, t)$ in B by $s = t$.
2. For every $f \in FS(A)$, replace every occurrence of $\exists y (F_f(\bar{t}, y) \wedge P(s_1, \ldots, s_{j-1}, y, s_{j+1}, \ldots s_m))$ in B by $P(s_1, \ldots, s_{j-1}, f(\bar{t}), s_{j+1}, \ldots s_m)$ and every remaining occurrence of $F_f(\bar{t}, s)$ by $f(\bar{t}) = s$.

For sets of first-order formulas Φ, let $\mathrm{T}(\Phi) \overset{\text{def}}{=} \bigcup_{A \in \Phi} \mathrm{T}(A)$ and $\mathrm{T}^{-1}(\Phi) \overset{\text{def}}{=} \bigcup_{A \in \Phi} \mathrm{T}^{-1}(A)$. $\triangle$

Lemma 3.3. *Let A be a first-order formula and Φ be a set of first-order formulas. Then* $\mathrm{T}^{-1}(\mathrm{T}(A)) = A$ *and* $\mathrm{T}^{-1}(\mathrm{T}(\Phi)) = \Phi$.

Proof. Step 1 and 2 in the transformation algorithm for T and T^{-1} are concerned with different symbols each and therefore do not interfere with each other. Moreover, the respective steps in both algorithms are the inverse of each other. For step 1, this is immediate and for step 2, consider that all occurrences of F_f for $f \in \mathrm{FS}(A)$ in $\mathrm{T}(A)$ have been introduced by T and are consequently of the form $\exists y (F_f(\bar{t}, y) \wedge P(s_1, \ldots, s_{j-1}, y, s_{j+1}, \ldots s_m))$, which in T^{-1} is replaced by $P(s_1, \ldots, s_{j-1}, f(\bar{t}), s_{j+1}, \ldots s_m)$. $\square$

Definition 3.4. For first-order formulas A, let $\mathrm{T}_{\mathrm{Ax}}(A) = \mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(A)) \land \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(A)) \land \mathrm{T}(A)$ and for sets of first-order formulas Φ, let $\mathrm{T}_{\mathrm{Ax}}(\Phi) = \{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Phi)), \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Phi))\} \cup \mathrm{T}(\Phi)$. △

Lemma 3.5. $\mathrm{T}_{\mathrm{Ax}}(\Gamma \cup \Delta) \Leftrightarrow \mathrm{T}_{\mathrm{Ax}}(\Gamma) \cup \mathrm{T}_{\mathrm{Ax}}(\Delta)$.

Proof.

$$\begin{aligned}
\mathrm{T}_{\mathrm{Ax}}(\Gamma \cup \Delta) &\Leftrightarrow \{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Gamma \cup \Delta)), \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Gamma \cup \Delta))\} \cup \mathrm{T}(\Gamma \cup \Delta) \\
&\Leftrightarrow \{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Gamma) \cup \mathrm{L}(\Delta)), \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Gamma) \cup \mathrm{L}(\Delta))\} \cup \mathrm{T}(\Gamma \cup \Delta) \\
&\Leftrightarrow \{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Gamma)) \land \mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Delta)), \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Gamma)) \land \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Delta))\} \cup \mathrm{T}(\Gamma) \cup \mathrm{T}(\Delta) \\
&\Leftrightarrow \{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Gamma)), \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Gamma))\} \cup \{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Delta)) \land \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Delta))\} \cup \mathrm{T}(\Gamma) \cup \mathrm{T}(\Delta) \\
&\Leftrightarrow \mathrm{T}_{\mathrm{Ax}}(\Gamma) \cup \mathrm{T}_{\mathrm{Ax}}(\Delta)
\end{aligned}$$
□

Note that $\mathrm{T}_{\mathrm{Ax}}(A)$ contains neither the equality predicate nor function symbols but additional predicate symbols instead. More formally:

Lemma 3.6. *Let Φ be a set of first-order formulas. Then $\mathrm{T}_{\mathrm{Ax}}(\Phi)$ is in the language $\mathrm{T}(\mathrm{L}(\Phi))$.*

Proposition 3.7. *Let Φ be a set of first-order formulas.*

1. *If Φ is satisfiable, then so is $\mathrm{T}_{\mathrm{Ax}}(\Phi)$.*
2. *Let $\mathcal{L}$ be a first-order language and Φ a set of first-order formulas in the language $\mathrm{T}(\mathcal{L})$. If $\{\mathrm{F}_{\mathrm{Ax}}(\mathcal{L}), \mathrm{E}_{\mathrm{Ax}}(\mathcal{L})\} \cup \Phi$ is satisfiable, then so is $\mathrm{T}^{-1}(\Phi)$.*

Proof. Suppose Φ is satisfiable. Let M be a model of Φ. We show that $\mathrm{T}_{\mathrm{Ax}}(\Phi)$ is satisfiable by extending M to the language $\mathrm{L}(\Phi) \cup \{E\} \cup \{F_f \mid f \in \mathrm{FS}(A)\}$ to satisfy to a model of $\mathrm{T}_{\mathrm{Ax}}(\Phi)$.

First, let $M \models E(s,t)$ if and only if $M \models s = t$. By reflexivity of equality, it follows that $M \models \mathrm{Refl}(E)$ and as equality satisfies the congruence axiom we get that M is a model $\mathrm{Congr}(E)$ and in consequence also of $\mathrm{E}_{\mathrm{Ax}}(\Phi)$.

Second, let $M \models F_f(\bar{x}, y)$ if and only if $M \models f(\bar{x}) = y$ for all $f \in \mathrm{FS}(\Phi)$. Since M is a model of Φ, it maps f to a function, which returns a unique result for every combination of parameters. This however is precisely the logical requirement on F_f stated by $\mathrm{F}_{\mathrm{Ax}}(\Phi)$, hence M is a model of $\mathrm{F}_{\mathrm{Ax}}(\Phi)$.

Lastly, we show that $M \models A$ for all $A \in \Phi$. By the above definition of E in M, step 1 of the algorithm in definition 3.2 yields a formula that is satisfied by M. For step 2, suppose $P(s_1, \dots, s_{j-1}, f(\bar{t}), s_{j+1}, \dots s_m)$ does (not) hold under M. Let y such that $M \models f(\bar{t}) = y$. By our definition of F under M, $M \models F(\bar{t}, y)$ with this unique y. Hence $\exists y(F(\bar{t}, y) \land P(s_1, \dots, s_{j-1}, y, s_{j+1}, \dots s_m))$ does (not) hold under M.

For the other direction, suppose $\{\mathrm{F}_{\mathrm{Ax}}(\mathcal{L}), \mathrm{E}_{\mathrm{Ax}}(\mathcal{L})\} \cup \Phi$ is satisfiable. We extend a model M of this set of formulas to a model of $\mathrm{T}^{-1}(\Phi)$ by extending it from the language $\mathrm{T}(\mathcal{L})$ to include $\{=\}$ and $\mathrm{FS}(\mathcal{L})$.

First, let $M \models s = t$ if and only if $M \models E(s,t)$. As M is a model of $\mathrm{E}_{\mathrm{Ax}}(A)$, E is reflexive. Since $M \models \mathrm{Congr}(E)$, we can derive the symmetry of E by the following instantiation of it using reflexivity: $M \models (E(s,t) \land E(s,s)) \supset (E(s,s) \supset E(t,s))$. By symmetry, we have that $M \models E(s,r) \Leftrightarrow E(r,s)$, and hence we can show the transitivity of E by another instance of Congr(E): $M \models (E(s,r) \wedge E(s,t)) \supset (E(s,s) \supset E(r,t))$, i.e. given $E(r,s)$ and $E(s,t)$, we can deduce $E(r,t)$ in M. As these properties directly also apply to $=$ in M, equality is defined properly in M.

Second, let $M \models f(\bar{t}) = s$ if and only if $M \models F_f(\bar{t}, s)$ for all $f \in \mathrm{FS}(\mathcal{L})$. As by assumption M is a model of $\mathrm{F_{Ax}}(A)$, we know that for every $\bar{t}$, some s with $M \models F(\bar{t}, s)$ exists and is uniquely defined. Hence f in M refers to a well-defined function.

Lastly, to show that $M \models \mathrm{T}^{\text{-}1}(\Phi)$, consider that the interpretations of the predicates E and $=$ coincide in M. Furthermore, let B be an occurence of $\exists y(F_f(\bar{t}, y) \wedge P(s_1, \ldots, s_{j-1}, y, s_{j+1}, \ldots s_m))$ for some $f \in \mathrm{FS}(\mathcal{L})$ in Φ. Then by the above definition of f in M, we have that B is in M equivalent to $\exists y f(\bar{t}) = y) \wedge P(s_1, \ldots, s_{j-1}, y, s_{j+1}, \ldots s_m))$, which due to f being a function is equivalent to $M \models P(s_1, \ldots, s_{j-1}, f(\bar{t}), s_{j+1}, \ldots s_m))$.

Similariły, let B be an occurrence of $F_f(\bar{t}, s)$ in Φ. Then by our above definition of f in M, we have that $M \models f(\bar{t}) = s$ iff $M \models B$. □

Corollary 3.8. *Let Φ be a set of first-order formulas. Then Φ is satisfiable if and only if $\mathrm{T_{Ax}}(\Phi)$ is satisfiable.*

Proof. The left-to-right direction is directly given in proposition 3.7. For the other direction, consider that by proposition 3.7, $\mathrm{T}^{\text{-}1}(\mathrm{T}(\Phi))$ is satisfiable and by Lemma 3.3, $\mathrm{T}^{\text{-}1}(\mathrm{T}(\Phi)) = \Phi$. □

3.2 Compuation of interpolants in first-order logic without equality and function symbols

Theorem 3.9. *Let Γ and Δ be sets of first-order clauses such that $\Gamma \cup \Delta$ is unsatisfiable. Then there is an algorithm which computes an interpolant I of Γ and Δ.*

TODO: put some kind of propositional interpolation algorithm here as soon at it is clear which to pick

3.3 Conclusion

Proof of Theorem 1.1 (Interpolation). Since $\Gamma \cup \Delta$ is unsatisfiable, by proposition 3.7, $\mathrm{T_{Ax}}(\Gamma \cup \Delta)$ is unsatisfiable. It follows from Lemma 3.5, that $\mathrm{T_{Ax}}(\Gamma) \cup \mathrm{T_{Ax}}(\Delta)$ is unsatisfiable as well.

Note that $\mathrm{T_{Ax}}(\Gamma) \cup \mathrm{T_{Ax}}(\Delta)$ contains neither function symbols nor the equality symbol.

Hence by Theorem 3.9, there is an interpolant I such that

1. $\mathrm{T_{Ax}}(\Gamma) \models I$
2. $\mathrm{T_{Ax}}(\Delta) \models \neg I$
3. $\mathrm{L}(I) \subseteq \mathrm{L}(\mathrm{T_{Ax}}(\Gamma)) \cap \mathrm{L}(\mathrm{T_{Ax}}(\Delta))$

$\mathrm{T}_{\mathrm{Ax}}(\Gamma) \models I$ is equivalent to $\mathrm{T}_{\mathrm{Ax}}(\Gamma) \cup \{\neg I\}$ being unsatisfiable. Through the unfolding of $\mathrm{T}_{\mathrm{Ax}}(\Gamma)$, we get that $\{\mathrm{F}_{\mathrm{Ax}}(\mathrm{L}(\Gamma)), \mathrm{E}_{\mathrm{Ax}}(\mathrm{L}(\Gamma))\} \cup \mathrm{T}(\Gamma) \cup \{\neg I\}$ is unsatisfiable.

Since $\mathrm{L}(\neg I) \subseteq \mathrm{L}(\mathrm{T}_{\mathrm{Ax}}(\Gamma))$, we can apply Proposition 3.7 by considering $\mathrm{T}(\Gamma) \cup \{\neg I\}$ as Φ to conclude that $\mathrm{T}^{\text{-1}}(\mathrm{T}(\Gamma) \cup \{\neg I\})$ is unsatisfiable. By pulling $\mathrm{T}^{\text{-1}}$ inward and an application of Lemma 3.3, we get that $\Gamma \cup \{\mathrm{T}^{\text{-1}}(\neg I)\} = \Gamma \cup \{\neg\, \mathrm{T}^{\text{-1}}(I)\}$ is unsatisfiable.

Therefore $\Gamma \models \mathrm{T}^{-1}(I)$.

TODO: by similar argument, Δ; language of I is right one $\square$

CHAPTER 4

Proofs

4.1 WT: Interpolation extraction in one pass

easy for constants, just as in huang but in one pass

terms can grow unpredictably, order cannot be determined during pass

4.2 WT: Interpolation extraction in two passes

4.2.1 huang proof revisited

propositional part

Let $\Gamma \cup \Delta$ be unsatisfiable. Let π be a proof of the empty clause from $\Gamma \cup \Delta$. Then PI is a function that returns a interpolant with respect to the current clause.

Definition 4.1. θ is a *propositional interpolant* with respect to a clause C in a resolution refutation π of $\Gamma \cup \Delta$ if

1. $\Gamma \models \theta \vee C$
2. $\Delta \models \neg\theta \vee C$
3. $\mathrm{PS}(\theta) \subseteq (\mathrm{PS}(\Gamma) \cap \mathrm{PS}(\Delta)) \cup \{\top, \bot\}$. △

The third condition will sometimes be referred to as *language restriction*. It is easy to see that a propositional interpolant with respect to □ is a propositional interpolant, i.e. it is an interpolant without the language restriction on constant, variable and function symbols.

add this to the definition, i.e. possible define rel prop interpol from prop interpol

We proceed by defining a procedure PI which extracts propositional interpolants from a resolution refutation.

Definition 4.2. PI is defined as follows:

Base case. If $C \in \Gamma$, $\mathrm{PI}(C) = \bot$. If otherwise $C \in \Delta$, $\Delta(C) = \top$.

Resolution. Suppose the clause C is the result of a resolution step. Then it has the following form:

If the clause C is the result of a resolution step of $C_1 : D \vee l$ and $C_2 : E \vee \neg l'$ using a unifier σ such that $l\sigma = l'\sigma$, then $\mathrm{PI}(C)$ is defined as follows:

1. If $\mathrm{PS}(l) \in \mathrm{L}(\Gamma) \setminus \mathrm{L}(\Delta)$:$\mathrm{PI}(C) = [\mathrm{PI}(C_1) \vee \mathrm{PI}(C_2)]\sigma$
2. If $\mathrm{PS}(l) \in \mathrm{L}(\Delta) \setminus \mathrm{L}(\Gamma)$: $\mathrm{PI}(C) = [\mathrm{PI}(C_1) \wedge \mathrm{PI}(C_2)]\sigma$
3. If $\mathrm{PS}(l) \in \mathrm{L}(\Gamma) \cap \mathrm{L}(\Delta)$: $\mathrm{PI}(C) = [(l \wedge \mathrm{PI}(C_2)) \vee (l' \wedge \mathrm{PI}(C_1))]\sigma$

change to "is Γ-colored?"

Factorisation. If the clause C is the result of a factorisation of $C_1 : l \vee l' \vee D$ using a unifier σ such that $l\sigma = l'\sigma$, then $\mathrm{PI}(C) = \mathrm{PI}(C_1)\sigma$.

Paramodulation. If the clause C is the result of a paramodulation of $C_1 : s = t \vee C$ and $C_2 : D[r]$ using a unifier σ such that $r\sigma = s\sigma$, then $\mathrm{PI}(C)$ is defined according to the following case distinction:

1. If r occurs in a maximal Δ-term $h(r)$ in $D[r]$ and $h(r)$ occurs more than once in $D[r] \vee \mathrm{PI}(D[r])$:
$\mathrm{PI}(C) = [(s = t \wedge \mathrm{PI}(C_2)) \vee (s \neq t \wedge \mathrm{PI}(C_1))]\sigma \vee (s = t \wedge h(s) \neq h(t))$
2. If r occurs in a maximal Γ-term $h(r)$ in $D[r]$ and $h(r)$ occurs more than once in $D[r] \vee \mathrm{PI}(D[r])$:
$\mathrm{PI}(C) = [(s = t \wedge \mathrm{PI}(C_2)) \vee (s \neq t \wedge \mathrm{PI}(C_1))]\sigma \wedge (s \neq t \vee h(s) = h(t))$
3. Otherwise:
$\mathrm{PI}(C) = [(s = t \wedge \mathrm{PI}(C_2)) \vee (s \neq t \wedge \mathrm{PI}(C_1))]\sigma$ $\triangle$

Proposition 4.3. *Let C be a clause of a resolution refutation. Then* $\mathrm{PI}(C)$ *is a propositional interpolant with respect to C.*

Proof. Proof by induction on the number of rule applications including the following strenghtenings: $\Gamma \models \mathrm{PI}(C) \vee C_\Gamma$ and $\Delta \models \neg\,\mathrm{PI}(C) \vee C_\Delta$, where D_Φ denotes the clause D with only the literals which are contained in $\mathrm{L}(\Phi)$. They clearly imply conditions 1 and 2 of definition 4.1.

Base case. Suppose no rules were applied. We distinguish two possible cases:

1. $C \in \Gamma$. Then $\mathrm{PI}(C) = \bot$. Clearly $\Gamma \models \bot \vee C_\Gamma$ as $C_\Gamma = C \in \Gamma$, $\Delta \models \neg\bot \vee C_\Delta$ and $\bot$ satisfies the restriction on the language.
2. $C \in \Delta$. Then $\mathrm{PI}(C) = \top$. Clearly $\Gamma \models \top \vee C_\Gamma$, $\Delta \models \neg\top \vee C_\Delta$ as $C_\Delta = C \in \Delta$ and $\top$ satisfies the restriction on the language.

Suppose the property holds for n rule applications. We show that it holds for $n+1$ applications by considering the last one:

Resolution. Suppose the last rule application is an instance of resolution. Then it is of the form:

$$\frac{C_1 : D \vee l \qquad C_2 : E \vee \neg l'}{C : (D \vee E)\sigma} \quad l\sigma = l'\sigma$$

By the induction hypothesis, we can assume that:

$\Gamma \models \mathrm{PI}(C_1) \vee (D \vee l)_\Gamma$

$\Delta \models \neg\,\mathrm{PI}(C_1) \vee (D \vee l)_\Delta$

$\Gamma \models \mathrm{PI}(C_2) \vee (E \vee \neg l')_\Gamma$

$\Delta \models \neg\,\mathrm{PI}(C_2) \vee (E \vee \neg l')_\Delta$

We consider the respective cases from definition 4.2:

1. $\mathrm{PS}(l) \in \mathrm{L}(\Gamma) \setminus \mathrm{L}(\Delta)$: Then $\mathrm{PI}(C) = [\mathrm{PI}(C_1) \vee \mathrm{PI}(C_2)]\sigma$.

   As $\mathrm{PS}(l) \in \mathrm{L}(\Gamma)$, $\Gamma \models (\mathrm{PI}(C_1) \vee D_\Gamma \vee l)\sigma$ as well as $\Gamma \models (\mathrm{PI}(C_2) \vee E_\Gamma \vee \neg l')\sigma$. By a resolution step, we get $\Gamma \models (\mathrm{PI}(C_1) \vee \mathrm{PI}(C_2))\sigma \vee ((D \vee E)\sigma)_\Gamma$.

   Furthermore, as $\mathrm{PS}(l) \notin \mathrm{L}(\mathrm{PI})$, $\Delta \models (\neg\,\mathrm{PI}(C_1) \vee D_\Delta)\sigma$ as well as $\Delta \models (\neg\,\mathrm{PI}(C_2) \vee E_\Delta)\sigma$. Hence it certainly holds that $\Delta \models (\neg\,\mathrm{PI}(C_1) \vee \neg\,\mathrm{PI}(C_2))\sigma \vee (D \vee E)\sigma_\Delta$.

   The language restriction clearly remains satisfied as no non-logical symbols are added.

2. $\mathrm{PS}(l) \in \mathrm{L}(\Delta) \setminus \mathrm{L}(\Gamma)$: Then $\mathrm{PI}(C) = [\mathrm{PI}(C_1) \wedge \mathrm{PI}(C_2)]\sigma$.

   As $\mathrm{PS}(l) \notin \mathrm{L}(\Gamma)$, $\Gamma \models (\mathrm{PI}(C_1) \vee D_\Gamma)\sigma$ as well as $\Gamma \models (\mathrm{PI}(C_2) \vee E_\Gamma)\sigma$. Suppose that in a model M of Γ, $M \not\models D_\Gamma$ and $M \not\models E_\Gamma$. Then $M \models \mathrm{PI}(C_1) \wedge \mathrm{PI}(C_2)$. Hence $\Gamma \models (\mathrm{PI}(C_1) \wedge \mathrm{PI}(C_2))\sigma \vee ((D \vee E)\sigma)_\Gamma$.

   Furthermore due to $\mathrm{PS}(l) \in \mathrm{L}(\Delta)$, $\Delta \models (\neg\,\mathrm{PI}(C_1) \vee D_\Delta \vee l)\sigma$ as well as $\Delta \models (\neg\,\mathrm{PI}(C_2) \vee E_\Delta \vee \neg l')\sigma$. By a resolution step, we get $\Delta \models (\neg\,\mathrm{PI}(C_1) \vee \neg\,\mathrm{PI}(C_2))\sigma \vee (D_\Delta \vee E_\Delta)\sigma$ and hence $\Delta \models \neg(\mathrm{PI}(C_1) \wedge \mathrm{PI}(C_2))\sigma \vee (D_\Delta \vee E_\Delta)\sigma$.

   The language restriction again remains intact.

3. $\mathrm{PS}(l) \in \mathrm{L}(\Delta) \cap \mathrm{L}(\Gamma)$: Then $\mathrm{PI}(C) = [(l \wedge \mathrm{PI}(C_2)) \vee (\neg l' \wedge \mathrm{PI}(C_1))]\sigma$

   First, we have to show that $\Gamma \models [(l \wedge \mathrm{PI}(C_2)) \vee (l' \wedge \mathrm{PI}(C_1))]\sigma \vee ((D \vee E)\sigma)_\Gamma$. Suppose that in a model M of Γ, $M \not\models D_\Gamma$ and $\Gamma \not\models E$. Otherwise we are done. The induction assumtion hence simplifies to $M \models \mathrm{PI}(C_1) \vee l$ and $M \models \mathrm{PI}(C_2) \vee \neg l'$ respectively. As $l\sigma = l'\sigma$, by a case distinction argument on the truth value of $l\sigma$, we get that either $M \models (l \wedge \mathrm{PI}(C_2))\sigma$ or $M \models (\neg l' \wedge \mathrm{PI}(C_1))\sigma$.

   Second, we show that $\Delta \models ((l \vee \neg\,\mathrm{PI}(C_1)) \wedge (\neg l' \vee \neg\,\mathrm{PI}(C_2)))\sigma \vee ((D \vee E)\sigma)_\Delta$. Suppose again that in a model M of Δ, $M \not\models D_\Delta$ and $\Gamma \not\models E_\Delta$. Then the required statement follows from the induction hypothesis.

   The language condition remains satisfied as only the common literal l is added to the interpolant.

Factorisation. Suppose the last rule application is an instance of factorisation. Then it is of the form:

$$\frac{C_1 : l \vee l' \vee D}{C_1 : (l \vee D)\sigma} \quad \sigma = \mathrm{mgu}(l, l')$$

Then the propositional interpolant $\mathrm{PI}(C)$ is defined as $\mathrm{PI}(C_1)$. By the induction hypothesis, we have:

$\Gamma \models \mathrm{PI}(C_1) \vee (l \vee l' \vee D)_\Gamma$

$\Delta \models \mathrm{PI}(C_1) \vee (l \vee l' \vee D)_\Delta$

It is easy to see that then also:

$\Gamma \models (\mathrm{PI}(C_1) \vee (l \vee D)_\Gamma)\sigma$

$\Delta \models (\mathrm{PI}(C_1)\sigma \vee (l \vee D)_\Delta)\sigma$

The restriction on the language trivially remains intract.

Paramodulation. Suppose the last rule application is an instance of paramodulation. Then it is of the form:

$$\frac{C_1 : D \vee s = t \qquad C_2 : E[r]}{C : (D \vee E[t])\sigma} \quad \sigma = \mathrm{mgu}(s, r)$$

By the induction hypothesis, we have:

$\Gamma \models \mathrm{PI}(C_1) \vee (D \vee s = t)_\Gamma$

$\Delta \models \neg\,\mathrm{PI}(C_1) \vee (D \vee s = t)_\Delta$

$\Gamma \models \mathrm{PI}(C_2) \vee (E[r])_\Gamma$

$\Delta \models \neg\,\mathrm{PI}(C_2) \vee (E[r])_\Delta$

First, we show that $\mathrm{PI}(C)$ as constructed in case 3 of the definition is a propositional interpolant in any of these cases:

$\mathrm{PI}(C) = (s = t \wedge \mathrm{PI}(C_2)) \vee (s \neq t \wedge \mathrm{PI}(C_1))$

Suppose that in a model M of Γ, $M \not\models D\sigma$ and $M \not\models E[t]\sigma$. Otherwise we are done. Furthermore, assume that $M \models (s = t)\sigma$. Then $M \not\models E[r]\sigma$, but then necessarily $M \models \mathrm{PI}(C_2)\sigma$.
On the other hand, suppose $M \models (s \neq t)\sigma$. As also $M \not\models D\sigma$, $M \models \mathrm{PI}(C_1)\sigma$. Consequently, $M \models [(s = t \wedge \mathrm{PI}(C_2)) \vee (s \neq t \wedge \mathrm{PI}(C_1))]\sigma \vee [(D \vee E)_\Gamma]\sigma$

By an analogous argument, we get $\Delta \models [(s = t \wedge \neg\,\mathrm{PI}(C_2)) \vee (s \neq t \wedge \neg\,\mathrm{PI}(C_1))]\sigma \vee [(D \vee E)_\Delta]\sigma$, which implies $\Delta \models [(s \neq t \vee \neg\,\mathrm{PI}(C_2)) \wedge (s = t \vee \neg\,\mathrm{PI}(C_1))]\sigma \vee ((D \vee E)_\Delta)\sigma$

The language restriction again remains satisfied as the only predicate, that is added to the interpolant, is =.

This concludes the argumentation for case 3.

The interpolant of case 1 differs only by an additional formula added via a disjunction and hence condition 1 of definition 4.1 holds by the above reasoning. As the adjoined formula is a contradiction, its negation is valid which in combination with the above reasoning establishes condition 2. Since no new predicated are added, the language condition remains intact.

The situation in case 2 is somewhat symmetric: As a tautology is added to the interpolant with respect to case 1, condition 1 is satisfied by the above reasoning. For condition 2, consider that the negated interpolant of case 1 implies the negated interpolant of this case. The language condition again remains intact. □

Before we are able to specify a procedure to transform the propositional interpolant generated by PI into a proper interpolant without any colored terms, we define a simpler but equally powerful form of resolution refutations.

Definition 4.4. A resolution refutation is a *tree refutation* if every clause is used at most once. △

The following lemma shows that this form does not restrict the calculus given that we allow multisets as inital clause sets.

Lemma 4.5. *Every resolution refutation can be transformed into a tree refutation.*

Proof. Let π be a resolution refutation of Φ. We prove that π can be transformed into a tree refutation by induction on the number of clauses, that are used multiple times.

Suppose that no clauses are used more than once in π. Then π is a tree refutation.

Otherwise let Ψ be the set of clauses which is used multiple times. Let $C \in \Psi$ be such that no clause $D \in \Psi$ is used in the derivation leading to C. Let χ be the derivation leading to C.

Suppose C is used m times. We create another resolution refutation π' from π which contains m copies of χ and replaces the ith use of the clause C by the final clause of the ith copy of χ, $1 \leq i \leq m$. In order to ensure that the sets of variables of the input clauses are disjoint, we rename the variables in each copy of χ and adapt π' accordingly. Hence π' is a resolution refutation of Φ where $m - 1$ clauses are used more than once. □

In a tree refutation where the input clauses have a disjoint sets of variables, every variable has a unique ancestor which traces back to an input clause and hence appears only along a certain path. This insight allows us to push substitutions of the variables upwards along this path and arrive at the following definition and lemma:

Definition 4.6. A resolution refutation is a *propositional refutation* if no nontrivial substitutions are employed. △

Lemma 4.7. *Let Φ be unsatisfiable. Then there is a propositional refutation of Φ which starts from instances of Φ.*

Proof. Let π be a resolution refutation of Φ. By Lemma 4.5, we can assume without loss of generality that π is a tree refutation where the sets of variables of the input clauses are disjoint. Furthermore, we can assume that only most general unifiers are employed in π.

Then any unifier in π is either trivial on x or there is one unique unifier σ with $x\sigma = t$ where x does not occur in t. Hence along the path through the deduction where x occurs, it remains unchanged. Therefore we can create a new resolution refutation π' from π where x is replaced by t. Clearly π' is rooted in instances of Φ.

By application of this procedure to all variable occurring in π, we obtain a desired resolution refutation. □

Even though propositional refutations have some nice properties for theoretical analysis, their use in practise is not desired as its construction involves a considerable blowup of the refutation. But still its use is justified in this instance as we can show for arbitrary refutations π that the algorithm stated in 4.2 gives closely related results for both π and its corresponding propositional refutation.

Lemma 4.8. *Let π be a resolution refutation of Φ and π' a propositional refutation corresponding to π. Then for every clause C in π and its corresponding clause C' in π', $\mathrm{PI}(C)\sigma = \mathrm{PI}(C')$ where σ is the composition of unifications of π applied to the variables occurring in C.*

Proof. For the construction of the propositional skeleton of $\mathrm{PI}(\cdot)$ only the coloring of the clauses is relevant and since this is the same in both π and π', it coincides for $\mathrm{PI}(C)$ and $\mathrm{PI}(C')$.

Hence $\mathrm{PI}(C)$ and $\mathrm{PI}(C')$ differ only in their term structure. To be more specific, in $\mathrm{PI}(C')$, the composition of substitutions that are applied in π have already been applied to the initial clauses of π'. Note that substitution commutes with the rules of substitution. Therefore the only difference between $\mathrm{PI}(C)$ and $\mathrm{PI}(C')$ is that at certain term positions, there are variables in $\mathrm{PI}(C)$ where in $\mathrm{PI}(C')$ by some substitution a different term is located. But these substitutions are certainly applied by σ, hence $\mathrm{PI}(C)\sigma = \mathrm{PI}(C')$. □

This establishes the theoretical framework which is required to define and show the correctness of a procedure to construct a proper interpolant from the propositional interpolant. The idea of this procedure will be to replace grey terms still occurring in the propositional interpolant with variables and quantifying them appropriately. First, we define the lifting of the maximal Δ-terms:

Definition 4.9. Let Γ and Δ be sets of first-order formulas, ϕ a formula or a term, $t_1, \ldots, t_n$ the maximal Φ-terms for $\Phi \in \{\Gamma, \Delta\}$ in ϕ and $x_1, \ldots, x_n$ fresh variables. Then $\mathrm{Lift}_{\Phi,x}(\phi)$ denotes $\phi\{t_1/x_1\} \ldots \{t_n/x_n\}$. △

Lemma 4.10. *Let π be a resolution refutation of $\Gamma \cup \Delta$. Then $\Gamma \models \mathrm{Lift}_{\Delta,x}(\mathrm{PI}(C) \vee C)$ for C in π.*

Proof. □

proof that we are allowed to overbind
TODO: define procedure
TODO: proof

overbinding

Algorithm (input: propositional interpolant θ):

1. Let $t_1, \dots, t_n$ be the maximal occurrences of noncommon terms in θ. Order t_i ascendingly by term size.

2. Let θ^* be θ with maximal occurrences of Δ-terms $r_1, \dots, r_k$ replaced by fresh variables $x_1, \dots, x_k$ and maximal occurrences of Γ-terms $s_1, \dots, s_{n-k}$ by fresh variables $x_{k+1}, \dots, x_n$

3. Return $Q_1 x_1, \dots Q_n x_n \theta^*$, where Q_i is $\forall$ if t_i is a Δ-term and $\exists$ otherwise.

Language condition easily established. To prove:
$\Gamma \models Q_1 x_1, \dots Q_n x_n \theta^*$
$\Delta \models \neg Q_1 x_1, \dots Q_n x_n \theta^*$
We know that θ works, just the terms are missing.

4.2.2 final step of huang's proof

Theorem 4.11. *$Q_1 z_1 \dots Q_n z_n \, \mathrm{PI}(\square)^*(z_1, \dots, z_n)$ is a craig interpolant (order as in huang).*

Proof. By Lemma **??**, $\Gamma \models \forall x_1 \dots \forall x_n \overline{\mathrm{PI}(\square)}(x_1, \dots, x_n)$.

The terms in $\overline{PI(\square)}$ are either among the x_i, $1 \leq i \leq n$ or grey terms or Γ-terms. Let t be a maximal Γ-term in $\overline{\mathrm{PI}(\square)}$. Then it is of the form $f(x_{i_1}, \dots, x_{i_{n_x}}, u_1, \dots, u_{n_u}, v_1, \dots, v_{n_v})$, where f is Γ-colored, the x_j are as before, the u_j are grey terms and the v_j are Γ-terms. Note that the Δ-terms, which are replaced by the $x_{i_1}, \dots, x_{i_{n_x}}$ are of strictly smaller size than t as they are "strict" subterms of t.

basically only need the x_j

In $\mathrm{PI}(\square)^*$, t will be replaced by some z_j, which is existentially quantified. For this z_j, t is a witness as due to the quantifier ordering, all the $x_{i_1}, \dots, x_{i_{n_x}}$ will be quantified before the existential quantification of z_j. Therefore $\Gamma \models Q_1 z_1 \dots Q_n z_n \, \mathrm{PI}(\square)^*(z_1, \dots, z_n)$.

TODO: Δ $\square$

Bibliography

[BJ13] Maria Paola Bonacina and Moa Johansson. On interpolation in automated theorem proving. Technical Report 86/2012, Dipartimento di Informatica, Università degli Studi di Verona, 2013. Submitted to journal August 2013.

[BL11] Matthias Baaz and Alexander Leitsch. *Methods of Cut-Elimination*. Trends in Logic. Springer, 2011.

[CK90] C.C. Chang and H.J. Keisler. *Model Theory*. Studies in Logic and the Foundations of Mathematics. Elsevier Science, 1990.

[Cra57a] William Craig. Linear Reasoning. A New Form of the Herbrand-Gentzen Theorem. *The Journal of Symbolic Logic*, 22(3):250–268, September 1957.

[Cra57b] William Craig. Three Uses of the Herbrand-Gentzen Theorem in Relating Model Theory and Proof Theory. *The Journal of Symbolic Logic*, 22(3):269–285, September 1957.

[Hua95] Guoxiang Huang. Constructing Craig Interpolation Formulas. In *Proceedings of the First Annual International Conference on Computing and Combinatorics*, COCOON '95, pages 181–190, London, UK, UK, 1995. Springer-Verlag.

[Kle67] Stephen Cole Kleene. *Mathematical logic*. Wiley, New York, NY, 1967.

[Kra97] Jan Krajíček. Interpolation Theorems, Lower Bounds for Proof Systems, and Independence Results for Bounded Arithmetic. *Journal of Symbolic Logic*, pages 457–486, 1997.

[Lyn59] Roger C. Lyndon. An interpolation theorem in the predicate calculus. *Pacific Journal of Mathematics*, 9(1):129–142, 1959.

[McM03] Kenneth L. McMillan. Interpolation and SAT-Based Model Checking. In Jr. Hunt, Warren A. and Fabio Somenzi, editors, *Computer Aided Verification*, volume 2725 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin Heidelberg, 2003.

[Pud97] Pavel Pudlák. Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations. *J. Symb. Log.*, 62(3):981–998, 1997.

[Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, January 1965.

[Sho67] Joseph R. Shoenfield. *Mathematical logic.* Addison-Wesley series in logic. Addison-Wesley Pub. Co., 1967.

[Sla70] James R. Slagle. Interpolation theorems for resolution in lower predicate calculus. *J. ACM*, 17(3):535–542, July 1970.

[Tak87] Gaisi Takeuti. *Proof Theory.* Studies in logic and the foundations of mathematics. North-Holland, 1987.

[Wei10] Georg Weissenbacher. *Program Analysis with Interpolants.* PhD thesis, 2010.