Predicate Calculus

Formal Methods
Lecture 6

Farn Wang
Dept. of Electrical Engineering
National Taiwan University

Predicate Logic

- Invented by Gottlob Frege (1848–1925).
- Predicate Logic is also called “first-order logic”.

“Every good mathematician is at least half a philosopher, and every good philosopher is at least half a mathematician.”

Motivation

There are some kinds of human reasoning that we can't do in propositional logic.

- For example:

  Every person likes ice cream.

  Billy is a person.

  Therefore, Billy likes ice cream.

- In propositional logic, the best we can do is $A \wedge B \rightarrow C$, which isn't a tautology.
  - We've lost the internal structure.

Motivation

- We need to be able to refer to objects.
- We want to symbolize both a claim and the object about which the claim is made.
- We also need to refer to relations between objects,
  - as in "Waterloo is west of Toronto".
- If we can refer to objects, we also want to be able to capture the meaning of every and some of.
- The predicates and quantifiers of predicate logic allow us to capture these concepts.

Apt-pet

- An apartment pet is a pet that is small
- Dog is a pet
- Cat is a pet
- Elephant is a pet
- Dogs and cats are small.
- Some dogs are cute
- Each dog hates some cat
- Fido is a dog

$$\forall x\, small(x) \wedge pet(x) \supset aptPet(x)$$
$$\forall x\, dog(x) \supset pet(x)$$
$$\forall x\, cat(x) \supset pet(x)$$
$$\forall x\, elephant(x) \supset pet(x)$$
$$\forall x\, dog(x) \supset small(x)$$
$$\forall x\, cat(x) \supset small(x)$$
$$\exists x\, dog(x) \wedge cute(x)$$
$$\forall x\, dog(x) \supset \exists y\, cat(y) \wedge hates(x, y)$$
$$dog(fido)$$

5

Quantifiers

- Universal quantification ($\forall$) corresponds to finite or infinite conjunction of the application of the predicate to all elements of the domain.
- Existential quantification ($\exists$) corresponds to finite or infinite disjunction of the application of the predicate to all elements of the domain.
- Relationship between $\forall$ and $\exists$:
  - $\exists$x.P(x) is the same as $\neg\forall$x. $\neg$P(x)
  - $\forall$x.P(x) is the same as $\neg\exists$x. $\neg$P(x)

6

Functions

- Consider how to formalize:
  Mary's father likes music
  One possible way: *∃x(f(x, Mary)∧Likes(x,Music))*
  which means: Mary has at least one father and he likes music.
- We'd like to capture the idea that Mary only has one father.
  - We use functions to capture the single object that can be in relation to another object.
  - Example: *Likes(father(Mary),Music)*
- We can also have *n*-ary functions.

7

Predicate Logic

- syntax (well-formed formulas)
- semantics
- proof theory
  - axiom systems
  - natural deduction
  - sequent calculus
  - resolution principle

8

Predicate Logic: Syntax

The syntax of predicate logic consists of:

- constants
- variables x, y, …
- functions f(), g(), …
- predicates P(), Q(), …
- logical connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$
- quantifiers $\forall$, $\exists$
- punctuations: , . ( )

9

Predicate Logic: Syntax

Definition. Terms are defined inductively as follows:

- Base cases
  - Every constant is a term.
  - Every variable is a term.
- inductive cases
  - If $t_1, t_2, t_3, \ldots, t_n$ are terms then $f(t_1, t_2, t_3, \ldots, t_n)$ is a term, where *f* is an n-ary function.
- Nothing else is a term.

10

Predicate Logic
- syntax

Definition. Well-formed formulas (wffs) are defined inductively as follows:

- Base cases:
  - $P(t_1,t_2,t_3,\ldots,t_n)$ is a wff, where t_i is a term, and P is an n-ary predicate. These are called atomic formulas.
- inductive cases:
  - If A and B are wffs, then so are

    $$\neg A,\ A \wedge B,\ A \vee B,\ A \Rightarrow B,\ A \Leftrightarrow B$$
  - If *A* is a wff, so is $\exists x.\ A$
  - If *A* is a wff, so is $\forall x.\ A$
- Nothing else is a wff.

We often omit the brackets using the same precedence rules as propositional logic for the logical connectives.

Scope and Binding of Variables (I)

- Variables occur both in nodes next to quantifiers and as leaf nodes in the parse tree.
- A variable x is bound if starting at the leaf of x, we walk up the tree and run into a node with a quantifier and x.
- A variable x is free if starting at the leaf of x, we walk up the tree and don't run into a node with a quantifier and x.

$$\forall x.(\forall x.(P(x) \wedge Q(x))) \Rightarrow (\neg P(x) \vee Q(y))$$

Scope and Binding of Variables (I)

The scope of a variable x is the subtree starting at the node with the variable and its quantifier (where it is bound) minus any subtrees with $\forall x$ or $\exists x$ at their root.

Example:

A wff is closed if it contains no free occurrences of any variable.

$$\forall x.(\forall x.(P(x) \wedge Q(x))) \Rightarrow (\neg P(x) \vee Q(y))$$

scope of this x

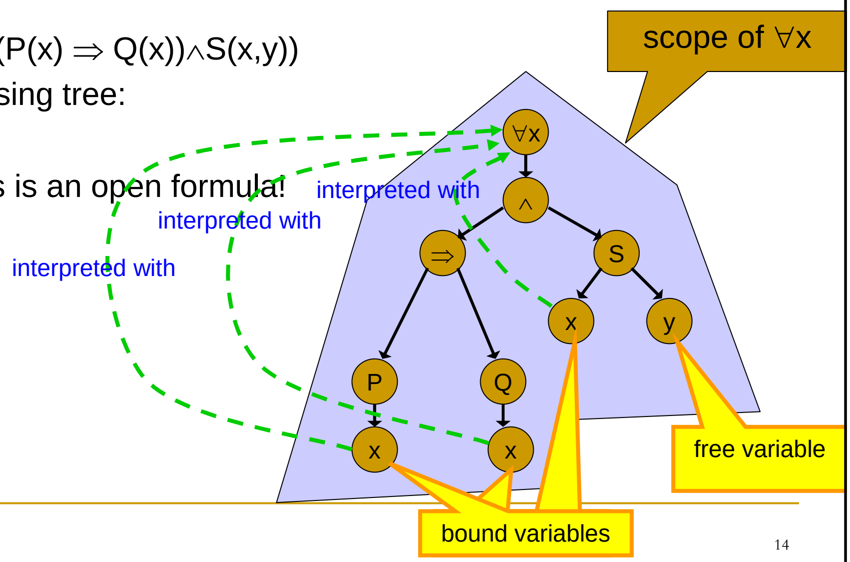
scope of this x

Scope and Binding of Variables

$\forall x((P(x) \Rightarrow Q(x)) \wedge S(x,y))$

Parsing tree:

This is an open formula!

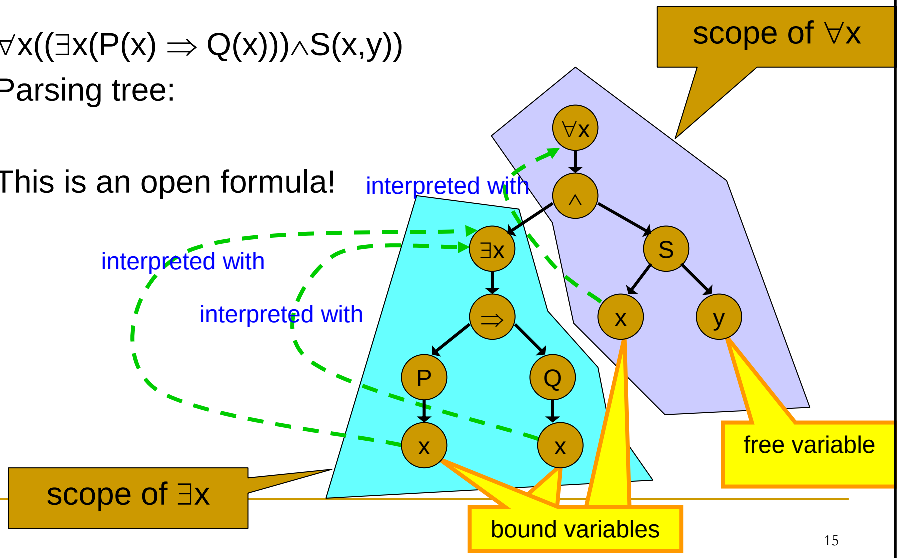


Scope and Binding of Variables

∀x((∃x(P(x) ⇒ Q(x)))∧S(x,y))

Parsing tree:

This is an open formula!



Substitution

Variables are place holders.

- Given a variable x, a term t and a formula P, we define $P[t / x]$ to be the formula obtained by replacing each free occurrence of variable x in P with t.
- We have to watch out for variable captures in substitution.

Substitution

In order not to mess up with the meaning of the original formula, we have the following restrictions on substitution.

- Given a term t, a variable x and a formula P,

  "t is not free for x in P"

  if
  - x in a scope of $\forall y$ or $\exists y$ in A; and
  - *t* contains a free variable *y*.
- Substitution $P[t / x]$ is allows only if *t* is free for *x* in *P*.

17

Substitution

[f(y)/x] not allowed since meaning of formulas messed up.

Example:

$\forall y(mom(x) \wedge dad(f(y))) \equiv \forall z(mom(x) \wedge dad(f(z)$

But

$(\forall y(mom(x) \wedge dad(y)))[f(y)/x] = \forall y(mom(f(y)) \wedge dad(f(y)))$

~~*equivalent*~~

$(\forall z(mom(x) \wedge dad(z)))[f(y)/x] = \forall z(mom(f(y)) \wedge dad(f(z)))$

18

Predicate Logic: Semantics

- Recall that a semantics is a mapping between two worlds.
- A model for predicate logic consists of:
  - a non-empty domain of objects: D_I
  - a mapping, called an interpretation that associates the terms of the syntax with objects in a domain
- It's important that D_I be non-empty, otherwise some tautologies wouldn't hold such as $(\forall x.A(x)) \Rightarrow (\exists x.A(x))$

19

Interpretations (Models)

- a fixed element $c' \in D_I$ to each constant c of the syntax
- an n-ary function $f':D_I^n \rightarrow D_I$ to each n-ary function, f, of the syntax
- an n-ary relation $R' \subseteq D_I^n$ to each n-ary predicate, R, of the syntax

20

Example of a Model

- Let's say our syntax has a constant c, a function f (unary), and two predicates P, and Q (both binary).

Example: $P(c,f(c))$

In our model, choose the domain to be the natural numbers

- $I(c)$ is 0.
- $I(f)$ is suc, the successor function.
- $I(P)$ is `<'
- $I(Q)$ is `='

21

Example of an Model

What's the meaning of $P(c,f(c))$ in this model?

$$
\begin{aligned}
I(P(c, f(c))) &= I(c) < I(f(c)) \\
&= 0 < suc(I(c)) \\
&= 0 < suc(0) \\
&= 0 < 1
\end{aligned}
$$

Which is true.

22

Valuations

Definition.
A valuation v, in an interpretation I, is a function from the terms to the domain D_I such that:

- $\nu(c) = I(c)$
- $\nu(f(t_1,\ldots,t_n)) = f'(\nu(t_1),\ldots, \nu(t_n))$
- $\nu(x) \in D_I$, i.e., each variable is mapped onto some element in D_I

23

Example of a Valuation

- D_I is the set of Natural Numbers
- g is the function +
- h is the function *suc*
- c (constant) is 3
- y (variable) is 1

$$\begin{aligned} v(g(h(c),y)) &= v(h(c)) + v(y) \\ &= suc(v(c)) + 1 \\ &= suc(3) + 1 \\ &= 5 \end{aligned}$$

24

Workout

- D_I is the set of Natural Numbers
- g is the function +
- h is the function *suc*
- c (constant) is 3
- y (variable) is 1

$$\nu(h(h(g(h(y),g(h(y),h(c)))))) = ?$$

On(A,B)	False
Clear(B)	True
On(C,Fl)	True
On(C,Fl)∧¬On(A,B)	True

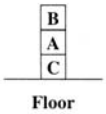


Figure 15.1

A Configuration of Blocks

Predicate Calculus	World
A	**A**
B	**B**
C	**C**
Fl	**Floor**
On	**On** = {< **B**, **A** >, < **A**, **C** >, < **C**, **Floor** >}
Clear	**Clear** = {< **B** >}

Table 15.1

A Mapping between Predicate Calculus and the World

Workout

Interpret the following formulas with respect to the world (model) in the previous page.

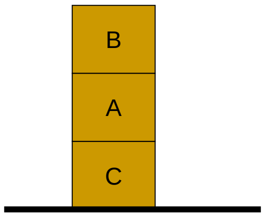
$On(A,Fl) \Rightarrow Clear(B)$

$Clear(B) \wedge Clear(C) \Rightarrow On(A,Fl)$

$Clear(B) \vee Clear(A)$

$Clear(B)$

$Clear(C)$



27

Konwoledge

Does the following knowledge base (set of formulae) have a model ?

$On(A,Fl) \Rightarrow Clear(B)$

$Clear(B) \wedge Clear(C) \Rightarrow On(A,Fl)$

$Clear(B) \vee Clear(A)$

$Clear(B)$

$Clear(C)$

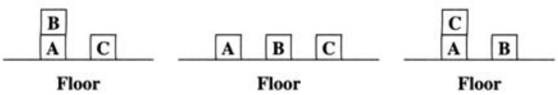


Figure 15.2

Three Blocks-World Situations

28

An example

$(\forall x)[On(x,C) \Rightarrow \neg Clear(C)]$

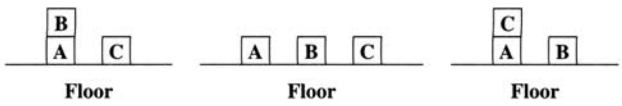


Figure 15.2

Three Blocks-World Situations

29

Closed Formulas

- Recall: A wff is closed if it contains no free occurrences of any variable.
- We will mostly restrict ourselves to closed formulas.
- For formulas with free variables, close the formula by universally quantifying over all its free variables.

30

Validity (Tautologies)

- **Definition.** A predicate logic formula is satisfiable if there is an interpretation and there is a valuation that satisfies the formula (i.e., in which the formula returns T).
- **Definition.** A predicate logic formula is logically valid (tautology) if it is true in every interpretation.
  - It must be satisfied by every valuation in every interpretation.
- **Definition.** A wff, A, of predicate logic is a contradiction if it is false in every interpretation.
  - It must be false in every valuation in every interpretation.

Satisfiability, Tautologies, Contradictions

- A closed predicate logic formula, is satisfiable if there is an interpretation I in which the formula returns true.
- A closed predicate logic formula, A, is a tautology if it is true in every interpretation.

$$\models A$$

- A closed predicate logic formula is a contradiction if it is false in every interpretation.

Tautologies

- How can we check if a formula is a tautology?
- If the domain is finite, then we can try all the possible interpretations (all the possible functions and predicates).
- But if the domain is infinite? Intuitively, this is why a computer cannot be programmed to determine if an arbitrary formula in predicate logic is a tautology (for all tautologies).
- Our only alternative is proof procedures!
- Therefore the soundness and completeness of our proof procedures is very important!

33

Semantic Entailment

Semantic entailment has the same meaning as it did for propositional logic.

$$\phi_1, \phi_2, \phi_3 \vDash \psi$$

means that if $v(\phi_1) = \mathrm{T}$ and $v(\phi_2) = \mathrm{T}$ and $v(\phi_3) = \mathrm{T}$ then $v(\psi) = \mathrm{T}$, which is equivalent to saying

$$(\phi_1 \wedge \phi_2 \wedge \phi_3) \Rightarrow \psi$$

is a tautology, i.e.,

$$(\phi_1, \phi_2, \phi_3 \vDash \psi) \equiv ((\phi_1 \wedge \phi_2 \wedge \phi_3) \Rightarrow \psi)$$

34

An Axiomatic System for Predicate Logic

FO_AL: An extension of the axiomatic system for propositional logic. Use only: $\Rightarrow, \neg, \forall$

$A \Rightarrow (B \Rightarrow A)$

$(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$

$(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$

$\forall x.A(x) \Rightarrow A(t)$, where t is free for x in A

$\forall x.(A \Rightarrow B) \Rightarrow (A \Rightarrow (\forall x.B))$, where A contains no free occurrences of x

35

FO_AL Rules of Inference

Two rules of inference:

- (modus ponens - MP) From A and $A \Rightarrow B$, B can be derived, where A and B are any well-formed formulas.
- (generalization) From A, $\forall x.A$ can be derived, where A is any well-formed formula and x is any variable.

36

Soundness and Completeness of FO_AL

- FO_AL is sound and complete.
- Completeness was proven by Kurt Gödel in 1929 in his doctoral dissertation.
- Predicate logic is not decidable

37

Deduction Theorem

- **Theorem.** If $H \cup \{A\} \vdash_{ph} B$ by a deduction containing no application of generalization to a variable that occurs free in A, then $H \vdash_{ph} A \Rightarrow B$
- **Corollary.** If A is closed and if $H \cup \{A\} \vdash_{ph} B$ then $H \vdash_{ph} (A \Rightarrow B)$

38

Proof by Refutation

- A closed formu is a its negation is a c
- In o alid iff its negati
- T alent to pro se
- T S becomes to check if ther pr ation for {P1,…, Pn, ¬S} .

How many interpretations are there ?

Counterexamples

- How can we show a formula is not a tautology?
- Provide a counterexample. A counterexample for a closed formula is an interpretation in which the formula does not have the truth value T.

Example

Prove $\forall x.\forall y.A \vdash_{ph} \forall y.\forall x.A$

1	$\forall x.\forall y.A$	premise
2	$\forall x.\forall y.A \Rightarrow \forall y.A$	Ax4
3	$\forall y.A$	MP 1, 2
4	$\forall y.A \Rightarrow A$	Ax4
5	A	MP 3, 4
6	$\forall x.A$	Gen of 5
7	$\forall y.\forall x.A$	Gen of 6

Workout: Counterexamples

Show that $(\forall x.P(x) \vee Q(x)) \Leftrightarrow ((\forall x.P(x)) \vee (\forall x.Q(x)))$ is not a tautology by constructing a model that makes the formula false.

What does '*first-order*' mean?

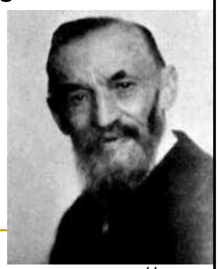
- We can only quantify over variables.
- In higher-order logics, we can quantify over functions, and predicates.
  - For example, in second-order logic, we can express the induction principle:

$$\forall P.(P(0) \wedge (\forall n.P(n) \Rightarrow P(n+1))) \Rightarrow (\forall n.P(n))$$

- Propositional logic can also be thought of as zero-order.

A rough timeline in ATP … (1/3)

450B.C.	Stoics	propositional logic (PL), inference (maybe)
322B.C.	Aristotle	``syllogisms" (inference rules), quantifiers
1565	Cardano	probability theory (PL + undertainty)
1646 -1716	Leibniz	research for a general decision procedure to check the validty of formulas
1847	Boole	PL (again)
1879	Frege	first-order logic (FOL)
1889	Peano	9 axioms for natural numbers



A rough timeline in ATP .

- To formalize all existing theories to a finite, complete, and consistent set of axioms.
- decision procedures for all mathematical theories
- 23 open problems.

Resolve the 2nd Hilbert's problem (in the theory of N)

	bert	Hilbert's program
	ttgenstein	proof by truth tables
	ödel	completeness theorem of FOL
	brand	a proof procedure for FOL based on propositionalization
1931	Gödel	incompleteness theorems for the consistency of Peano axioms
1936	Gentzen	a proof for the consisitency of Peano axioms in set theory
1936	Church, Turing	undecidability of FOL
1958	Gödel	a method to prove the co axioms with type theory

Who is to

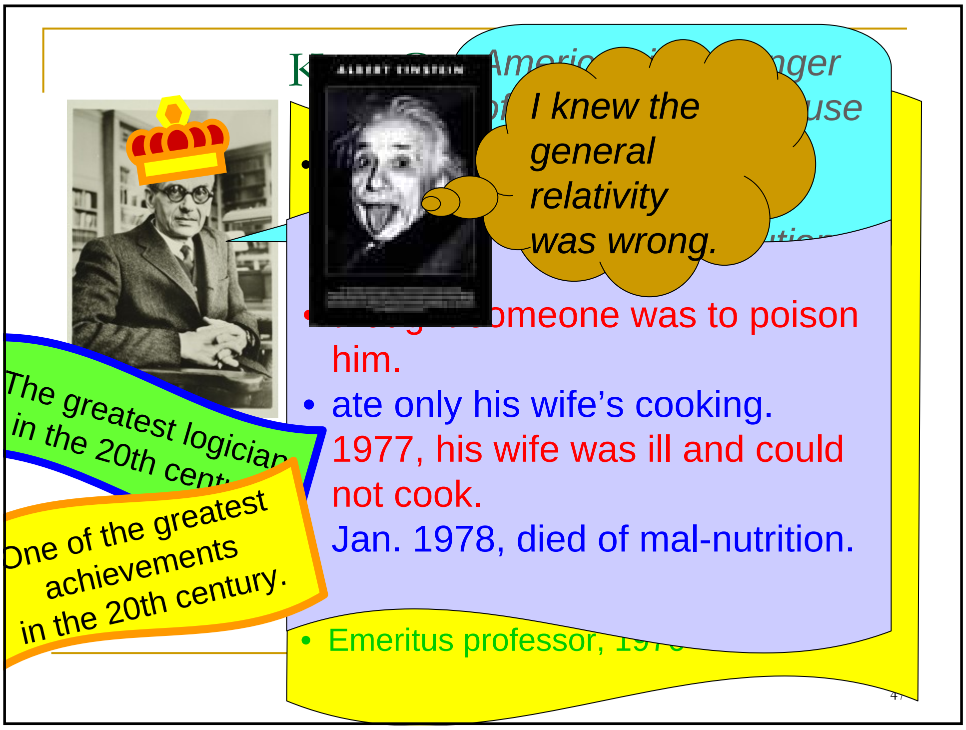
Is type theory consistent ?

45

A rough timeline in ATP ... (3/3)

1954	Davis	First machine-generated proof
1955	Beth, Hintikka	Semantic Tableaus
1957	Newell, Simon	First machine-generated proof in Logic Calculus
1957	Kangar, Prawitz	Lazy substitution by free (dummy) Vars
1958	Prawitz	First prover for FOL
1959	Gilmore Wang	More provers
1960	Davis Putnam, Longman	Davis-Putnam Procedure
1963	Robinson	Unification, resolution

46



2007/04/03 stopped here.

48

Predicate Logic: Natural Deduction

Extend the set of rules we used for propositional logic with ones to handle quantifiers.

Universal Quantifi cation

forall-elimination

$$\frac{\forall x.P}{P[t/x]}\,\forall e$$

forall-introduction

$$\frac{\begin{bmatrix} x_0 \\ \vdots \\ P[x_0/x] \end{bmatrix}}{\forall x.P}\,\forall i$$

x_0 must be arbitrary, meaning it doesn't appear outside the subproof. t must be free for x in P.

49

Predicate Logic: Natural Deduction

Existential Quantifi cation

exists-introduction

$$\frac{P[t/x]}{\exists x.P}\,\exists i$$

$$\frac{\exists x.\,P \quad \begin{bmatrix} x_0 \quad P[x_0/x] \quad \text{assumption} \\ \vdots \\ Q \end{bmatrix}}{Q}\,\exists e$$

Informally: If we know that the predicate is true for some value, and using an arbitrary variable, we derive that a formula holds, then we can conclude that the formula holds.

x_0 must be arbitrary. t must be free for x in P.

50

Example

Show $\forall x.P(x) \Rightarrow Q(x), \forall x.P(x) \vdash_{ND} \forall x.Q(x)$

1	$\forall x.\ P(x) \Rightarrow Q(x)$	premise
2	$\forall x.\ P(x)$	premise
3	x_0	
4	$P(x_0) \Rightarrow Q(x_0)$	$\forall$e 1
5	$P(x0)$	$\forall$e 2
6	$Q(x0)$	$\Rightarrow$ e 4, 5
7	$\forall x.\ Q(x)$	$\forall$i 3 – 6

Workout

- Show $P(a), \forall x.P(x) \Rightarrow \neg Q(x) \vdash_{ND} \neg Q(a)$
- Show $\neg \forall x.P(x) \vdash_{ND} \exists x.\neg P(x)$

Proof by Refutation

- To prove {P1,…, Pn} ⊨ S is equivalent to prove that there is no interpretation for
  {P1,…, Pn, ¬S} .
- But there are infinitely many interpretations!
- Can we limit the range of interpretations ?
- Yes, Herbrand interpretations!

Herbrand's theorem
- *Herbrand universe of a formula S*

- Let H_0 be the set of constants appearing in S.
  - If no constant appears in S, then H_0 is to consist of a single constant, $H_0=\{a\}$.
- For i=0,1,2,…

$H_{i+1}=H_i \cup \{f^n(t_1,\ldots,t_n) |$ f is an n-place function in S; $t_1,\ldots,t_n \in H_i\}$

- H_i is called the i-level constant set of S.
- H_∞ is the *Herbrand universe* of S.

Herbrand's theorem

- Herbrand universe of a formula S

Example 1: $S=\{P(a),\sim P(x)\vee P(f(x))\}$

- $H_0=\{a\}$
- $H_1=\{a,f(a)\}$
- $H_2=\{a,f(a),f(f(a))\}$
- .
- .
- $H_\infty=\{a,f(a),f(f(a)),f(f(f(a))),\ldots\}$

Herbrand's theorem

- Herbrand universe of a formula S

Example 2: $S=\{P(x)\vee Q(x),R(z),T(y)\vee\sim W(y)\}$

- There is no constant in S, so we let $H_0=\{a\}$
- There is no function symbol in S, hence $H=H_0=H_1=\ldots=\{a\}$

Example 3: $S=\{P(f(x),a,g(y),b)\}$

- $H_0=\{a,b\}$
- $H_1=\{a,b,f(a),f(b),g(a),g(b)\}$
- $H_2=\{a,b,f(a),f(b),g(a),g(b),f(f(a)),f(f(b)),f(g(a)),f(g(b)),g(f(a)),g(f(b)),g(g(a)),g(g(b))\}$
- …

Herbrand's theorem

- *Herbrand universe of a formula S*

Expression

- a term, a set of terms, an atom, a set of atoms, a literal, a clause, or a set of clauses.

Ground expressions

- *expressions without variables*.

It is possible to use a ground term, a ground atom, a ground literal, and a ground clause – this means that no variable occurs in respective expressions.

Subexpression of an expression *E*

- an expression that occurs in *E*.

Herbrand's theorem

- *Herbrand base of a formula S*

- Ground atoms $P^n(t_1,\ldots,t_n)$
  - P^n is an *n*-place predicate occurring in *S*,
  - $t_1,\ldots,t_n \in H_\infty$
- Herbrand base of *S* (atom set)
  - the set of all ground atoms of S
- Ground instance of *S*
  - obtained by replacing variables in *S* by members of the Herbrand universe of *S*.

Herbrand's theorem

- Herbrand universe & base of a formula S

Example

- $S=\{P(x),Q(f(y))\vee R(y)\}$
- $C=P(x)$ is a clause in S
- $H=\{a,f(a),f(f(a)),\ldots\}$ is the Herbrand universe of S.
- $P(a)$, Q(f(a)), Q(a), R(a), R(f(f(a))), and $P(f(f(a)))$ are ground atoms of C.

Workout

$\{P(x), Q(g(x,y),a)\vee R(f(x))\}$

- please construct the set of ground terms
- please construct the set of ground atoms

Herbrand's theorem
- *Herbrand interpretation of a formula S*

- S, a set of clauses.
  - i.e., a conjunction of the clauses
- *H,* the Herbrand universe of *S* and
- *H*-interpretation $\mathscr{I}$ of *S*
  - $\mathscr{I}$ maps all constants in *S* to themselves.
  - Forall *n*-place function symbol f and $h_1,\ldots,h_n$ elements of *H*,

$$\mathscr{I}(f\ (h_1,\ldots,h_n)\ ) = f(h_1,\ldots,h_n)$$

61

Herbrand's theorem
- *Herbrand interpretation of a formula S*

- There is no restriction on the assignment to each *n*-place predicate symbol in *S.*
- Let $A=\{A_1,A_2,\ldots,A_n,\ldots\}$ be the atom set of *S.*
- An *H*-interpretation *I* can be conveniently represented as a subset of A.
  - If $A_j \in$ I, then A_j is assigned “true”,
  - otherwise A_j is assigned “false”.

62

Herbrand's theorem

- Herbrand interpretation of a formula S

Example: $S=\{P(x)\vee Q(x),R(f(y))\}$

- The Herbrand universe of S is
  $H=\{a,f(a),f(f(a)),\ldots\}$.
- Predicate symbols: P, Q, R.
- The atom set of S:
  $A=\{P(a),Q(a),R(a),P(f(a)),Q(f(a)),R(f(a)),\ldots\}$.
- Some H-interpretations for S:
  - $I_1=\{P(a),Q(a),R(a),P(f(a)),Q(f(a)),R(f(a)),\ldots\}$
  - $I_2=\varnothing$
  - $I_3=\{P(a),Q(a),P(f(a)),Q(f(a)),\ldots\}$

63

Herbrand's theorem

- Herbrand interpretation of a formula S

- An interpretation of a set S of clauses does not necessarily have to be defined over the Herbrand universe of S.
- Thus an interpretation may not be an H-interpretation.

Example:

- $S=\{P(x),Q(y,f(y,a))\}$
- $D=\{1,2\}$

64

Herbrand's theorem

- *Herbrand interpretation of a formula S*

- But Herbrand is conceptually general enough.

Example (cont.) *S*={*P*(*x*),*Q*(*y*,*f*(*y*,*a*))}

- *D*={1,2}
- – an interpretation of *S*:

a	*f*(1,1)	*f*(1,2)	*f*(2,1)	*f*(2,2)
2	1	2	2	1

P(1)	*P*(2)	*Q*(1,1)	*Q*(1,2)	*Q*(2,1)	*Q*(2,2)
T	*F*	*F*	*T*	*F*	*T*

65

Herbrand's theorem

- *Herbrand interpretation of a formula S*

- But Herbrand is conceptually general enough.

Example (cont.) – we can define an *H*-interpretation *I** corresponding to *I*.

- First we find the atom set of *S*
  - *A*={*P*(*a*),*Q*(*a*,*a*),*P*(*f*(*a*,*a*)),*Q*(*a*,*f*(*a*,*a*)),*Q*(*f*(*a*,*a*),*a*),*Q*(*f*(*a*,*a*),*f*(*a*,*a*)),…}
- Next we evaluate each member of *A* by using the given table
  - *P*(*a*)=*P*(2)=*F*
  - *Q*(*a*,*a*)=*Q*(2,2)=*T*
  - *P*(*f*(*a*,*a*))=*P*(*f*(2,2))=*P*(1)=*T*
  - *Q*(*a*,*f*(*a*,*a*))=*Q*(2,*f*(2,2))=*Q*(2,1)=*F*
  - *Q*(*f*(*a*,*a*),*a*)=*Q*(*f*(2,2),2)=*Q*(1,2)=*T*
  - *Q*(*f*(*a*,*a*),*f*(*a*,*a*))=*Q*(*f*(2,2),*f*(2,2))=*Q*(1,1)=*T*

a	*f*(1,1)	*f*(1,2)	*f*(2,1)	*f*(2,2)
2	1	2	2	1

P(1)	*P*(2)	*Q*(1,1)	*Q*(1,2)	*Q*(2,1)	*Q*(2,2)
T	*F*	*F*	*T*	*F*	*T*

- *Then I**={*Q*(*a*,*a*),*P*(*f*(*a*,*a*)),*Q*(*f*(*a*,*a*),*a*),…}.

66

Herbrand's theorem
- *Herbrand interpretation of a formula S*

- If there is no constant in *S*, the element a used to initiate the Herbrand universe of *S* can be mapped into any element of the domain *D*.
- If there is more than one element in *D*, then there is more than one *H*-interpretation corresponding to *I*.

Herbrand's theorem
- *Herbrand interpretation of a formula S*

- Example: *S*={*P*(*x*),*Q*(*y*,*f*(*y*,*z*))}, *D*={1,2}

f(1,1)	*f*(1,2)	*f*(2,1)	*f*(2,2)
1	2	2	1

P(1)	*P*(2)	*Q*(1,1)	*Q*(1,2)	*Q*(2,1)	*Q*(2,2)
T	*F*	*F*	*T*	*F*	*T*

- Two *H*-interpretations corresponding to *I* are:
  - *I**={*Q*(a,a),*P*(*f*(a,a)),*Q*(*f*(a,a),a),…} if *a*=2,
  - *I**={*P*(a),*P*(*f*(a,a)),…} if *a*=1.

Herbrand's theorem

- *Herbrand interpretation of a formula S*

Definition: Given an interpretation *I* over a domain *D*, an *H*-interpretation *I** corresponding to *I* is an *H*-interpretation that satisfies the condition:

- Let $h_1,\ldots,h_n$ be elements of *H* (the Herbrand universe of *S*).
- Let every h_i be mapped to some d_i in *D*.
- If $P(d_1,\ldots,d_n)$ is assigned *T* (*F*) by *I*, then $P(h_1,\ldots,h_n)$ is also assigned *T*(*F*) in *I**.

Lemma: If an interpretation *I* over some domain *D* satisfies a set *S* of clauses, then any *H*-interpretation *I** corresponding to *I* also satisfies *S*.

69

Herbrand's theorem

A set S of clauses is unsatisfiable if and only if S is false under all the H-interpretations of S.

- We need consider only *H*-interpretations for checking whether or not a set of clauses is unsatisfiable.
- Thus, whenever the term "interpretation" is used, a *H*-interpretation is meant.

70

Herbrand's theorem

Let ∅ denote an empty set. Then:

- A ground instance *C*' of a clause *C* is satisfied by an interpretation *I* if and only if there is a ground literal *L*' in *C*' such that *L*' is also in *I*, i.e. $C' \cap I \neq \emptyset$.
- A clause *C* is satisfied by an interpretation *I* if and only if every ground instance of *C* is satisfied by *I*.
- A clause *C* is falsified by an interpretation *I* if and only if there is at least one ground instance *C*' of *C* such that *C*' is not satisfied by *I*.
- A set *S* of clauses is unsatisfiable if and only if for every interpretation *I* there is at least one ground instance *C*' of some clause *C* in *S* such that *C*' is not satisfied by *I*.

Herbrand's theorem

Example: Consider the clause $C=\sim P(x) \vee Q(f(x))$. Let I_1, I_2, and I_3 be defined as follows:

- $I_1 = \emptyset$
- $I_2=\{P(a),Q(a),P(f(a)),Q(f(a)),P(f(f(a))),Q(f(f(a))),\ldots\}$
- $I_3=\{P(a),P(f(a)),P(f(f(a))),\ldots\}$

C is satisfied by I_1 and I_2, but falsified by I_3.

Example: $S=\{P(x),\sim P(a)\}$.

The only two *H*-interpretations are:

- $I_1=\{P(a)\}$,
- $I_2= \emptyset$.

S is falsified by both *H*-interpretations and therefore is unsatisfiable.

Resolution Principle - Clausal Forms

Clauses are universally quantified disjunctions of literals;

all variables in a clause are universally quantified

$$(\forall x_1, \ldots, x_n)(l_1 \vee \ldots \vee l_n)$$

written as

$$l_1 \vee \ldots \vee l_n$$

or

$$\{l_1, \ldots, l_n\}$$

73

Resolution Principle - Clausal form

- Examples:

`{Nat(s(A)),¬Nat(A)}`
`{Nat(A)}`
gives
`{Nat(s(A))}`

`{Nat(s(A)),¬Nat(A)}`
`{Nat(x)}`
gives
`{Nat(s(A))}`

`{Nat(s(s(x))),¬Nat(s(x))}`
`{Nat(s(A))}`
gives
`{Nat(s(s(A)))}`

- We need to be able to work with variables !
- Unification of two expressions/literals

74

Resolution Principle
- Terms and instances

- Consider following atoms

```
P(x,f(y),B)

P(z,f(w),B) alphabetic variant
P(x,f(A),B) instance
P(g(z),f(A),B) instance
P(C,f(A),A) not an instance
```

- *Ground* expressions do not contain any variables

Resolution Principle
- Substitution

A substitution $s = \{t_1 / v_1, \dots, t_n / v_n\}$ substitutes variables v_i for terms t_i (t_i does NOT contain v_i)

Applying a substitution s to an expression ω yields the expression ωs which is ω with all occurrences of v_i replaced by t_i

```
P(x,f(y),B)

P(z,f(w),B)        s={z/x,w/y}
P(x,f(A),B)        s={A/y}
P(g(z),f(A),B)     s={g(z)/x,A/y}
P(C,f(A),A)        no substitution !
```

Workout

Calculate the substitutions for the resolution of the two clauses and the result clauses after the substitutions.

- ¬P(x), P(f(a))∨Q(f(y),g(a,b))
- ¬P(g(x,a)), P(y)∨Q(f(y),g(a,b))
- ¬P(g(x,f(a))), P(g(b,y))∨Q(f(y),g(a,b))
- ¬P(g(f(x),x)), P(g(y,f(f(y)))∨Q(f(y),g(a,b)))

77

Resolution Principle - Composing substitutions

- Composing substitutions s_1 and s_2 gives $s_1 s_2$ which is that substitution obtained by first applying s_2 to the terms in s_1 and adding remaining term/vars pairs to s_1

```
θ={g(x,y)/z}{A/x,B/y,C/w,D/z}=
{g(A,B)/z,A/x,B/y,C/w}
```

- Apply to

```
P(x,y,z)θ
gives
P(A,B,g(A,B))
```

78

ωs

Resolution Principle
- Properties of substitutions

$$(\omega s_1)s_2 = \omega(s_1 s_2)$$

$$(s_1 s_2)s_3 = s_1(s_2 s_3) \text{ associativity}$$

$$s_1 s_2 \neq s_2 s_1 \text{ not commutative}$$

Resolution Principle
- Unification

- Unifying a set of expressions $\{w_i\}$
  - Find substitution s such that $w_i s = w_j s$ for all i, j
  - Example `{P(x,f(y),B),P(x,f(B),B)}`
    `s={B/y,A/x}` not the simplest unifier
    `s={B/y}` most general unifier (mgu)
- The most general unifier, the mgu, g of $\{w_i\}$ has the property that if s is any unifier of $\{w_i\}$ then there exists a substitution s' such that $\{w_i\}s=\{w_i\}gs'$
- The common instance produced is unique up to alphabetic variants (variable renaming)
- *usually we assume there is no common variables in the two atoms*

Workout

P(B,f(x),g(A)) and P(y,z,f(w))

- construct an mgu
- construct a unifier that is not the most general.

81

Workout

Determine if each of the following sets is unifiable. If yes, construct an mgu.

- {Q(a), Q(b)}
- {Q(a,x),Q(a,a)}
- {Q(a,x,f(x)),Q(a,y,y)}
- {Q(x,y,z),Q(u,h(v,v),u)}
- $\{P(x_1,g(x_1),x_2,h(x_1,x_2),x_3,k(x_1,x_2,x_3)), P(y_1,y_2,e(y_2),y_3,f(y_2,y_3),y_4)\}$

82

Resolution Principle
- Disagreement set in unification

The disagreement set of a set of expressions $\{w_i\}$ is the set of subterms $\{ t_i \}$ of $\{w_i\}$ at the first position in $\{w_i\}$ for which the $\{w_i\}$ disagree

`{P(x,A,f(y)),P(w,B,z)}` gives `{x,w}`
`{P(x,A,f(y)),P(x,B,z)}` gives `{A,B}`
`{P(x,y,f(y)),P(x,B,z)}` gives `{y,B}`

Resolution Principle
- Unification algorithm

Unify($Terms$)
Initialize $k \leftarrow 0$;
Initialize $T_k = Terms$;
Initialize $\sigma_k = \{\}$;
* If T_k is a singleton, then output σ_k. Otherwise, continue.
Let D_k be the disagreement set of T_k
If there exists a var v_k and a term t_k in D_k such that v_k does not occur in t_k, continue. Otherwise, exit with failure.
$\sigma_{k+1} \leftarrow \sigma_k \{t_k / v_k\}$;
$T_{k+1} \leftarrow T_k \{t_k / v_k\}$;
$k \leftarrow k+1$;
Goto *

Predicate calculus Resolution

John Allan Robinson (1965)

Let C_1 and C_2 be two clauses with literals $l_1 \in C_1$ and $\neg l_2 \in C_2$ such that C_1 and C_2 do not contain common variables, and $mgu(l_1, l_2) = \theta$

then $C = [\{C_1 - \{l_1\}\} \cup \{C_2 - \{\neg l_2\}\}]\theta$

is a resolvent of C_1 and C_2

85

Predicate calculus Resolution

John Allan Robinson (1965)

Given

C: $l_1 \vee l_2 \vee \ldots \vee l_m$

C: $\neg k_1 \vee k_2 \vee \ldots \vee k_n$

$\theta = mgu(l_1, k_1)$

no common variables!

the *resolvent* is

$$l_2\theta \vee \ldots \vee l_m\theta \vee k_2\theta \vee \ldots \vee k_n\theta$$

86

Resolution Principle - Example

P(x)∨Q(f(x)) and R(g(x))∨¬Q(f(A))

Standardizing the variables apart

P(x)∨Q(f(x)) and R(g(y))∨¬Q(f(A))

Substitution θ={A/x}

Resolvent P(A)∨R(g(y))

Why can we do this ?

P(x)∨Q(x,y) and ¬P(A)∨¬R(B,z)

Standardizing the variables apart

Substitution θ={A/x}

Resolvent Q(A,y)∨¬R(B,z)

Why we think the variables in 2 clauses are irrelevant ?

Workout

Find all the possible resolvents (if any) of the following pairs of clauses.

- ¬P(x)∨Q(x,b),

Workout

Find all the possible resolvents (if any) of the following pairs of clauses.

- ¬P(x)∨Q(x,b), P(a)∨Q(a,b)
- ¬P(x)∨Q(x,x), ¬Q(a,f(a))
- ¬P(x,y,u)∨¬P(y,z,v)∨¬P(x,v,w)∨P(u,z,w), P(g(x,y),x,y)
- ¬P(v,z,v)∨P(w,z,w), P(w,h(x,x),w)

Resolution Principle - A stronger version of resolution

Use more than one literal per clause

{P(u),P(v)} and {¬P(x),¬P(y)}
do not resolve to empty clause.
However, ground instances
{P(A)} and {¬P(A)} resolve to empty clause

Resolution Principle
- Factors

Let C_1 be a clause such that there exists a substitution θ that is a mgu of a set of literals in C_1. Then $C_1\theta$ is a factor of C_1

Each clause is a factor of itself.

Also, {P(f(y)),R(f(y),y)}is a factor of {P(x),P(f(y)),R(x,y)} with $\theta = \{f(y)/x\}$

91

Resolution Principle
- Example of refutation

1.	{F(Art,Jon)}	Δ
2.	{F(Bob,Kim)}	Δ
3.	{¬F(x,y),P(x,y)}	Δ
4.	{¬P(Art,Jon)}	Γ
5.	{P(Art,Jon)}	1, 3
6.	{P(Bob,Kim)}	2, 3
7.	{¬F(Art,Jon)}	3, 4
8.	{}	4, 5
9.	{}	1, 7

92

Resolution Principle - Example

Hypothesies

∀x (dog(x) ⇒ animal(x))
dog(fido)
∀y (animal(y) ⇒ die(y))

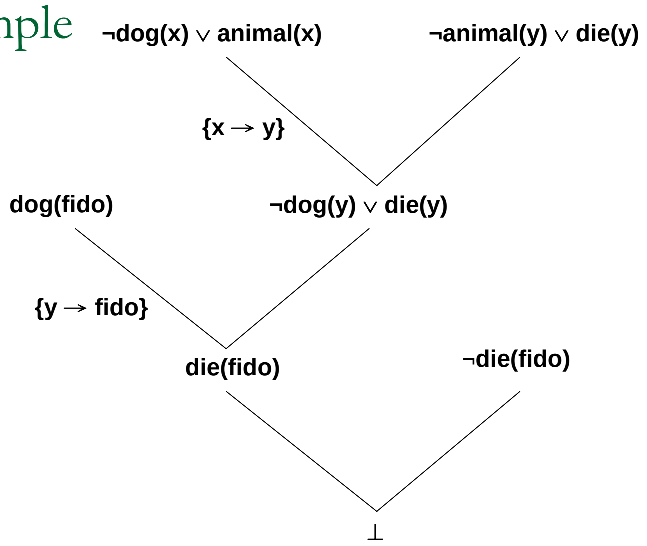
Conclusion
die(fido)

Clausal Form

¬dog(x) ∨ animal(x)
dog(fido)
¬animal(y) ∨ die(y)

Negate the goal
¬die(fido)

Resolution Principle - Example



Workout (resolution)
- Proof with resolution principle

Hypotheses:

- P(m(x),x) ∨ Q(m(x))
- ¬P(y,z) ∨ R(y)
- ¬Q(m(f(x,y))) ∨ ¬T(x,g(y))
- S(a) ∨ T(f(a),g(x))
- ¬R(m(y))
- ¬S(x) ∨ W(x,f(x,y))

Conclusion

W(a, y)

Resolution

Properties

- Resolution is sound
- Incomplete

  Given `P(A)`

  Infer `{P(A),P(B)}`

- But fortunately it is refutation complete
  - If KB is unsatisfiable then KB |- □

Resolution Principle
- Refutation Completeness

To decide whether a formula KB ⊨ w, do

- Convert KB to clausal form KB'
- Convert ¬w to clausal form ¬w'
- Combine ¬w' and KB' to give Δ
- Iteratively apply resolution to Δ and add the results back to Δ until either no more resolvents can be added, or until the empty clause is produced.

97

Resolution Principle
- Converting to clausal form (1/2)

To convert a formula KB into clausal form

1. Eliminate implication signs*

   $(p \Rightarrow q)$ becomes $(\neg p \vee q)$

2. Reduce scope of negation signs*

   $\neg(p \wedge q)$ becomes $(\neg p \vee \neg q)$

3. Standardize variables

   $(\forall x)[\neg P(x) \vee (\exists x) Q(x)]$ becomes $(\forall x)[\neg P(x) \vee (\exists y) Q(y)]$

4. Eliminate existential quantifiers using *Skolemization*

* Same as in prop. logic

98

Resolution Principle
- Converting to clausal form (2/2)

5. Convert to prenex form
   - Move all universal quantifiers to the front
6. Put the matrix in conjunctive normal form*
   - Use distribution rule
7. Eliminate universal quantifiers
8. Eliminate conjunction symbol *
9. Rename variables so that no variable occurs in more than one clause.

Resolution Principle
- Skolemization

Consider $(\forall x)[(\exists y)Height(x,y)]$

The y depends on the x

Define this dependence explicitly using a skolem function h(x)

Formula becomes $(\forall x)[Height(x,h(x))]$

General rule is that each occurrence of an existentially quantified variable is replaced by a *skolem* function whose arguments are those universally quantified variables whose scopes includes the scope of the existentially quantified one

Skolem functions do not yet occur elsewhere !

Resulting formula is *not* logically equivalent !

Resolution Principle
- Examples of Skolemization

[(∀w)Q(w)]⇒(∀x){(∀y){(∃z)[P(x,y,z)⇒(∀u)R(x,y,u,z)]}}

gives

[(∀w)Q(w)]⇒(∀x){(∀y)[P(x,y,g(x,y))⇒(∀u)R(x,y,u,g(x,y))]}

(∀x)[(∃y) F(x,y)] gives (∀x)F(x,h(x))

but

(∃y)[(∀x)F(x,y)] gives [(∀x)F(x,sk)] skolem constant

Not logically equivalent !

A well formed formula and its Skolem form are not logically equivalent.

However, a set of formulae is (un)satisfiable if and only if its skolem form is (un)satisfiable.

101

Resolution Principle
- Example of conversion to clausal form

```
initial: ∀x (∀y P(x,y)) ⇒ ¬(∀y Q(x,y) ⇒ R(x,y))
step 1: ∀x ¬(∀y P(x,y)) ∨ ¬(∀y ¬Q(x,y) ∨ R(x,y))

step 2: ∀x (∃y ¬P(x,y)) ∨ (∃y Q(x,y) ∧ ¬R(x,y))
step 3: ∀x (∃y ¬P(x,y)) ∨ (∃z Q(x,z) ∧ ¬R(x,z))
step 4: ∀x ¬P(x,F1(x)) ∨ (Q(x,F2(x)) ∧ ¬R(x,F2(x)))
step 5: ¬P(x,F1(x)) ∨ (Q(x,F2(x)) ∧ ¬R(x,F2(x)))
step 6: (¬P(x,F1(x)) ∨ Q(x,F2(x))) ∧
        (¬P(x,F1(x)) ∨ ¬R(x,F2(x)))
step 7: {¬P(x,F1(x)), Q(x,F2(x))}
        {¬P(x,F1(x)), ¬R(x,F2(x))}
step 8: {¬P(x1,F1(x1)), Q(x1,F2(x1))}
        {¬P(x2,F1(x2)), ¬R(x2,F2(x2))}
```

102

Workout

Convert the following formula to clausal form.

- ∃x(P(x)∧∀y ((∃z.Q(x,y,s(z)))→(Q(x,s(y),x)∧R(y))))
- ∀x∀y(S(x,y,z)→ ∃z(S(x,z) ∧ S(z,x)))

103

Resolution Principle
- Example of refutation by resolution

1.¬P(x)∨¬P(y)∨¬I(x,27)∨¬I(y,28)∨S(x,y)

all packages in room 27 are smaller than any of those in 28

2.P(A)

3.P(B)

4.I(A,27)∨I(A,28)

5.I(B,27)

6.¬S(B,A)

Prove I(A,27)

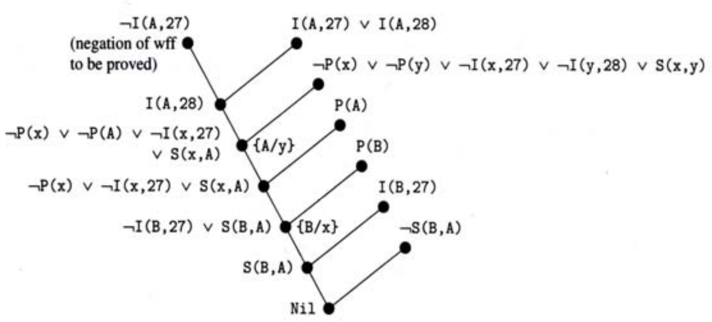


Figure 16.1

A Resolution Refutation

Resolution Principle
- Search Strategies

- Ordering strategies
  - In what order to perform resolution ?
  - Breadth-first, depth-first, iterative deepening ?
  - Unit-preference strategy :
    - Prefer those resolution steps in which at least one clause is a unit clause (containing a single literal)
- Refinement strategies
  - Unit resolution : allow only resolution with unit clauses

Resolution Principle
- Input Resolution

- at least one of the clauses being resolved is a member of the original set of clauses
- Input resolution is complete for Horn-clauses but incomplete in general
- E.g. $\{P,Q\},\{\neg P,Q\},\{P,\neg Q\},\{\neg P,\neg Q\}$
- One of the parents of the empty clause should belong to original set of clauses

Workout

Use input resolution to prove the theorem in page workout(resolution)!

Resolution Principle - Linear Resolution

- Linear resolvent is one in which at least one of the parents is either
  - an initial clause or
  - the resolvent of the previous resolution step.
- Refutation complete
- Many other resolution strategies exist

workout

Use linear resolution to prove the theorem in page workout(resolution)!

Resolution Principle - Set of support

- Ancestor : c2 is a descendant of c1 iff c2 is a resolvent of c1 (and another clause) or if c2 is a resolvent of a descendant of c1 (and another clause); c1 is an ancestor of c2
- Set of support : the set of clauses coming from the negation of the theorem (to be proven) and their descendants
- Set of support strategy : require that at least one of the clauses in each resolution step belongs to the set of support

workout

Use set of support to prove the theorem in page workout(resolution)!

Resolution Principle - Answer extraction

Suppose we wish to prove whether KB |= (∃w)f(w)

We are probably interested in knowing the w for which f(w) holds.

Add Ans(w) literal to each clause coming from the negation of the theorem to be proven; stop resolution process when there is a clause containing only Ans literal

Resolution Principle - Example of answer extraction

1.¬P(x)∨¬P(y)∨¬I(x,27)∨¬I(y,28)∨S(x,y)

all packages in room 27 are smaller than any of those in 28

2.P(A)

3.P(B)

4.I(A,27)∨I(A,28)

5.I(B,27)

6.¬S(B,A)

Prove (∃u)I(A,u), i.e. in which room is A?

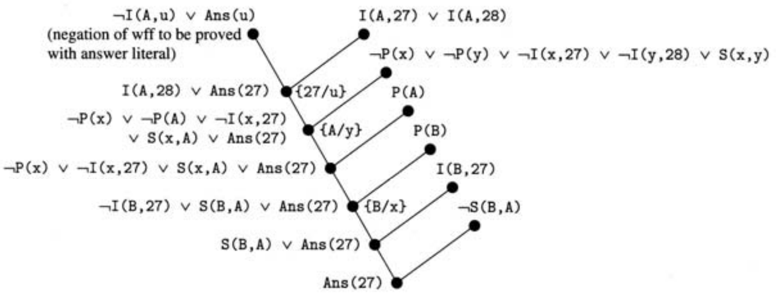


Figure 16.2

Answer Extraction

Workout

- Use answer extraction to prove the theorem in page workout(resolution)!

Theory of Equality

- Herbrand Theorem does not apply to FOL with equality.
- So far we've looked at predicate logic from the point of view of what is true in all interpretations.
  - This is very open-ended.
- Sometimes we want to assume at least something about our interpretation to enrich the theory in what we can express and prove.
- The meaning of equality is something that is common to all interpretations.
  - Its interpretation is that of equivalence in the domain.
  - If we add = as a predicate with special meaning in predicate logic, we can also add rules to our various proof procedures.
- Normal models are models in which the symbol = is interpreted as designating the equality relation.

115

Theory of Equality - An Axiomatic System with Equality

To the previous axioms and rules of inference, we add:

$$\text{EAx1} \quad \forall x. x = x$$

$$\text{EAx2} \quad \forall x. \forall y. x = y \Rightarrow (A(x,x) \Rightarrow A(x,y))$$

$$\text{EAx3} \quad \forall x. \forall y. x = y \Rightarrow f(x) = f(y)$$

116

Theory of Equality
- Natural Deduction Rules for Equality

Reflexivity

$$\frac{}{t = t} = \mathrm{i}$$

This inference rule is called an axiom, because it has no premises.

Theory of Equality
- Natural Deduction Rules for Equality

Substitution

$$\frac{t_1 = t_2 \quad P[t_2/x]}{P[t_1/x]} = \mathrm{e} \qquad \frac{t_1 = t_2 \quad P[t_1/x]}{P[t_2/x]} = \mathrm{e}$$

where t_1 and t_2 are free in x in P.

Theory of Equality - Substitution

- Recall: Given a variable x, a term t and a formula P, we define $P[t/x]$ to be the formula obtained by replacing **ALL free** occurrence of variable x in P with t.
- But with equality, we sometimes don't want to substitute for all occurrences of a variable.
- When we write $P[t/x]$ above the line, we get to choose what P is and therefore can choose the occurrences of a term that we wish to substitute for.

119

Theory of Equality - Substitution

Recall from existential introduction:

$$\frac{P[t/x]}{\exists x.\, P}\exists\text{i}$$

1	$\forall x.\, Q(x,x)$	premise
2	x_0	
3	$Q(x_0,x_0)$	$\forall$e 1
4	$\exists y.\, Q(x_0,y)$	$\exists$i 3
5	$\forall x.\, \exists y.\, Q(x,y)$	$\forall$i 5

- Matching the top of our rule, $P = Q(x_0, x)$, so line 3 of the proof is $P[x_0/x]$, which is $Q(x_0, x)$
- So we don't have to substitute in for every occurrence of a term.

120

Theory of Equality
- Examples

From these two inference rules, we can derive two other properties that we expect equality to have:

- Symmetry : $\vdash_{ND} \forall x, y.(x = y) \Rightarrow (y = x)$
- Transitivity : $\vdash_{ND} \forall x, y, z.(x = y) \land (y = z) \Rightarrow (x = z)$

121

Theory of Equality
- Example

$\vdash_{ND} \forall x, y.\ (x = y) \Rightarrow (y = x)$

1	x_0		
2	y_0		
3	$x_0 = y_0$	assumption	
4	$x_0 = x_0$	$=$ i	
5	$y_0 = x_0$	$=$ e 3, 4	
6	$x_0 = y_0 \Rightarrow y_0 = x_0$	$\Rightarrow$ i 3 – 5	
7	$\forall y.\ (x_0 = y) \Rightarrow (y = x_0)$	$\forall$i 2 – 6	
8	$\forall x, y.\ (x = y) \Rightarrow (y = x)$	$\forall$i 1 – 7	

122

Theory of Equality
- Example

$\vdash_{ND} \forall x, y, z.\ (x = y) \wedge (y = z) \Rightarrow (x = z)$

1	x_0	
2	y_0	
3	$(x_0 = y_0) \wedge (y_0 = z_0)$	assumption
4	$x_0 = y_0$	$\wedge e\ 3$
5	$y_0 = z_0$	$\wedge e\ 3$
6	$x_0 = z_0$	$= e\ 4, 5$
7	$(x_0 = y_0) \wedge (y_0 = z_0) \Rightarrow (x_0 = z_0)$	$\Rightarrow i\ 3-6$
8	$\forall y.\ (x_0 = y) \wedge (y = z_0) \Rightarrow (x_0 = z_0)$	$\forall i\ 2-7$
9	$\forall x, y.\ (x = y) \wedge (y = z) \Rightarrow (x = z)$	$\forall i\ 1-8$

Theory of Equality
- Leibniz's Law

- The substitution inference rule is related to Leibniz's Law.
- Leibniz's Law:

  if $t_1 = t_2$ is a theorem, then so is $P[t_1 / x] \Leftrightarrow P[t_2 / x]$
- Leibniz's Law is generally referred to as the ability to substitute "equals for equals".

Leibniz

Gottfried Wilhelm von Leibniz (1646-1716)

- The founder of differential and integral calculus.
- Another of Leibniz's lifelong aims was to collate all human knowledge.

"[He was] one of the last great polymaths – not in the frivolous sense of having a wide general knowledge, but in the deeper sense of one who is a citizen of the whole world of intellectual inquiry."

125

Theory of Equality - Example

- From our natural deduction rules, we can derive Leibniz's Law:

$$t_1 = t_2 \vdash_{ND} P(t_1) \Leftrightarrow P(t_2)$$

126

Theory of Equality
- Equality: Semantics

- The semantics of the equality symbol is equality on the objects of the domain.
- In ALL interpretations it means the same thing.
- Normal interpretations are interpretations in which the symbol = is interpreted as designating the equality relation on the domain.
- We will restrict ourselves to normal interpretations from now on.

127

Theory of Equality
- Extensional Equality

- Equality in the domain is extensional, meaning it is equality in meaning rather than form.
- This is in contrast to intensional equality which is equality in form rather than meaning.
- In logic, we are interested in whether two terms represent the same object, not whether they are the same symbols.
- If two terms are intensionally equal then they are also extensionally equal, but not necessarily the other way around.

128

Theory of Equality
- Equality: Counterexamples

- Show the following argument is not valid:
  $\exists x.P(x) \wedge Q(x), P(A), A = B \models Q(B)$
- where A,B are constants

Theory of Arithmetic

- Another commonly used theory is that of arithmetic.
- It was formalized by Dedekind in 1879 and also by Peano in 1889.
- It is generally referred to as Peano's Axioms.
- The model of the system is the natural numbers with the constants 0 and 1, the functions +, *, and the relation <.

Peano's Axioms

P1:	$\forall x.\forall y.(x + y = y + x)$	Commutativity of +
P2:	$\forall x.\forall y.(x * y = y * x)$	Commutativity of *
P3:	$\forall x.\forall y.\forall z.x + (y + z) = (x + y) + z$	Associativity of +
P4:	$\forall x.\forall y.\forall z.x * (y * z) = (x * y) * z$	Associativity of *
P5:	$\forall x.\forall y.\forall z.x * (y + z) = (x * y) + (x * z)$	Distributivity
P6:	$\forall x.x + 0 = x$	Property of 0
P7:	$\forall x.x * 1 = x$	Property of 1
P8:	$\forall x.\neg(x + 1 = 0)$	0 is not a successor
P9:	$\forall x.\forall y.x + 1 = y + 1 \Rightarrow x = y$	
P10:	$\forall x.\forall y.x < y \Leftrightarrow \exists z.\neg(z = 0) \wedge y = x + z$	Property of $<$
P11:	$P[0/x] \wedge (\forall y.P[y/x] \Rightarrow P[y + 1/x]) \Rightarrow \forall x.P$	Induction Scheme

Intuitionistic Logic

- "A proof that something exists is constructive if it provides a method for actually constructing it."
- In intuitionistic logic, only constructive proofs are allowed.
- Therefore, they disallow proofs by contradiction. To show ϕ, you can't just show $\neg\phi$ is impossible.
- They also disallow the law of the excluded middle arguing that you have to actually show one of ϕ or $\neg\phi$ before you can conclude $\phi \vee \neg\phi$
- Intuitionistic logic was invented by Brouwer. Theorem provers that use intuitionistic logic are Nuprl, Coq, Elf, and Lego.
- In this course, we will only be studying classical logic.

Summary

- Predicate Logic (motivation, syntax and terminology, semantics, axiom systems, natural deduction)
- Equality, Arithmetic
- Mechanical theorem proving