Lecture 2:
First-Order Logic -
Syntax, Semantics, Resolution

Ruzica Piskac

Ecole Polytechnique Fédérale de Lausanne, Switzerland
ruzica.piskac@epfl.ch

Seminar on Automated Reasoning 2010

Acknowledgments

These slides were originally developed by Harald Ganzinger (1950 - 2004)

`http://www.mpi-inf.mpg.de/~hg/`

Part 1: First-Order Logic

- formalizes fundamental mathematical concepts
- expressive (Turing-complete)
- not too expressive (not axiomatizable: natural numbers, uncountable sets)
- rich structure of decidable fragments
- rich model and proof theory

First-order logic is also called (first-order) predicate logic.

1.1 Syntax

- non-logical symbols (domain-specific)
  $\Rightarrow$ terms, atomic formulas
- logical symbols (domain-independent)
  $\Rightarrow$ Boolean combinations, quantifiers

Signature

Usage: fixing the alphabet of non-logical symbols

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

where

- Ω a set of function symbols f with arity $n \geq 0$, written f/n,
- Π a set of predicate symbols p with arity $m \geq 0$, written p/m.

If $n = 0$ then f is also called a constant (symbol). If $m = 0$ then p is also called a propositional variable. We use letters P, Q, R, S, to denote propositional variables.

Refined concept for practical applications: many-sorted signatures (corresponds to simple type systems in programming languages); not so interesting from a logical point of view

Variables

Predicate logic admits the formulation of abstract, schematic assertions. (Object) variables are the technical tool for schematization. We assume that

$$X$$

is a given countably infinite set of symbols which we use for (the denotation of) variables.

Terms

Terms over Σ (resp., Σ-terms) are formed according to these syntactic rules:

$$\begin{array}{llll} s,t,u,v & ::= & x & , x \in X & \text{(variable)} \\ & \mid & f(s_1,\ldots,s_n) & , f/n \in \Omega & \text{(functional term)} \end{array}$$

By $T_\Sigma(X)$ we denote the set of Σ-terms (over X). A term not containing any variable is called a ground term. By T_Σ we denote the set of Σ-ground terms.

In other words, terms are formal expressions with well-balanced brackets which we may also view as marked, ordered trees. The markings are function symbols or variables. The nodes correspond to the subterms of the term. A node v that is marked with a function symbol f of arity n has exactly n subtrees representing the n immediate subterms of v.

Atoms

Atoms (also called atomic formulas) over Σ are formed according to this syntax:

$$\begin{array}{llll} A, B & ::= & p(s_1, \ldots, s_m) & , p/m \in \Pi \\ \Big[& \mid & (s \approx t) & \text{(equation)} \quad \Big] \end{array}$$

Whenever we admit equations as atomic formulas we are in the realm of first-order logic with equality. Admitting equality does not really increase the expressiveness of first-order logic, (cf. exercises). But deductive systems where equality is treated specifically can be much more efficient.

Literals

$$
\begin{array}{rcll}
L & ::= & A & \text{(positive literal)} \\
 & \mid & \neg A & \text{(negative literal)}
\end{array}
$$

Clauses

$$\begin{array}{lcll} C, D & ::= & \bot & \text{(empty clause)} \\ & \mid & L_1 \vee \ldots \vee L_k,\ \ k \geq 1 & \text{(non-empty clause)} \end{array}$$

General First-Order Formulas

$F_\Sigma(X)$ is the set of first-order formulas over Σ defined as follows:

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

Notational Conventions

- We omit brackets according to the following rules:
  - $\neg \quad >_p \quad \vee \quad >_p \quad \wedge \quad >_p \quad \Longrightarrow \quad >_p \quad \equiv$ (binding precedences)
  - $\vee$ and $\wedge$ are associative and commutative
  - $\Longrightarrow$ is right-associative
- $Qx_1, \ldots, x_n\, F$ abbreviates $Qx_1 \ldots Qx_n\, F$.
- infix-, prefix-, postfix-, or mixfix-notation with the usual operator precedences; examples:

$$\begin{array}{rcc} s + t * u & \text{for} & +(s, *(t, u)) \\ s * u \leq t + v & \text{for} & \leq(*(s, u), +(t, v)) \\ -s & \text{for} & -(s) \\ 0 & \text{for} & 0() \end{array}$$

Example: Peano Arithmetic

$$
\begin{aligned}
\Sigma_{PA} &= (\Omega_{PA},\ \Pi_{PA}) \\
\Omega_{PA} &= \{0/0,\ +/2,\ */2,\ s/1\} \\
\Pi_{PA} &= \{\leq/2,\ </2\}
\end{aligned}
$$

$+, *, <, \leq$ infix; $* >_p + >_p < >_p \leq$

Exampes of formulas over this signature are:

$\forall x, y(x \leq y \equiv \exists z(x + z \approx y))$

$\exists x \forall y(x + y \approx y)$

$\forall x, y(x * s(y) \approx x * y + x)$

$\forall x, y(s(x) \approx s(y) \implies x \approx y)$

$\forall x \exists y\ (x < y \land \neg \exists z(x < z \land z < y))$

Remarks About the Example

We observe that the symbols $\leq, <, 0, s$ are redundant as they can be defined in first-order logic with equality just with the help of $+$. The first formula defines $\leq$, while the second defines zero. The last formula, respectively, defines s.

Eliminating the existential quantifiers by Skolemization (cf. below) reintroduces the “redundant” symbols.

Consequently there is a trade-off between the complexity of the quantification structure and the complexity of the signature.

Bound and Free Variables

In $QxF,\ Q \in \{\exists, \forall\}$, we call F the scope of the quantifier Qx. An occurrence of a variable x is called bound, if it is inside the scope of a quantifier Qx. Any other occurrence of a variable is called free.
Formulas without free variables are also called closed formulas or sentential forms.
Formulas without variables are called ground.

Example

$$\forall y \quad \overbrace{(\forall \overbrace{x \quad p(x)}^{scope} \quad \implies \quad q(x, y))}^{scope}$$

The occurrence of y is bound, as is the first occurrence of x. The second occurrence of x is a free occurrence.

Substitutions

Substitution is a fundamental operation on terms and formulas that occurs in all inference systems for first-order logic. In the presence of quantification it is surprisingly complex.

By $F[s/x]$ we denote the result of substituting all free occurrences of x in F by the term s.

Formally we define $F[s/x]$ by structural induction over the syntactic structure of F by the equations depicted on the next page.

Substitution of a Term for a Free Variable

$$\begin{aligned}
x[s/x] &= s \\
x'[s/x] &= x' \;;\;\; \text{if } x' \neq x \\
f(s_1, \ldots, s_n)[s/x] &= f(s_1[s/x], \ldots, s_n[s/x]) \\
\bot[s/x] &= \bot \\
\top[s/x] &= \top \\
p(s_1, \ldots, s_n)[s/x] &= p(s_1[s/x], \ldots, s_n[s/x]) \\
(u \approx v)[s/x] &= (u[s/x] \approx v[s/x]) \\
\neg F[s/x] &= \neg(F[s/x]) \\
(F \rho G)[s/x] &= (F[s/x] \rho G[s/x]) \;;\;\; \text{for each binary connective } \rho \\
(QyF)[s/x] &= Qz((F[z/y])[s/x]) \;;\; \text{with } z \text{ a “fresh” variable}
\end{aligned}$$

Why Substitution is Complicated

We need to make sure that the (free) variables in s are not *captured* upon placing s into the scope of a quantifier, hence the renaming of the bound variable y into a “fresh”, that is, previously unused, variable z. Why this definition of substitution is well-defined will be discussed below.

General Substitutions

In general, substitutions are mappings

$$\sigma : X \to T_\Sigma(X)$$

such that the domain of σ, that is, the set

$$dom(\sigma) = \{x \in X \mid \sigma(x) \neq x\},$$

is finite. The set of variables introduced by σ, that is, the set of variables occurring in one of the terms $\sigma(x)$, with $x \in dom(\sigma)$, is denoted by $codom(\sigma)$.

Substitutions are often written as $[s_1/x_1, \ldots, s_n/x_n]$, with x_i pairwise distinct, and then denote the mapping

$$[s_1/x_1, \ldots, s_n/x_n](y) = \begin{cases} s_i, & \text{if } y = x_i \\ y, & \text{otherwise} \end{cases}$$

We also write $x\sigma$ for $\sigma(x)$.

Modifying a Substitution

The modification of a substitution σ at x is defined as follows:

$$\sigma[x \mapsto t](y) = \begin{cases} t, & \text{if } y = x \\ \sigma(y), & \text{otherwise} \end{cases}$$

Application of a Substitution

“Homomorphic” extension of σ to terms and formulas:

$$\begin{aligned}
f(s_1,\ldots,s_n)\sigma &= f(s_1\sigma,\ldots,s_n\sigma)\\
\bot\sigma &= \bot\\
\top\sigma &= \top\\
p(s_1,\ldots,s_n)\sigma &= p(s_1\sigma,\ldots,s_n\sigma)\\
(u \approx v)\sigma &= (u\sigma \approx v\sigma)\\
\neg F\sigma &= \neg(F\sigma)\\
(F\rho G)\sigma &= (F\sigma\,\rho\,G\sigma)\ ; \quad \text{for each binary connective } \rho\\
(Qx\,F)\sigma &= Qz\,(F\,\sigma[x \mapsto z])\ ; \quad \text{with } z \text{ a fresh variable}
\end{aligned}$$

E: Convince yourself that for the special case $\sigma = [t/x]$ the new definition coincides with our previous definition (modulo the choice of fresh names for the bound variables).

Structural Induction

Theorem

*Let $G = (N, T, P, S)$ be a context-free grammar[a] and let q be a property of T^* (the words over the alphabet T of terminal symbols of G).
q holds for all words $w \in L(G)$, whenever one can prove these 2 properties:*

1. *(base cases)*
   $q(w')$ holds for each $w' \in T^*$ such that $X ::= w'$ is a rule in P.
2. *(step cases)*
   If $X ::= w_0 X_0 w_1 \ldots w_n X_n w_{n+1}$ is in P with $X_i \in N$, $w_i \in T^*$, $n \geq 0$, then for all $w_i' \in L(G, X_i)$, whenever $q(w_i')$ holds for $0 \leq i \leq n$, then also $q(w_0 w_0' w_1 \ldots w_n w_n' w_{n+1})$ holds.

Here $L(G, X_i) \subseteq T^*$ denotes the language generated by the grammar G from the nonterminal X_i.

[a] Infinite grammars are also admitted.

Structural Recursion

Theorem

Let $G = (N, T, P, S)$ be a unambiguous context-free grammar. A function f is well-defined on $L(G)$ (that is, unambiguously defined) whenever these 2 properties are satisfied:

1. *(base cases)*
   f is well-defined on the words $w' \in \Sigma^*$ for each rule $X ::= w'$ in P.
2. *(step cases)*
   If $X ::= w_0 X_0 w_1 \ldots w_n X_n w_{n+1}$ is a rule in P then $f(w_0 w_0' w_1 \ldots w_n w_n' w_{n+1})$ is well-defined, assuming that each of the $f(w_i')$ is well-defined.

Q: Why should G be unambigous?

Substitution Revisited

Q: Does Theorem 2 justify that our homomorphic extension

$$apply : F_\Sigma(X) \times (X \to T_\Sigma(X)) \quad \to \quad F_\Sigma(X),$$

with $apply(F, \sigma)$ denoted by $F\sigma$, of a substitution is well-defined?
A: We have two problems here. One is that "fresh" is (deliberately) left unspecified. That can be easily fixed by adding an extra variable counter argument to the apply function.
The second problem is that Theorem 2 applies to unary functions only. The standard solution to this problem is to curryfy, that is, to consider the binary function as a unary function producing a unary (residual) function as a result:

$$apply : F_\Sigma(X) \quad \to \quad ((X \to T_\Sigma(X)) \to F_\Sigma(X))$$

where we have denoted $(apply(F))(\sigma)$ as $F\sigma$.
E: Convince yourself that this does the trick.

1.2. Semantics

To give semantics to a logical system means to define a notion of truth for the formulas. The concept of truth that we will now define for first-order logic goes back to Tarski.
In classical logic (dating back to Aristoteles) there are “only” two truth values “true” and “false” which we shall denote, respectively, by 1 and 0.
There are multi-valued logics having more than two truth values.

Structures

A Σ-algebra (also called Σ-interpretation or Σ-structure) is a triple

$$\mathcal{A} = (U,\ (f_{\mathcal{A}} : U^n \to U)_{f/n \in \Omega},\ (p_{\mathcal{A}} \subseteq U^m)_{p/m \in \Pi})$$

where $U \neq \emptyset$ is a set, called the universe of $\mathcal{A}$.
Normally, by abuse of notation, we will have $\mathcal{A}$ denote both the algebra and its universe.
By Σ-Alg we denote the class of all Σ-algebras.

Assignments

A variable has no intrinsic meaning. The meaning of a variable has to be defined externally (explicitly or implicitly in a given context) by an assignment.

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

Variable assignments are the semantic counterparts of substitutions.

Value of a Term in $\mathcal{A}$ with Respect to β

By structural induction we define

$$\mathcal{A}(\beta) : T_\Sigma(X) \to \mathcal{A}$$

as follows:

$$\begin{aligned}
\mathcal{A}(\beta)(x) &= \beta(x), \qquad && x \in X \\
\mathcal{A}(\beta)(f(s_1, \dots, s_n)) &= f_{\mathcal{A}}(\mathcal{A}(\beta)(s_1), \dots, \mathcal{A}(\beta)(s_n)), \qquad && f/n \in \Omega
\end{aligned}$$

In the scope of a quantifier we need to evaluate terms with respect to modified assigments. To that end, let $\beta[x \to a] : X \to \mathcal{A}$, for $x \in X$ and $a \in \mathcal{A}$, denote the assignment

$$\beta[x \mapsto a](y) := \begin{cases} a & \text{if } x = y \\ \beta(y) & \text{otherwise} \end{cases}$$

Truth Value of a Formula in $\mathcal{A}$ with Respect to β

The set of truth values is given as $\{0, 1\}$. $\mathcal{A}(\beta) : \Sigma$-formulas $\to \{0, 1\}$ is defined inductively over the structure of F as follows:

$$\begin{aligned}
\mathcal{A}(\beta)(\bot) &= 0 \\
\mathcal{A}(\beta)(\top) &= 1 \\
\mathcal{A}(\beta)(p(s_1, \ldots, s_n)) &= 1 \;\Leftrightarrow\; (\mathcal{A}(\beta)(s_1), \ldots, \mathcal{A}(\beta)(s_n)) \in p_{\mathcal{A}} \\
\mathcal{A}(\beta)(s \approx t) &= 1 \;\Leftrightarrow\; \mathcal{A}(\beta)(s) = \mathcal{A}(\beta)(t) \\
\mathcal{A}(\beta)(\neg F) &= 1 \;\Leftrightarrow\; \mathcal{A}(\beta)(F) = 0 \\
\mathcal{A}(\beta)(F \rho G) &= \mathbb{B}_\rho(\mathcal{A}(\beta)(F), \mathcal{A}(\beta)(G)) \\
&\quad \text{with } \mathbb{B}_\rho \text{ the Boolean function associated with } \rho \\
\mathcal{A}(\beta)(\forall x F) &= \min_{a \in U}\{\mathcal{A}(\beta[x \mapsto a])(F)\} \\
\mathcal{A}(\beta)(\exists x F) &= \max_{a \in U}\{\mathcal{A}(\beta[x \mapsto a])(F)\}
\end{aligned}$$

Ex: “Standard” Interpretation $\mathbb{N}$ for Peano Arithmetic

$$\begin{array}{rcl}
U_{\mathbb{N}} & = & \{0, 1, 2, \ldots\} \\
0_{\mathbb{N}} & = & 0 \\
s_{\mathbb{N}} & : & n \mapsto n + 1 \\
+_{\mathbb{N}} & : & (n, m) \mapsto n + m \\
*_{\mathbb{N}} & : & (n, m) \mapsto n * m \\
\leq_{\mathbb{N}} & = & \{(n, m) \mid n \text{ less than or equal to } m\} \\
<_{\mathbb{N}} & = & \{(n, m) \mid n \text{ less than } m\}
\end{array}$$

Note that $\mathbb{N}$ is just one out of many possible Σ_{PA}-interpretations.

Values over $\mathbb{N}$ for Sample Terms and Formulas

Under the assignment $\beta : x \mapsto 1, y \mapsto 3$ we obtain

$$\begin{aligned}
\mathbb{N}(\beta)(s(x) + s(0)) &= 3 \\
\mathbb{N}(\beta)(x + y \approx s(y)) &= 1 \\
\mathbb{N}(\beta)(\forall x, y(x + y \approx y + x)) &= 1 \\
\mathbb{N}(\beta)(\forall z\, z \leq y) &= 0 \\
\mathbb{N}(\beta)(\forall x \exists y\, x < y) &= 1
\end{aligned}$$

1.3 Models, Validity, and Satisfiability

Validity and Satisfiability

F is valid in $\mathcal{A}$ under assigment β:

$$\mathcal{A}, \beta \models F \;\; :\Leftrightarrow \;\; \mathcal{A}(\beta)(F) = 1$$

F is valid in $\mathcal{A}$ ($\mathcal{A}$ is a model of F):

$$\mathcal{A} \models F \;\; :\Leftrightarrow \;\; \mathcal{A}, \beta \models F, \text{ for all } \beta \in X \to U_{\mathcal{A}}$$

F is valid (or is a tautology):

$$\models F \;\; :\Leftrightarrow \;\; \mathcal{A} \models F, \text{ for all } \mathcal{A} \in \Sigma\text{-Alg}$$

F is called satisfiable iff there exist $\mathcal{A}$ and β such that $\mathcal{A}, \beta \models F$.
Otherwise F is called unsatisfiable.

Substitution Lemma

The following theorems, to be proved by structural induction, hold for all Σ-algebras $\mathcal{A}$, assignments β, and substitutions σ.

Theorem

For any Σ-term t

$$\mathcal{A}(\beta)(t\sigma) = \mathcal{A}(\beta \circ \sigma)(t),$$

where $\beta \circ \sigma : X \to \mathcal{A}$ is the assignment $\beta \circ \sigma(x) = \mathcal{A}(\beta)(x\sigma)$.

Theorem

For any Σ-formula F, $\mathcal{A}(\beta)(F\sigma) = \mathcal{A}(\beta \circ \sigma)(F)$.

Corollary

$$\mathcal{A}, \beta \models F\sigma \quad \Leftrightarrow \quad \mathcal{A}, \beta \circ \sigma \models F$$

These theorems basically express that the syntactic concept of substitution corresponds to the semantic concept of an assignment.

Entailment and Equivalence

F entails (implies) G (or G is entailed by F), written $F \models G$

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma$-Alg and $\beta \in X \to U_{\mathcal{A}}$, whenever $\mathcal{A}, \beta \models F$ then $\mathcal{A}, \beta \models G$.

F and G are called equivalent

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma$-Alg und $\beta \in X \to U_{\mathcal{A}}$ we have $\mathcal{A}, \beta \models F \;\Leftrightarrow\; \mathcal{A}, \beta \models G$.

Theorem

F entails G iff $(F \implies G)$ is valid

Theorem

F and G are equivalent iff $(F \equiv G)$ is valid.

Extension to sets of formulas N in the “natural way”, e.g., $N \models F$

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma$-Alg and $\beta \in X \to U_{\mathcal{A}}$:
if $\mathcal{A}, \beta \models G$, for all $G \in N$, then $\mathcal{A}, \beta \models F$.

Validity vs. Unsatisfiability

Validity and unsatisfiability are just two sides of the same medal as explained by the following proposition.

Theorem

$$F \textit{ valid } \Leftrightarrow \neg F \textit{ unsatisfiable}$$

Hence in order to design a theorem prover (validity checker) it is sufficient to design a checker for unsatisfiability.
Q: In a similar way, entailment $N \models F$ can be reduced to unsatisfiability. How?

Theory of a Structure

Let $\mathcal{A} \in \Sigma$-Alg. The (first-order) theory of $\mathcal{A}$ is defined as

$$Th(\mathcal{A}) =_{df} \{G \in F_\Sigma(X) \mid \mathcal{A} \models G\}$$

Problem of axiomatizability:
For which structures $\mathcal{A}$ can one axiomatize $Th(\mathcal{A})$, that is, can one write down a formula F (or a recursively enumerable set F of formulas) such that

$$Th(\mathcal{A}) = \{G \mid F \models G\}?$$

Analoguously for sets of structures.

Two Interesting Theories

Let $\Sigma_{Pres} = (\{0/0, s/1, +/2\},\ \emptyset)$ and $\mathbb{Z}_+ = (\mathbb{Z}, 0, s, +)$ its standard interpretation on the integers.[1] $Th(\mathbb{Z}_+)$ is called Presburger arithmetic.[2] Presburger arithmetic is decidable in 3EXPTIME[3] (and there is a constant $c \geq 0$ such that $Th(\mathbb{Z}_+) \not\in$ NTIME$(2^{2^{cn}})$) and in 2EXPSPACE; usage of automata-theoretic methods.

However, $\mathbb{N}_* = (\mathbb{N}, 0, s, +, *)$, the standard interpretation of $\Sigma_{PA} = (\{0/0, s/1, +/2, */2\}, \emptyset)$, has as theory the so-called Peano arithmetic which is undedidable, not even recursively enumerable.

Note: The choice of signature can make a big difference with regard to the compational complexity of theories.

[1] There is no essential difference when one, instead of $\mathbb{Z}$, considers the natural numbers $\mathbb{N}$ as standard interpretation.

[2] M. Presburger (1929)

[3] D. Oppen: A $2^{2^{2^n}}$ upper bound on the complexity of Presburger arithmetic. Journal of Computer and System Sciences, 16(3):323–332, July 1978

1.4 Algorithmic Problems

Validity(F):

$\models F$?

Satisfiability(F):

F satisfiable?

Entailment(F,G):

does F entail G?

Model(A,F):

$A \models F$?

Solve(A,F):

find an assignment β such that $A, \beta \models F$

Solve(F):

find a substitution σ such that $\models F\sigma$

Abduce(F):

find G with “certain properties” such that G entails F

Gödel's Famous Theorems

1. For most signatures Σ, validity is undecidable for Σ-formulas. (We will prove this below.)
2. For each signature Σ, the set of valid Σ-formulas is recursively enumerable. (We will prove this by giving complete deduction systems.)
3. For $\Sigma = \Sigma_{PA}$ and $\mathbb{N}_* = (\mathbb{N}, 0, s, +, *)$, the theory $Th(\mathbb{N}_*)$ is not recursively enumerable.

These complexity results motivate the study of subclasses of formulas (fragments) of first-order logic

Q: Can you think of any fragments of first-order logic for which validity is decidable?

Some Decidable Fragments

- Monadic class: no function symbols, all predicates unary; validity NEXPTIME-complete
- Variable-free formulas without equality: satisfiability NP-complete
  Q: why?
- Variable-free Horn clauses (clauses with at most 1 positive atom): entailment is decidable in linear time (cf. below)
- Finite model checking is decidable in time polynomial in the size of the structure and the formula.

1.5 Normal Forms, Skolemization, Herbrand Models

Study of normal forms motivated by

- reduction of logical concepts,
- efficient data structures for theorem proving.

The main problem in first-order logic is the treatment of quantifiers. The subsequent normal form transformations are intended to eliminate many of them.

Prenex Normal Form

Prenex formulas have the form

$$Q_1x_1 \ldots Q_nx_n \; F,$$

where F quantifier-free, $Q_i \in \{\forall, \exists\}$; we call $Q_1x_1 \ldots Q_nx_n$ the quantifier prefix and F the matrix of the formula.

Computing prenex normal form by the rewrite relation $\Rightarrow_P$:

$$\begin{aligned}
(F \equiv G) \quad &\Rightarrow_P \quad (F \implies G) \land (G \implies F) \\
\neg QxF \quad &\Rightarrow_P \quad \overline{Q}x\neg F \qquad\qquad\qquad\qquad (\neg Q) \\
(QxF \;\rho\; G) \quad &\Rightarrow_P \quad Qy(F[y/x] \;\rho\; G),\; y \text{ fresh},\; \rho \in \{\land, \lor\} \\
(QxF \implies G) \quad &\Rightarrow_P \quad \overline{Q}y(F[y/x] \implies G),\; y \text{ fresh} \\
(F \;\rho\; QxG) \quad &\Rightarrow_P \quad Qy(F \;\rho\; G[y/x]),\; y \text{ fresh},\; \rho \in \{\land, \lor, \implies\}
\end{aligned}$$

Here $\overline{Q}$ denotes the quantifier dual to Q, i.e., $\overline{\forall} = \exists$ and $\overline{\exists} = \forall$.

Skolemization

Intuition: replacement of $\exists y$ by a concrete choice function computing y from all the arguments y depends on.

Transformation $\Rightarrow_S$ (to be applied outermost, not in subformulas):

$$\forall x_1, \dots, x_n \exists y F \quad \Rightarrow_S \quad \forall x_1, \dots, x_n F[f(x_1, \dots, x_n)/y]$$

where f/n is a new function symbol (Skolem function).

Together: $F \overset{*}{\Rightarrow}_P \underbrace{G}_{\text{prenex}} \overset{*}{\Rightarrow}_S \underbrace{H}_{\text{prenex, no } \exists}$

Theorem

Let F, G, and H as defined above and closed. Then

(i) F and G are equivalent.

(ii) $H \models G$ but the converse is not true in general.

(iii) G satisfiable (wrt. Σ-Alg) $\Leftrightarrow$ H satisfiable (wrt. Σ'-Alg)

where $\Sigma' = (\Omega \cup SKF, \Pi)$, if $\Sigma = (\Omega, \Pi)$.

Clausal Normal Form (Conjunctive Normal Form)

$$
\begin{array}{rcl}
(F \equiv G) & \Rightarrow_K & (F \implies G) \wedge (G \implies F) \\
(F \implies G) & \Rightarrow_K & (\neg F \vee G) \\
\neg(F \vee G) & \Rightarrow_K & (\neg F \wedge \neg G) \\
\neg(F \wedge G) & \Rightarrow_K & (\neg F \vee \neg G) \\
\neg\neg F & \Rightarrow_K & F \\
(F \wedge G) \vee H & \Rightarrow_K & (F \vee H) \wedge (G \vee H) \\
(F \wedge \top) & \Rightarrow_K & F \\
(F \wedge \bot) & \Rightarrow_K & \bot \\
(F \vee \top) & \Rightarrow_K & \top \\
(F \vee \bot) & \Rightarrow_K & F
\end{array}
$$

These rules are to be applied modulo associativity and commutativity of $\wedge$ and $\vee$. The first five rules, plus the rule $(\neg Q)$, compute the negation normal form (NNF) of a formula.

The Complete Picture

$$F \quad \overset{*}{\Rightarrow}_P \quad Q_1 y_1 \ldots Q_n y_n \; G$$ (*G* quantifier-free)

$$\overset{*}{\Rightarrow}_S \quad \forall x_1, \ldots, x_m \; H$$ ($m \leq n$, *H* quantifier-free)

$$\overset{*}{\Rightarrow}_K \quad \underbrace{\underbrace{\forall x_1, \ldots, x_m}_{\text{leave out}} \; \bigwedge_{i=1}^{k} \; \underbrace{\bigvee_{j=1}^{n_i} L_{ij}}_{\text{clauses } C_i}}_{F'}$$

$N = \{C_1, \ldots, C_k\}$ is called the clausal (normal) form (CNF) of F.

Note: the variables in the clauses are implicitly universally quantified.

Theorem

Let F be closed. $F' \models F$. The converse is not true in general.

Theorem

Let F be closed. F satisfiable iff F' satisfiable iff N satisfiable

Optimization

Here is lots of room for optimization since we only can preserve satisfiability anyway:

- size of the CNF exponential when done naively;
- want to preserve the original formula structure;
- want small arity of Skolem functions (cf. Info IV and tutorials)!

Herbrand Interpretations for FOL without Equality

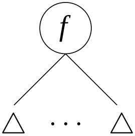
From now an we shall consider PL without equality. Ω shall contains at least one constant symbol.

A Herbrand interpretation (over Σ) is a Σ-algebra $\mathcal{A}$ such that

(i) $U_{\mathcal{A}} = T_{\Sigma}$ (= the set of ground terms over Σ)

(ii) $f_{\mathcal{A}} : (s_1, \ldots, s_n) \mapsto f(s_1, \ldots, s_n),\ f/n \in \Omega$

$$f_{\mathcal{A}}(\triangle, \ldots, \triangle) =$$



In other words, values are fixed to be ground terms and functions are fixed to be the term constructors. Only predicate symbols $p/m \in \Pi$ may be freely interpreted as relations $p_{\mathcal{A}} \subseteq T_{\Sigma}^{m}$.

Herbrand Interpretations as Sets of Ground Atoms

Theorem

Every set of ground atoms I uniquely determines a Herbrand interpretation $\mathcal{A}$ via

$$(s_1, \dots, s_n) \in p_{\mathcal{A}} \;:\Leftrightarrow\; p(s_1, \dots, s_n) \in I$$

Thus we shall identify Herbrand interpretations (over Σ) with sets of Σ-ground atoms.

Example: $\Sigma_{Pres} = (\{0/0, s/1, +/2\},\ \{< /2, \leq /2\})$

$\mathbb{N}$ as Herbrand interpretation over Σ_{Pres}:

$$\begin{aligned} I = \{ \quad & 0 \leq 0,\ 0 \leq s(0),\ 0 \leq s(s(0)),\ \dots, \\ & 0+0 \leq 0,\ 0+0 \leq s(0),\ \dots, \\ & \dots,\ (s(0)+0)+s(0) \leq s(0)+(s(0)+s(0)) \\ & \dots \\ & s(0)+0 < s(0)+0+0+s(0) \\ & \dots\} \end{aligned}$$

Existence of Herbrand Models

A Herbrand interpretation I is called a Herbrand model of F,
if $I \models F$.

Theorem (Herbrand)

Let N be a set of Σ clauses.

$$\begin{array}{lcl} N \textit{ satisfiable} & \Leftrightarrow & N \textit{ has a Herbrand model (over } \Sigma\textit{)} \\ & \Leftrightarrow & G_\Sigma(N) \textit{ has a Herbrand model (over } \Sigma\textit{)} \end{array}$$

where

$$G_\Sigma(N) = \{C\sigma \textit{ ground clause} \mid C \in N,\ \sigma : X \rightarrow T_\Sigma\}$$

the set of ground instances of N.

[Proof to be given below in the context of the completeness proof for resolution.]

Example of a G_Σ

For Σ_{Pres} one obtains for

$$C = (x < y) \vee (y \leq s(x))$$

the following ground instances:

$(0 < 0) \vee (0 \leq s(0))$
$(s(0) < 0) \vee (0 \leq s(s(0)))$
$\ldots$
$(s(0) + s(0) < s(0) + 0) \vee (s(0) + 0 \leq s(s(0) + s(0)))$
$\ldots$

1.6 Inference Systems, Proofs

Inference systems Γ (proof calculi) are sets of tuples

$$(F_1, \ldots, F_n, F_{n+1}),\ n \geq 0,$$

called inferences or inference rules, and written

$$\frac{\overbrace{F_1 \ \ldots\ F_n}^{\text{premises}}}{\underbrace{F_{n+1}}_{\text{conclusion}}}\ .$$

Clausal inference system: premises and conclusions are clauses. One also considers inference systems over other data structures (cf. below).

A proof in Γ of a formula F from a a set of formulas N (called assumptions) is a sequence $F_1, \ldots, F_k$ of formulas where (i) $F_k = F$, (ii) for all $1 \leq i \leq k$: $F_i \in N$, or else there exists an inference $(F_{i_1},\ \ldots,\ F_{i_{n_i}},\ F_i)$ in Γ, such that $0 \leq i_j < i$, for $1 \leq j \leq n_i$.

Soundness, Completeness

Provability $\vdash_\Gamma$ of F from N in Γ:
$N \vdash_\Gamma F$ $:\Leftrightarrow$ there exists a proof Γ of F from N.
Γ is called sound $:\Leftrightarrow$

$$\frac{F_1 \ \ldots \ F_n}{F} \in \Gamma \quad \Rightarrow \quad F_1, \ldots, F_n \models F$$

Γ is called complete $:\Leftrightarrow$

$$N \models F \quad \Rightarrow \quad N \vdash_\Gamma F$$

Γ is called refutationally complete $:\Leftrightarrow$

$$N \models \bot \quad \Rightarrow \quad N \vdash_\Gamma \bot$$

Proofs as Trees

markings $\triangleq$ formulas

leaves $\triangleq$ assumptions and axioms

other nodes $\triangleq$ inferences: conclusion $\triangleq$ ancestor

premises $\triangleq$ direct descendants

$$\dfrac{P(g(a,b)) \qquad \dfrac{\dfrac{\dfrac{P(f(a)) \lor Q(b) \qquad \dfrac{\dfrac{P(f(a)) \lor Q(b) \qquad \neg P(f(a)) \lor \neg P(f(a)) \lor Q(b)}{\neg P(f(a)) \lor Q(b) \lor Q(b)}}{\neg P(f(a)) \lor Q(b)}}{Q(b) \lor Q(b)}}{Q(b)} \qquad \neg P(f(a)) \lor \neg Q(b)}{\neg P(g(a,b))}}{\bot}$$

Theorem

(i) Let Γ be sound. Then $N \vdash_\Gamma F \;\Rightarrow\; N \models F$

(ii) $N \vdash_\Gamma F \;\Rightarrow\;$ there exist $F_1, \ldots, F_n \in N$ s.t. $F_1, \ldots, F_n \vdash_\Gamma F$ (resembles compactness).

1.7 Propositional Resolution

We observe that propositional clauses and ground clauses are the same concept.
In this section we only deal with ground clauses.

The Resolution Calculus *Res*

Definition

- Resolution inference rule

$$\frac{C \vee A \qquad \neg A \vee D}{C \vee D}$$

- (positive) factorisation

$$\frac{C \vee A \vee A}{C \vee A}$$

These are schematic inference rules; for each substitution of the schematic variables C, D, and A, respectively, by ground clauses and ground atoms we obtain an inference rule.

As “$\vee$” is considered associative and commutative, we assume that A and $\neg A$ can occur anywhere in their respective clauses.

Sample Refutation

Example

1.	$\neg P(f(a)) \lor \neg P(f(a)) \lor Q(b)$	(given)
2.	$P(f(a)) \lor Q(b)$	(given)
3.	$\neg P(g(b,a)) \lor \neg Q(b)$	(given)
4.	$P(g(b,a))$	(given)
5.	$\neg P(f(a)) \lor Q(b) \lor Q(b)$	(Res. 2. into 1.)
6.	$\neg P(f(a)) \lor Q(b)$	(Fact. 5.)
7.	$Q(b) \lor Q(b)$	(Res. 2. into 6.)
8.	$Q(b)$	(Fact. 7.)
9.	$\neg P(g(b,a))$	(Res. 8. into 3.)
10.	$\bot$	(Res. 4. into 9.)

Resolution with Implicit Factorization RIF

$$\frac{C \vee A \vee \ldots \vee A \qquad \neg A \vee D}{C \vee D}$$

Example

1.	$\neg P(f(a)) \vee \neg P(f(a)) \vee Q(b)$	(given)
2.	$P(f(a)) \vee Q(b)$	(given)
3.	$\neg P(g(b,a)) \vee \neg Q(b)$	(given)
4.	$P(g(b,a))$	(given)
5.	$\neg P(f(a)) \vee Q(b) \vee Q(b)$	(Res. 2. into 1.)
6.	$Q(b) \vee Q(b) \vee Q(b)$	(Res. 2. into 5.)
7.	$\neg P(g(b,a))$	(Res. 6. into 3.)
8.	$\perp$	(Res. 4. into 7.)

Soundness of Resolution

Theorem

Propositional resolution is sound.

Proof of L. et $I \in \Sigma$-Alg. To be shown:

(i) for resolution: $I \models C \vee A,\ I \models D \vee \neg A \quad \Rightarrow \quad I \models C \vee D$

(ii) for factorization: $I \models C \vee A \vee A \ \Rightarrow \ I \models C \vee A$

ad (i): Assume premises are valid in I. Two cases need to be considered:

(a) A is valid, or (b) $\neg A$ is valid.

a) $I \models A \ \Rightarrow \ I \models D \ \Rightarrow \ I \models C \vee D$

b) $I \models \neg A \ \Rightarrow \ I \models C \ \Rightarrow \ I \models C \vee D$

ad (ii): even simpler. ■

NB: In propositional logic (ground clauses) we have:

1. $I \models L_1 \vee \ldots \vee L_n \quad \Leftrightarrow \quad$ there exists i: $I \models L_i$.
2. $I \models A$ or $I \models \neg A$.

1.8 Well-Founded Orderings

Literature: Baader F., Nipkow, T.: Term rewriting and all that. Cambridge U. Press, 1998, Chapter 2.

For showing completeness of resolution we will make use of the concept of well-founded orderings. A partial ordering $\succ$ on a set M is called well-founded (Noetherian) iff there exists no infinite descending chain

$$a_0 \succ a_1 \succ \ldots$$

in M.

NB: A partial ordering is transitive and irreflexive and not necessarily total (however our orderings usually are total).

An $x \in M$ is called minimal, if there is no y in M such that $x \succ y$.

Notation

$\prec$ for the inverse relation $\succ^{-1}$

$\succeq$ for the reflexive closure $(\succ \cup =)$ of $\succ$

Examples

Natural numbers. $(\mathbb{N}, >)$

Lexicographic orderings. Let $(M_1, \succ_1), (M_2, \succ_2)$ be well-founded orderings. Then let their lexicographic combination

$$\succ = (\succ_1, \succ_2)_{lex}$$

on $M_1 \times M_2$ be defined as

$(a_1, a_2) \succ (b_1, b_2) \;:\Leftrightarrow\; a_1 \succ_1 b_1,$ or else $a_1 = b_1 \;\&\; a_2 \succ_2 b_2$

This again yields a well-founded ordering (proof below).

Length-based ordering on words. For alphabets Σ with a well-founded ordering $>_\Sigma$, the relation $\succ$, defined as

$w \succ w' \quad := \quad \alpha)\ |w| > |w'|$ or

$\beta)\ |w| = |w'|$ and $w >_{\Sigma,lex} w'$,

is a well-founded ordering on Σ^* (proof below).

Basic Properties of Well-Founded Orderings

Lemma

$(M, \succ)$ *is well-founded* $\Leftrightarrow$ *every* $\emptyset \subset M' \subseteq M$ *has a minimal element.*

Lemma

$(M_i, \succ_i)$ *well-founded ,* $i = 1, 2$ $\Leftrightarrow$ $(M_1 \times M_2, (\succ_1, \succ_2)_{lex})$ *well-founded.*

Proof of (. i) “$\Rightarrow$”: Suppose $(M_1 \times M_2,\ \succ)$, with $\succ = (\succ_1, \succ_2)_{lex}$, is not well-founded. Then there is an infinite sequence

$$(a_0, b_0) \succ (a_1, b_1) \succ (a_2, b_2) \succ \ldots \quad .$$

Consider $A = \{a_i \mid i \geq 0\} \subseteq M_1$. A has a minimal element a_n, since $(M_1, \succ_1)$ is well-founded. But then $B = \{b_i \mid i \geq n\} \subseteq M_2$ can not have a minimal element; *contradition* to the well-foundedness of $(M_2, \succ_2)$.
(ii) “$\Leftarrow$”: obvious. ■

Noetherian Induction

Let $(M, \succ)$ be a well-founded ordering.

Theorem (Noetherian Induction)

A property $Q(m)$ holds for all $m \in M$, whenever for all $m \in M$ this implication is satisfied:

> *if $Q(m')$, for all $m' \in M$ such that $m \succ m'$,*[a]
> *then $Q(m)$.*[b]

[a] induction hypothesis

[b] induction step

Proof of L. et $X = \{m \in M \mid Q(m)$ false$\}$. Suppose, $X \neq \emptyset$. Since $(M, \succ)$ is well-founded, X has a minimal element m_1. Hence for all $m' \in M$ with $m' \prec m_1$ the property $Q(m')$ holds. On the other hand, the implication which is presupposed for this theorem holds in particular also for m_1, hence $Q(m_1)$ must be true so that m_1 can not be in X. *Contradiction*. ■

Multi-Sets

Let M be a set. A multi-set S over M is a mapping $S : M \to \mathbb{N}$. Hereby $S(m)$ specifies the number of occurrences of elements m of the base set M within the multi-set S.

m is called an element of S, if $S(m) > 0$. We use set notation ($\in$, $\subset$, $\subseteq$, $\cup$, $\cap$, etc.) with analogous meaning also for multi-sets, e.g.,

$$\begin{array}{rcl} (S_1 \uplus S_2)(m) & = & S_1(m) + S_2(m) \\ (S_1 \cap S_2)(m) & = & \min\{S_1(m), S_2(m)\} \end{array}$$

A multi-set is called finite, if

$$|\{m \in M |\ s(m) > 0\}| < \infty,$$

for each m in M.

From now on we only consider finite multi-sets.

Example. $S = \{a, a, a, b, b\}$ is a multi-set over $\{a, b, c\}$, where $S(a) = 3$, $S(b) = 2$, $S(c) = 0$.

Multi-Set Orderings

Definition ($\succ_{mul}$)

Let $(M, \succ)$ be a partial ordering. The multi-set extension of $\succ$ to multi-sets over M is defined by

$$\begin{aligned} S_1 \succ_{mul} S_2 \;:\Leftrightarrow\; & S_1 \neq S_2 \\ & \text{and } \forall m \in M : [S_2(m) > S_1(m) \\ & \quad \Rightarrow \; \exists m' \in M : (m' \succ m \text{ and } S_1(m') > S_2(m'))] \end{aligned}$$

Theorem

a) $\succ_{mul}$ *is a partial ordering.*
b) $\succ$ *well-founded* $\Rightarrow \succ_{mul}$ *well-founded*
c) $\succ$ *total* $\Rightarrow \succ_{mul}$ *total*

Clause Orderings

1. We assume that $\succ$ is any fixed ordering on ground atoms that is total and well-founded. (There exist many such orderings, e.g., the lenght-based ordering on atoms when these are viewed as words over a suitable alphabet such as ASCII.)
2. Extension to literals:

$$\begin{array}{lcll} [\neg]A & \succ_L & [\neg]B & \text{, if } A \succ B \\ \neg A & \succ_L & A & \end{array}$$

3. Extension to an ordering $\succ_C$ on ground clauses: $\succ_C = (\succ_L)_{mul}$, the multi-set extension of the literal ordering $\succ_L$.

Notation: $\succ$ also for $\succ_L$ and $\succ_C$.

Example

Example

Suppose $A_5 \succ A_4 \succ A_3 \succ A_2 \succ A_1 \succ A_0$.
Order the following clauses:

$\neg A_1 \vee \neg A_4 \vee A_3$

$\neg A_1 \vee A_2$

$\neg A_1 \vee A_4 \vee A_3$

$A_0 \vee A_1$

$\neg A_5 \vee A_5$

$A_1 \vee A_2$

Example

Example

Suppose $A_5 \succ A_4 \succ A_3 \succ A_2 \succ A_1 \succ A_0$.
Then:

$$\begin{array}{rc} & A_0 \vee A_1 \\ \prec & A_1 \vee A_2 \\ \prec & \neg A_1 \vee A_2 \\ \prec & \neg A_1 \vee A_4 \vee A_3 \\ \prec & \neg A_1 \vee \neg A_4 \vee A_3 \\ \prec & \neg A_5 \vee A_5 \end{array}$$

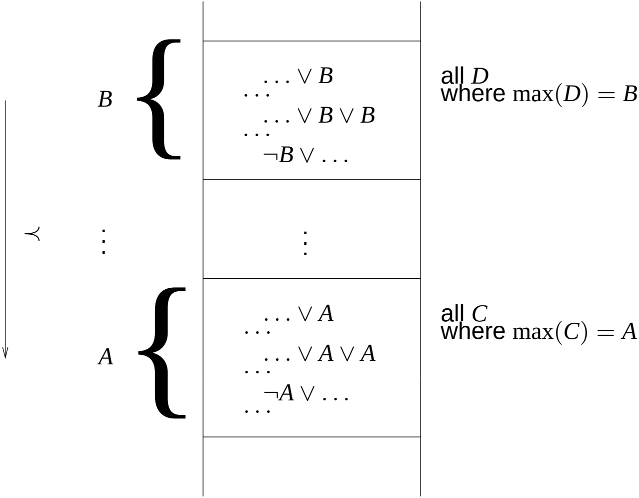
Properties of the Clause Ordering

Theorem

1. *The orderings on literals and clauses are total and well-founded.*
2. *Let C and D be clauses with $A = \max(C)$, $B = \max(D)$, where $\max(C)$ denotes the maximal atom in C.*
   - (i) *If $A \succ B$ then $C \succ D$.*
   - (ii) *If $A = B$, A occurs negatively in C but only positively in D, then $C \succ D$.*

Stratified Structure of Clause Sets

Let $A \succ B$. Clause sets are then stratified in this form:



$\prec$

B

$\dots \vee B$

$\dots$

$\dots \vee B \vee B$

$\dots$

$\neg B \vee \dots$

all D where $\max(D) = B$

$\vdots$

$\vdots$

A

$\dots \vee A$

$\dots$

$\dots \vee A \vee A$

$\dots$

$\neg A \vee \dots$

$\dots$

all C where $\max(C) = A$

Closure of Clause Sets under Res

Definition

$$\begin{aligned} Res(N) &= \{C \mid C \text{ is concl. of a rule in } Res \text{ w/ premises in } N\} \\ Res^0(N) &= N \\ Res^{n+1}(N) &= Res(Res^n(N)) \cup Res^n(N), \text{ for } n \geq 0 \\ Res^*(N) &= \textstyle\bigcup_{n\geq 0} Res^n(N) \end{aligned}$$

N is called saturated (wrt. resolution), if $Res(N) \subseteq N$.

Theorem

(i) $Res^*(N)$ *is saturated.*

(ii) Res *is refutationally complete, iff for each set* N *of ground clauses:*

$$N \models \bot \quad \Leftrightarrow \quad \bot \in Res^*(N)$$

Construction of Interpretations

Given:

set N of ground clauses, atom ordering $\succ$.

Wanted:

Herbrand interpretation I such that

- “many” clauses from N are valid in I;
- *$I \models N$, if N is saturated and $\bot \notin N$.*

Construction according to $\succ$, starting with the minimal clause.

Construction of Interpretations

Example

Let $A_5 \succ A_4 \succ A_3 \succ A_2 \succ A_1 \succ A_0$ (max. literals in red)

	clauses C	I_C	Δ_C	Remarks
1	$\neg A_0$	$\emptyset$	$\emptyset$	true in I_C
2	$A_0 \vee A_1$	$\emptyset$	$\{A_1\}$	A_1 maximal
3	$A_1 \vee A_2$	$\{A_1\}$	$\emptyset$	true in I_C
4	$\neg A_1 \vee A_2$	$\{A_1\}$	$\{A_2\}$	A_2 maximal
5	$\neg A_1 \vee A_4 \vee A_3 \vee A_0$	$\{A_1, A_2\}$	$\{A_4\}$	A_4 maximal
6	$\neg A_1 \vee \neg A_4 \vee A_3$	$\{A_1, A_2, A_4\}$	$\emptyset$	A_3 not maximal; *min. counter-ex.*
7	$\neg A_1 \vee A_5$	$\{A_1, A_2, A_4\}$	$\{A_5\}$	

$I = \{A_1, A_2, A_4, A_5\}$ is not a model of the clause set

$\Rightarrow$ there exists a counterexample.

Main Ideas of the Construction

- Clauses are considered in the order given by $\prec$.
- When considering C, one already has a partial interpretation I_C (initially $I_C = \emptyset$) available.
- If C is true in the partial interpretation I_C, nothing is done. $(\Delta_C = \emptyset)$.
- If C is false, one would like to change I_C such that C becomes true.
- Changes should, however, be monotone. One never deletes anything from I_C and the truthvalue of clauses smaller than C shouldb be maintained the way it was in I_C.
- Hence, one chooses $\Delta_C = \{A\}$ if, and only if, C is false in I_C, if A occurs positively in C (adding A will make C become true) and if this occurrence in C is strictly maximal in the ordering on literals (changing the truthvalue of A has no effect on smaller clauses).

Resolution Reduces Counterexamples

Example

$$\frac{\neg A_1 \lor A_4 \lor A_3 \lor A_0 \qquad \neg A_1 \lor \neg A_4 \lor A_3}{\neg A_1 \lor \neg A_1 \lor A_3 \lor A_3 \lor A_0}$$

Construction of I for the extended clause set:

clauses C	I_C	Δ_C	Remarks
$\neg A_0$	$\emptyset$	$\emptyset$	
$A_0 \lor A_1$	$\emptyset$	$\{A_1\}$	
$A_1 \lor A_2$	$\{A_1\}$	$\emptyset$	
$\neg A_1 \lor A_2$	$\{A_1\}$	$\{A_2\}$	
$\neg A_1 \lor \neg A_1 \lor A_3 \lor A_3 \lor A_0$	$\{A_1, A_2\}$	$\emptyset$	A_3 occurs twice *minimal counter-ex.*
$\neg A_1 \lor A_4 \lor A_3 \lor A_0$	$\{A_1, A_2\}$	$\{A_4\}$	
$\neg A_1 \lor \neg A_4 \lor A_3$	$\{A_1, A_2, A_4\}$	$\emptyset$	counterexample
$\neg A_1 \lor A_5$	$\{A_1, A_2, A_4\}$	$\{A_5\}$	

The same I, but smaller counterexample, hence some progress was made.

Factorization Reduces Counterexamples

Example

$$\frac{\neg A_1 \vee \neg A_1 \vee A_3 \vee A_3 \vee A_0}{\neg A_1 \vee \neg A_1 \vee A_3 \vee A_0}$$

Construction of I for the extended clause set:

clauses C	I_C	Δ_C	Remarks
$\neg A_0$	$\emptyset$	$\emptyset$	
$A_0 \vee A_1$	$\emptyset$	$\{A_1\}$	
$A_1 \vee A_2$	$\{A_1\}$	$\emptyset$	
$\neg A_1 \vee A_2$	$\{A_1\}$	$\{A_2\}$	
$\neg A_1 \vee \neg A_1 \vee A_3 \vee A_0$	$\{A_1, A_2\}$	$\{A_3\}$	
$\neg A_1 \vee \neg A_1 \vee A_3 \vee A_3 \vee A_0$	$\{A_1, A_2, A_3\}$	$\emptyset$	true in I_C
$\neg A_1 \vee A_4 \vee A_3 \vee A_0$	$\{A_1, A_2, A_3\}$	$\emptyset$	
$\neg A_1 \vee \neg A_4 \vee A_3$	$\{A_1, A_2, A_3\}$	$\emptyset$	true in I_C
$\neg A_3 \vee A_5$	$\{A_1, A_2, A_3\}$	$\{A_5\}$	

The resulting $I = \{A_1, A_2, A_3, A_5\}$ is a model of the clause set.

Construction of Candidate Models Formally

Definition

Let $N, \succ$ be given. We define sets I_C and Δ_C for all ground clauses C over the given signature inductively over $\succ$:

$$I_C \quad := \quad \bigcup_{C \succ D} \Delta_D$$

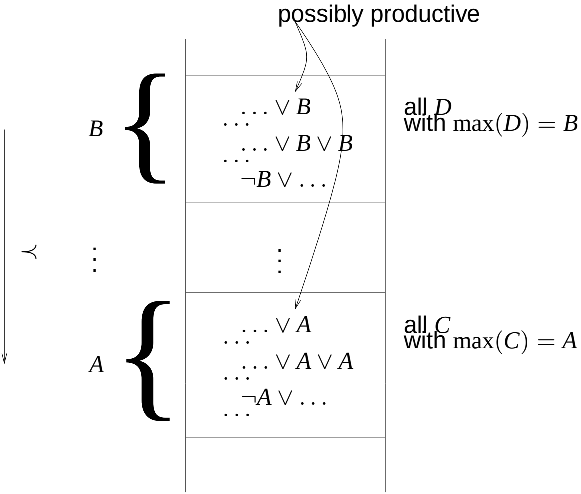
$$\Delta_C \quad := \quad \begin{cases} \{A\}, & \text{if } C \in N,\ C = C' \vee A,\ A \succ C',\ I_C \not\models C \\ \emptyset, & \text{otherwise} \end{cases}$$

We say that C produces A, if $\Delta_C = \{A\}$.

The candidate model for N (wrt. $\succ$) is given as $I_N^{\succ} := \bigcup_C \Delta_C$.
We also simply write I_N, or I, for $I_N^{\succ}$ if $\succ$ is either irrelevant or known from the context.

Structure of $N, \succ$

Let $A \succ B$; producing a new atom does not affect smaller clauses.



Some Properties of the Construction

Theorem

(i) $C = \neg A \vee C' \;\Rightarrow\;$ *no* $D \succeq C$ *produces* A*.*

(ii) C *productive* $\Rightarrow I_C \cup \Delta_C \models C$*.*

(iii) *Let* $D' \succ D \succeq C$*. Then*

$$I_D \cup \Delta_D \models C \;\Rightarrow I_{D'} \cup \Delta_{D'} \models C \text{ and } I_N \models C.$$

If, in addition, $C \in N$ *or* $\max(D) \succ \max(C)$*:*

$$I_D \cup \Delta_D \not\models C \;\Rightarrow I_{D'} \cup \Delta_{D'} \not\models C \text{ and } I_N \not\models C.$$

(iv) *Let* $D' \succ D \succ C$*. Then*

$$I_D \models C \;\Rightarrow I_{D'} \models C \text{ and } I_N \models C.$$

If, in addition, $C \in N$ *or* $\max(D) \succ \max(C)$*:*

Model Existence Theorem

Theorem (Bachmair, Ganzinger 1990)

Let $\succ$ be a clause ordering, let N be saturated wrt. Res, and suppose that $\bot \notin N$. Then $I_N^{\succ} \models N$.

Proof of S. uppose $\bot \notin N$, but $I_N^{\succ} \not\models N$. Let $C \in N$ minimal (in $\succ$) such that $I_N^{\succ} \not\models C$. Since C is false in I_N, C is not productive. As $C \neq \bot$ there exists a maximal atom A in C.

Case 1: $C = \neg A \lor C'$ (i.e., the maximal atom occurs negatively)

$\Rightarrow I_N \models A$ and $I_N \not\models C'$

$\Rightarrow$ some $D = D' \lor A \in N$ produces A. As $\frac{D' \lor A \qquad \neg A \lor C'}{D' \lor C'}$, we infer that $D' \lor C' \in N$, and $C \succ D' \lor C'$ and $I_N \not\models D' \lor C'$

$\Rightarrow$ contradicts minimality of C.

Case 2: $C = C' \lor A \lor A$. Then $\frac{C' \lor A \lor A}{C' \lor A}$ yields a smaller counterexample $C' \lor A \in N$. Contradiction. ∎

Model Existence Theorem

Theorem (Bachmair, Ganzinger 1990)

Let $\succ$ be a clause ordering, let N be saturated wrt. Res, and suppose that $\bot \notin N$. Then $I_N^{\succ} \models N$.

Corollary

Let N be saturated wrt. Res. Then $N \models \bot \;\Leftrightarrow\; \bot \in N$.

Compactness of Propositional Logic

Theorem (Compactness)

Let N be a set of propositional formulas. Then N unsatisfiable if, and only if, there exists $M \subseteq N$, with $|M| < \infty$, and M unsatisfiable.

Proof of .

"$\Leftarrow$": trivial.

"$\Rightarrow$": Let N be unsatisfiable.

$\Rightarrow$ $Res^*(N)$ unsatisfiable

$\Rightarrow$ (completeness of resolution) $\bot \in Res^*(N)$

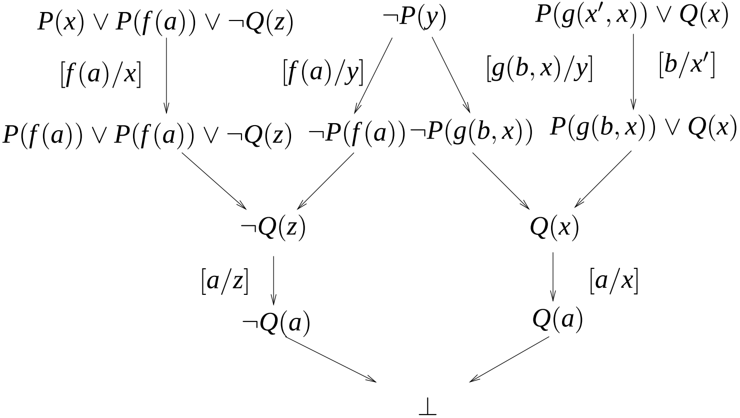
$\Rightarrow$ $\exists n \geq 0 : \bot \in Res^n(N)$

$\Rightarrow$ $\bot$ has a finite resolution proof P;

choose M as the set of assumptions in P. ■

General Resolution through Instantiation

(We use *RIF*, resolution with implicit factorisation.) Observe that (i) upon instantiation two literals in a clause can become equal; and (ii) generally more than one instance of a clause participate in a proof.



$P(x) \lor P(f(a)) \lor \neg Q(z)$

$\neg P(y)$

$P(g(x', x)) \lor Q(x)$

$[f(a)/x]$

$[f(a)/y]$

$[g(b, x)/y]$

$[b/x']$

$P(f(a)) \lor P(f(a)) \lor \neg Q(z)$

$\neg P(f(a))$

$\neg P(g(b, x))$

$P(g(b, x)) \lor Q(x)$

$\neg Q(z)$

$Q(x)$

$[a/z]$

$[a/x]$

$\neg Q(a)$

$Q(a)$

$\bot$

Lifting Principle

Problem: Make saturation of infinite sets of clauses as they arise from taking the (ground) instances of finitely many general clauses (with variables) effective and efficient.

Idea (Robinson 65):
- Resolution for general clauses
- Equality of ground atoms is generalized to unifiability of general atoms
- Only compute most general (minimal) unfiers

Significance: The advantage of the method in (Robinson 65) compared with (Gilmore 60) is that unification enumerates only those instances of clauses that participate in an inference. Moreover, clauses are not right away instantiated into ground clauses. Rather they are instantiated only as far as required for an inference. Inferences with non-ground clauses in general represent infinite sets of ground inferences which are computed simultaneously in a single step.

Resolution for General Clauses

General binary resolution *Res*:

$$\frac{C \vee A \qquad D \vee \neg B}{(C \vee D)\sigma} \quad \text{if } \sigma = \text{mgu}(A, B) \qquad [\text{resolution}]$$

$$\frac{C \vee A \vee B}{(C \vee A)\sigma} \quad \text{if } \sigma = \text{mgu}(A, B) \quad [\text{factorization}]$$

General resolution *RIF* with implicit factorization:

$$\frac{C \vee A_1 \vee \ldots \vee A_n \qquad D \vee \neg B}{(C \vee D)\sigma} \quad \text{if } \sigma = \text{mgu}(A_1, \ldots, A_n, B)$$

We additionally assume that the variables in one of the two premises of the resolutions rule are (bijectively) renamed such that they become different to any variable in the other premise. We do not formalize this. Which names one uses for variables is otherwise irrelevant.

Unification

Definition

Let $E = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ (s_i, t_i terms or atoms) a multi-set of equality problems. A substitution σ is called a unifier of E :⇔

$$\forall 1 \leq i \leq n : s_i\sigma = t_i\sigma.$$

If a unifier exists, E is called unifiable. If a unifier of E is more general than any other unifier of E, then we speak of a most general unifier (mgu) of E. Hereby a substitution σ is called more general than a substitution τ

$$\sigma \leq \tau \;:\Leftrightarrow\; \text{there exists a substitution } \varrho \text{ s.t. } \varrho \circ \sigma = \tau$$

where $(\varrho \circ \sigma)(x) := (x\sigma)\varrho$ is the composition of σ and ϱ als mappings.[4]

[4] Note that $\varrho \circ \sigma$ has a finite domain as required for a substitution.

Unification

Theorem

(Exercise)

(i) $\leq$ is a quasi-ordering on substitutions, and $\circ$ is associative.

(ii) If $\sigma \leq \tau$ and $\tau \leq \sigma$ (we write $\sigma \sim \tau$ in this case), then $x\sigma$ and $x\tau$ are equal up to (bijective) variable renaming, for any x in X.

Unification after Martelli/Montanari

$$
\begin{array}{rll}
t \doteq t, E & \Rightarrow_{MM} & E \\[1em]
f(s_1, \ldots, s_n) \doteq f(t_1, \ldots, t_n), E & \Rightarrow_{MM} & s_1 \doteq t_1, \ldots, s_n \doteq t_n, E \\[1em]
f(\ldots) \doteq g(\ldots), E & \Rightarrow_{MM} & \bot \\[1em]
x \doteq t, E & \Rightarrow_{MM} & x \doteq t, E[t/x] \\
 & \multicolumn{2}{l}{\text{if } x \in var(E), x \notin var(t)} \\[1em]
x \doteq t, E & \Rightarrow_{MM} & \bot \\
 & \multicolumn{2}{l}{\text{if } x \neq t, x \in var(t)} \\[1em]
t \doteq x, E & \Rightarrow_{MM} & x \doteq t, E \\
 & \multicolumn{2}{l}{\text{if } t \notin X}
\end{array}
$$

MM: Main Properties

A substitution σ is called idempotent, if $\sigma \circ \sigma = \sigma$.

Theorem

σ is idempotent iff $dom(\sigma) \cap codom(\sigma) = \emptyset$.

If $E = x_1 \doteq u_1, \ldots, x_k \doteq u_k$, with x_i pw. distinct, $x_i \notin var(u_j)$, then E is called an (equational problem in) solved form representing the solution $\sigma_E = [u_1/x_1, \ldots, u_k/x_k]$.

Theorem

If E is a solved form then σ_E is am mgu of E.

Theorem

1. *If $E \Rightarrow_{MM} E'$ then σ unifier of E iff σ unfier of E'*
2. *If $E \Rightarrow^*_{MM} \perp$ then E is not unifiable.*
3. *If $E \Rightarrow^*_{MM} E'$, with E' a solved form, then $\sigma_{E'}$ is an mgu of E.*

Main Unification Theorem

Theorem

E unifiable $\Leftrightarrow$ there exists a most general unifier σ of E, such that σ is idempotent and $dom(\sigma) \cup codom(\sigma) \subseteq var(E)$.

Notation: $\sigma = \text{mgu}(E)$
Problem: exponential growth of terms possible

Proof of the Unification Theorem

- Systems E irreducible wrt. $\Rightarrow_{MM}$ are either $\perp$ or a solved form.
- $\Rightarrow_{MM}$ is Noetherian. A suitable lexicographic ordering on the multisets E (with $\perp$ minimal) shows this. Compare in this order:
  1. the number of defined variables (d.h. variables x in equations $x \doteq t$ with $x \notin var(t)$), which also occur outside their definition elsewhere in E;
  2. the multi-set ordering induced by (i) the size (number of symbols) in an equation; (ii) if sizes are equal consider $x \doteq t$ smaller than $t \doteq x$, if $t \notin X$.
- Therefore, reducing any E by MM with end (no matter what reduction strategy we apply) in an irreducible E' having the same unifiers as E, and we can read off the mgu (or non-unifiability) of E from E' (Theorem 41, Proposition 40).
- σ is idempotent because of the substitution in rule 4. $dom(\sigma) \cup codom(\sigma) \subseteq var(E)$, as no new variables are generated.

Lifting Lemma

Lemma

Let C and D be variable-disjoint clauses. If

$$\begin{tikzcd} C \arrow[d, "\sigma"] & D \arrow[d, "\varrho"] \\ C\sigma & D\varrho \end{tikzcd}$$

$$\frac{C\sigma \qquad D\varrho}{C'}$$

[propositional resolution]

then there exists a substitution τ such that

$$\begin{tikzcd} \dfrac{C \qquad D}{C''} \arrow[d, "\tau"] \\ C' = C''\tau \end{tikzcd}$$

[general resolution]

Same for factorization.

Saturation of Sets of General Clauses

Corollary

Let N be a set of general clauses saturated unter Res, i.e., $\mathit{Res}(N) \subseteq N$. Then also $G_\Sigma(N)$ is saturated, that is,

$$\mathit{Res}(G_\Sigma(N)) \subseteq G_\Sigma(N).$$

Proof of W. olog we may assume that clauses in N are pairwise variable-disjoint. (Otherwise make them disjoint, and this renaming process does neither change $\mathit{Res}(N)$ nor $G_\Sigma(N)$.)

Let $C' \in \mathit{Res}(G_\Sigma(N))$, meaning (i) there exist resolvable ground instances $C\sigma$ and $D\varrho$ of N with resolvent C', or else (ii) C' is a factor of a ground instance $C\sigma$ of C.

Ad (i): By the Lifting Lemma, C and D are resolvable with a resolvent C'' with $C''\tau = C'$, for a suitable substitution τ. As $C'' \in N$ by assumption, we obtain that $C' \in G_\Sigma(N)$.

Ad (ii): Similar. ■

Herbrand’s Theorem

Theorem (Herbrand)

Let N be a set of Σ-clauses.

$$N \text{ satisfiable } \Leftrightarrow N \text{ has a Herbrand model over } \Sigma$$

Proof of “$\Leftarrow$”. trivial
“$\Rightarrow$”

$$\begin{aligned} N \not\models \bot &\Rightarrow \bot \notin Res^*(N) && \text{(resolution is sound)} \\ &\Rightarrow \bot \notin G_\Sigma(Res^*(N)) \\ &\Rightarrow I_{G_\Sigma(Res^*(N))} \models G_\Sigma(Res^*(N)) && (\text{Theorem 34; Corollary 44}) \\ &\Rightarrow I_{G_\Sigma(Res^*(N))} \models Res^*(N) && (I \text{ is a Herbrand model}) \\ &\Rightarrow I_{G_\Sigma(Res^*(N))} \models N && (N \subseteq Res^*(N)) \end{aligned}$$

■

The Theorem of Löwenheim-Skolem

Theorem (Löwenheim-Skolem)

Let Σ be a countable signature and let S be a set of closed Σ-formulas. Then S is satisfiable iff S has a model over a countable universe.

Proof of S. et S can be at most countably infinite if both X and Σ are countable. Now generate, maintaining satisfiability, a set N of clauses from S. This extends Σ by at most countably many new Skolem functions to Σ'. As Σ' is countable, so is $T_{\Sigma'}$, the universe of Herbrand-interpretations over Σ'. Now apply Thereom 45. ■

Refutational Completeness of General Resolution

Theorem

Let N be a set of general clauses where $Res(N) \subseteq N$. Then

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

Proof of L. et $Res(N) \subseteq N$. By Corollary 44: $Res(G_\Sigma(N)) \subseteq G_\Sigma(N)$

$$\begin{aligned} N \models \bot \ &\Leftrightarrow\ G_\Sigma(N) \models \bot && (\text{Theorem 45}) \\ &\Leftrightarrow\ \bot \in G_\Sigma(N) && (\text{propositional resolution sound and complete}) \\ &\Leftrightarrow\ \bot \in N \end{aligned}$$

■

Compactness of Predicate Logic

Theorem (Compactness Theorem for First-Order Logic)

Let Φ be a set of first-order Formulas. Φ unsatisfiable $\Leftrightarrow$ there exists $\Psi \subseteq \Phi$, $|\Psi| < \infty$, Ψ unsatisfiable.

Proof of .

"$\Leftarrow$": trivial.

"$\Rightarrow$": Let Φ be unsatisfiable and let N be the set of clauses obtained by Skolemization and CNF transformation of the formulas in Φ.

$\Rightarrow$ $Res^*(N)$ unsatisfiable
$\Rightarrow$ (Thm 47) $\bot \in Res^*(N)$
$\Rightarrow$ $\exists n \geq 0 : \bot \in Res^n(N)$
$\Rightarrow$ $\bot$ has finite resolution proof B of depth $\leq n$.

Choose Ψ als the subset of formulas in Φ such that the corresponding clauses contain the assumptions (leaves) of B. ■

Complexity of Unification

Literature:

1. Paterson, Wegman: Linear Unification, JCSS 17, 348-375 (1978)
2. Dwork, Kanellakis, Mitchell: On the sequential nature of unification, Journal Logic Prog. 1, 35-50 (1984)
3. Baader, Nipkow: Term rewriting and all that. Cambridge U. Press 1998, Capter 4.8

Theorem (Paterson, Wegman 1978)

Unifiability is decidable is in linear time. A most general unifiers can be computed sind in linearer time.

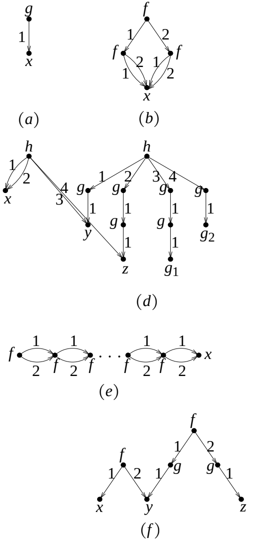
Theorem (Dwork, Kanellakis, Mitchell 1984)

Unifiability is log-space complete for P, that is, every problem in P can be reduced in log space to a unifiability problem.

As a consequence, unifiability can, most probably, not be efficiently parallelized.

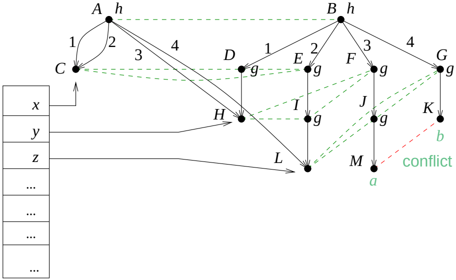
Acyclic Term Graphs

Terms and term sets as marked, ordered, acyclic graphs; each variable appears at most once



Propagation of Equality Constraints

Since variables occur at most once they don't appear as markings $m(u)$
$\Rightarrow$ binding table.



Rules (modulo symmetry of $\doteq$) for propagation of $\doteq$ in G:

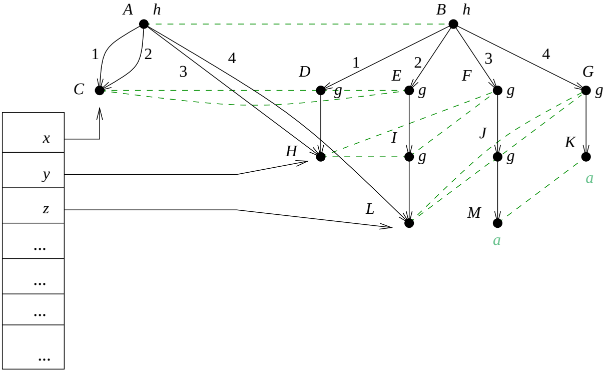
$$\begin{array}{rcl} u \doteq v & \Rightarrow & u.i \doteq v.i,\ 1 \le i \le \text{arity}(u) \\ u \doteq v,\ v \doteq w & \Rightarrow & u \doteq w \\ m(u) \neq m(v) & \Rightarrow & \perp \text{ (not unifiable)} \end{array}$$

If $G/\doteq$ contains a cycle (through oriented term-subterm edges) $\Rightarrow$ not unifiable.

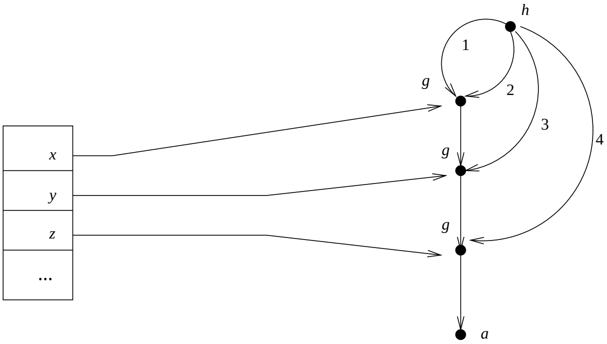
(Otherwise a term would have to be unified with a proper subterm of itself.)

Another Example

problem $h(x, x, y, z) \doteq h(g(y), g(g(z)), g(g(a)), g(a))$
after propagation:



After Forming the Quotient



the quotient graph is cycle-free
$\Rightarrow [g(g(g(a)))/x, g(g(a))/y, g(a)/z]$ is a mgu.

Analysis

For a unification problem with term graph of size n we obtain without much effort these complexity bounds:

- additional space in $O(log^2 n)$
- runtime in $O(n^3)$

In fact, at most n^2 edges can be generated by propagation, and each of those requires time $O(n)$ for a reachability test. For the quotient we have to compute the strongly connected components and then do the cycle test. This is both possible in time linear in the size of the graph, that is, in $O(n^2)$.

Matching

Let s, t be terms or atoms.

s matches t :

$s \leq t \;\; :\Leftrightarrow \;\;$ there exists a substitution σ s.t. $s\sigma = t$

(σ is called a matching substitution.)

$s \leq t \quad \Rightarrow \; \sigma =$ mgu(s, t), if $var(t) \cap var(s) = \emptyset$.

Theorem (Dwork, Kanellakis, Mitchell 1984)

Matching can be efficiently parallelized.

1.9 Ordered Resolution with Selection

Motivation: Search space for Res very large. Idea for improvement:

1. In the completeness proof (Model Existence Theorem 34) one only needs to resolve and factor maximal atoms $\Rightarrow$ order restrictions
2. Choice of negative literals don't-care $\Rightarrow$ selection

A selection function is a mapping

$$S : C \quad \mapsto \quad \text{set of occurrences of negative literals in } C$$

Example of selection with selected literals indicated as $\boxed{X}$:

$$\boxed{\neg A} \lor \neg A \lor B$$

$$\boxed{\neg B_0} \lor \boxed{\neg B_1} \lor A$$

Resolution Calculus $Res_S^\succ$

Let $\succ$ be an atom ordering and S a selection function. A literal L is called [strictly] maximal wrt. a clause C $:\Leftrightarrow$ there exists a ground substitution σ such that for all L' in C: $L\sigma \succeq L'\sigma$ $[L\sigma \succ L'\sigma]$.

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

if $\sigma = \text{mgu}(A, B)$ 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 $\neg B \lor D$ and $\neg B\sigma$ is maximal wrt. $D\sigma$.

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

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

Special Case: Propositional Logic

For ground clauses the resolution inference simplifies to

$$\frac{C \vee A \qquad D \vee \neg A}{C \vee D}$$

if

(i) $A \succ C$;
(ii) nothing is selected in C by. S;
(iii) $\neg A$ is selected in $D \vee \neg A$,
or else nothing is selected in $D \vee \neg A$ and $\neg A \succeq \max(D)$.

NB: For positive literals, $A \succ C$ is the same as $A \succ \max(C)$.

Search Spaces Become Smaller

1)	$A \vee B$	
2)	$A \vee \boxed{\neg B}$	
3)	$\neg A \vee B$	
4)	$\neg A \vee \boxed{\neg B}$	
5)	$B \vee B$	1&3
6)	B	5
7)	$\neg A$	6&4
8)	A	6&2
9)	$\bot$	8&7

we assume $A \succ B$ and S as indicated by $\boxed{X}$; the maximal atom in a clause is depicted in red.

With this ordering and selection function the refutation proceeds strictly deterministinally in this example. Generally, proof search will still be non-deterministic but the search space will be much smaller than with unrestricted resolution.

Avoiding Rotation Redundancy

From

$$\dfrac{\dfrac{C_1 \lor A \quad C_2 \lor \neg A \lor B}{C_1 \lor C_2 \lor B} \qquad C_3 \lor \neg B}{C_1 \lor C_2 \lor C_3}$$

we can obtain by rotation

$$\dfrac{C_1 \lor A \qquad \dfrac{C_2 \lor \neg A \lor B \quad C_3 \lor \neg B}{C_2 \lor \neg A \lor C_3}}{C_1 \lor C_2 \lor C_3}$$

another proof of the same clause. In large proofs many rotations are possible. However, if $A \succ B$, then the second proof does not fulfill the orderings restrictions.

Conclusion: In the presence of orderings restrictions (however one chooses $\succ$) no rotations are possible. In other words, orderings identify exactly one representant in any class of of rotation-equivalent proofs.

Lifting-Lemma for $Res_S^{\succ}$

Lemma

$$\begin{tikzcd} C \arrow[d, "\sigma"] & D \arrow[d, "\rho"] \\ C\sigma & D\rho \end{tikzcd}$$

$$\frac{C\sigma \qquad D\rho}{C'}$$

[propositional inference in $Res_S^{\succ}$]

and $S(C\sigma) \simeq S(C)$, $S(D\rho) \simeq S(D)$ (that is, “corresponding” literals are selected), implies that there exists a substitution τ such that

$$\frac{C \qquad D}{C''}$$

[Inference in $Res_S^{\succ}$]

$$\begin{tikzcd} C'' \arrow[d, "\tau"] \\ C' = C''\tau \end{tikzcd}$$

Saturation of General Clause Sets

Theorem

Let N *be a set of general clauses saturated under* $Res_S^{\succ}$*, i.e.* $Res_S^{\succ}(N) \subseteq N$*. Then there exists a selection function* S' *such that* $S|_N = S'|_N$ *and* $G_\Sigma(N)$ *is also saturated, i.e.,*

$$Res_{S'}^{\succ}(G_\Sigma(N)) \subseteq G_\Sigma(N).$$

Proof of W. e first define the selection function S' such that $S'(C) = S(C)$ for all clauses $C \in G_\Sigma(N) \cap N$, and for $C \in G_\Sigma(N) \setminus N$ we choose a fixed but arbitrary clause $D \in N$ mit $C \in G_\Sigma(D)$ and define $S'(C)$ to be those occurrences of literals which are the ground instances of the occurrences selected by S in D.
The rest of the proof proceeds as in the proof of Corollary 44 using the above lifting lemma. ■

Soundness and Refutational Completeness

Theorem

Let $\succ$ be an atom ordering and S a selection function such that $Res_S^{\succ}(N) \subseteq N$. Then

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

Proof of "$\Leftarrow$". trivial

"$\Rightarrow$"

(i) propositional level: construction of a candidate model I_N as for unrestricted resolution, except that clauses C in N that have selected literals are not productive, even when they are false in I_C and when their maximal atom occurs only once and positively.

(ii) general clauses: (i) + Corollary 53.

■

Craig-Interpolation

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 prop. variables occurring both in F and in G, and such that $F \models H$ and $H \models G$.

Proof of T. ranslate F and $\neg G$ into CNF. let N and M, resp., denote the resulting clause set. Choose an atom ordering $\succ$ for which the prop. variables that occur in F but not in G are maximal. Saturate N into N^* wrt. $Res_S^{\succ}$ with an empty selection function S . Then saturate $N^* \cup M$ wrt. $Res_S^{\succ}$ to derive $\perp$. 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. ■

Craig-Interpolation

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 prop. variables occurring both in F and in G, and such that $F \models H$ and $H \models G$.

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.

Global Redundancy: Rules for Simplifications and Deletion

Redundancy

- many proof attempts cannot be completed to proofs: dead ends in proof search
- one proof attempt may subsume another one

Rules for simplification of TP states N (that we would like to employ)

- Deletion of tautologies

$$N \cup \{C \vee A \vee \neg A\} \rhd N$$

- Deletion of subsumed clauses

$$N \cup \{C, D\} \rhd N \cup \{C\}$$

if $C\sigma \subseteq D$ (C subsumes D), and $C\sigma \neq D$ (subsumption is strict).

- Reduction (also called subsumption resolution)

$$N \cup \{C \vee L,\ \ D \vee C\sigma \vee \overline{L}\sigma\} \rhd N \cup \{C \vee L, D \vee C\sigma\}$$

Resolution Prover *RP*

3 clause sets: N(ew) containing new resolvents
P(rocessed) containing simplified resolvents
clauses get into O(ld) once their inferences have been computed

Strategy: Inferences will only be computed when there are no possibilites for simplification

Transition Rules for *RP*

Tautology elimination

$\boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O} \quad \rhd \quad \boldsymbol{N} \mid \boldsymbol{P} \mid \boldsymbol{O}$

if C is a tautology

Forward subsumption

$\boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O} \quad \rhd \quad \boldsymbol{N} \mid \boldsymbol{P} \mid \boldsymbol{O}$

if some $D \in \boldsymbol{P} \cup \boldsymbol{O}$ subsumes C

Backward subsumption

$\boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \cup \{D\} \mid \boldsymbol{O} \quad \rhd \quad \boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O}$

$\boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O} \cup \{D\} \quad \rhd \quad \boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O}$

if C strictly subsumes D

Forward reduction

$\boldsymbol{N} \cup \{C \vee L\} \mid \boldsymbol{P} \mid \boldsymbol{O} \quad \rhd \quad \boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O}$

if there exists $D \vee L' \in \boldsymbol{P} \cup \boldsymbol{O}$ such that $\overline{L} = L'\sigma$ and $D\sigma \subseteq C$

Transition Rules for RP (II)

Backward reduction

$\boldsymbol{N} \mid \boldsymbol{P} \cup \{C \vee L\} \mid \boldsymbol{O} \quad \triangleright \quad \boldsymbol{N} \mid \boldsymbol{P} \cup \{C\} \mid \boldsymbol{O}$

$\boldsymbol{N} \mid \boldsymbol{P} \mid \boldsymbol{O} \cup \{C \vee L\} \quad \triangleright \quad \boldsymbol{N} \mid \boldsymbol{P} \cup \{C\} \mid \boldsymbol{O}$

if there exists $D \vee L' \in \boldsymbol{N}$ such that $\overline{L} = L'\sigma$ and $D\sigma \subseteq C$

Clause processing

$\boldsymbol{N} \cup \{C\} \mid \boldsymbol{P} \mid \boldsymbol{O} \quad \triangleright \quad \boldsymbol{N} \mid \boldsymbol{P} \cup \{C\} \mid \boldsymbol{O}$

Inference computation

$\emptyset \mid \boldsymbol{P} \cup \{C\} \mid \boldsymbol{O} \quad \triangleright \quad \boldsymbol{N} \mid \boldsymbol{P} \mid \boldsymbol{O} \cup \{C\}$, mit $\boldsymbol{N} = Res_S^{\succ}(\boldsymbol{O} \cup \{C\})$

Soundness and Completeness

Theorem

$$N \models \bot \quad \Leftrightarrow \quad N \mid \emptyset \mid \emptyset \quad \overset{*}{\rhd} \quad N' \cup \{\bot\} \mid \_ \mid \_$$

Proof in
L. Bachmair, H. Ganzinger: Resolution Theorem Proving
appeared in the Handbook on Automated Theorem Proving, 2001
Basis for the completeness proof is a formal notion of redundancy as defined subsequently.

A Formal Notion of Redundancy

Let N be a set of ground clauses and C a ground clause (not necessarliy in N).

C is called redundant in N $:\Leftrightarrow$ there exists $C_1, \ldots, C_n \in N,\ n \geq 0$:
$C_i \prec C$ and $C_1, \ldots, C_n \models C$

Redundancy for general clauses:
C is called redundant in N $:\Leftrightarrow$ $C\sigma$ redundant in $G_\Sigma(N)$,
for all ground instances $C\sigma$ of C

Intuition: Redundant clauses are no minimal counterexamples for any interpretation.

NB: The same ordering $\succ$ is used both for ordering restrictions and for redundancy.

Examples of Redundancy

Theorem

- *C tautology (i.e., $\models C$)* $\Rightarrow$ *C redundant in any set N.*
- *$C\sigma \subset D$* $\Rightarrow$ *D redundant in $N \cup \{C\}$*
  (strict[a] Subsumption: $N \cup \{C, D\} \rhd N \cup \{C\}$)
- *$C\sigma \subseteq D$* $\Rightarrow$ *$D \vee \overline{L}\sigma$ redundant in $N \cup \{C \vee L,\ D\}$*

An application of the latter is reduction (subsumption resolution) in RP

[a] cf. RP for cases when clauses can be deleted even if subsumption is not strict.

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

Let N be saturated up to redundancy. Then

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

Proof of [. 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 54. ■

Monotonicity Properties of Redundancy

Theorem

(i) $N \subseteq M \ \Rightarrow\ Red(N) \subseteq Red(M)$
(ii) $M \subseteq Red(N) \ \Rightarrow\ Red(N) \subseteq Red(N \setminus M)$

Proof: Exercise.
We conclude that redundancy is preserved when, during a theorem proving process, one adds (derives) new clauses or deletes redundant clauses.
The theorems 59 and 60 are the basis for the completeness proof of our prover RP.

Hyperresolution (Robinson 65)

We define an improved version of hyperresolution with ordering restrictions and selection. As for Res the calculus is parameterized by an atom ordering $\succ$ and a selection function S.

$$\frac{C_1 \vee A_1 \quad \ldots \quad C_n \vee A_n \qquad \neg B_1 \vee \ldots \vee \neg B_n \vee D}{(C_1 \vee \ldots \vee C_n \vee D)\sigma}$$

with $\sigma = \text{mgu}(A_1 \doteq B_1, \ldots, A_n \doteq B_n)$, if

(i) $A_i\sigma$ strictly maximal wrt. $C_i\sigma$, $1 \le i \le n$;

(ii) nothing is selected in C_i;

(iii) the indicated occurrences of the $\neg B_i$ are exactly the ones selected by S, or else nothing is selected in the right premise and $n = 1$ and $\neg B_1\sigma$ is maximal wrt. $D\sigma$.

HR needs to be complemented by a factoring inference as for $Res_S^{\succ}$.

Hyperresolution (ctnd)

Hyperresolution can be simulated by iterated binary resolution. However this yields intermediate clauses which HR might not derive, and many of them might not be extendable into a full HR inference.

There are *many* more variants of resolution.

We refer to [Bachmair, Ganzinger: Resolution Theorem Proving] for further reading.