Neuro-Symbolic Artificial Intelligence
Chapter 4
Predicate Logic

Nils Holzenberger

March 12, 2024

Outline

1. Propositional logic (recap)
   - Tautologies
   - Proof by resolution
2. Predicate logic: introduction
   - From language to logic
   - Quantification
   - Propositional *vs* predicate logic
3. Predicate logic: definition
   - Syntax and semantics
   - Valid formulas
4. Proof by resolution for predicate logic
   - Examples
   - Normalized forms
   - Proof procedure

Outline

Syntax *vs* Semantics

Last lecture:

- We defined a language of symbols and how to manipulate them
  - connectives $\neg$, $\vee$, $\wedge$, $\uparrow$...
  - atomic formulas A, B, C, X, Y, Z
- We defined how to map propositional formulas to True, False
  - connectives Not, And, Or, Nand...
- The link between syntax and semantics is the valuation function
  - correspondence between syntactic and semantic connectives

Syntactic connective	Semantic connective
$\neg$	Not
$\wedge$	And
$\vee$	Or
$\supset$	$\Rightarrow$
...	...

Natural Logic

r	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\between$	$\smile$	$\curlywedge$	$\#$
$\rho(r)$	$\equiv$	$\sqsupseteq$	$\sqsubseteq$	$\smile$	$\between$	$\curlywedge$	$\#$

Table 1: The projection function ρ, shown for downward polarity contexts only. The input r is the lexical relation between two words in a sentence; the projected relation $\rho(r)$ is the relation between the two sentences differing only by that word. Note that ρ is the identity function in upward polarity contexts.

$\bowtie$	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\curlywedge$	$\between$	$\smile$	$\#$
$\equiv$	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\curlywedge$	$\between$	$\smile$	$\#$
$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	$\#$	$\between$	$\between$	$\#$	$\#$
$\sqsupseteq$	$\sqsupseteq$	$\#$	$\sqsupseteq$	$\smile$	$\#$	$\smile$	$\#$
$\curlywedge$	$\curlywedge$	$\smile$	$\between$	$\equiv$	$\sqsupseteq$	$\sqsubseteq$	$\#$
$\between$	$\between$	$\#$	$\between$	$\sqsubseteq$	$\#$	$\sqsubseteq$	$\#$
$\smile$	$\smile$	$\smile$	$\#$	$\sqsupseteq$	$\sqsupseteq$	$\#$	$\#$
$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$

Table 2: The join table as shown in Icard (2012). Entries in the table are the result of joining a row with a column.

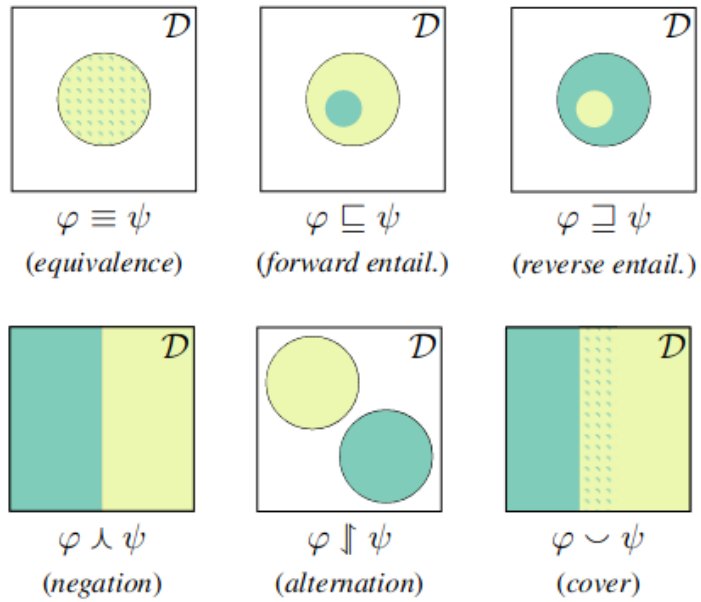

Figure 2: The model-theoretic interpretation of the MacCartney relations. The figure shows the relation between the denotation of φ (dark) and ψ (light). The universe is denoted by $\mathcal{D}$.

From Angeli and Manning, *NaturalLI: Natural Logic Inference for Common Sense Reasoning*, EMNLP 2014

$\vdash$ *vs* $\models$

- $S \models X$
  - S is a set of formulas
  - X is a formula
  - For any valuation v, if v assigns True to all elements in S, then v assigns True to X
  - This is equivalent to $S \cup \{\neg X\}$ not being satisfiable
- $S \vdash X$
  - S is a set of formulas
  - X is a formula
  - There exists a syntactic proof (e.g. proof by resolution) that derives X from S
  - In the case of proof by resolution, it shows that $S \cup \{\neg X\}$ is not satisfiable
- An axiomatic defines $\vdash$ and $\models$
  - A sound axiomatic has the property "if $S \vdash X$ then $S \models X$"
  - A complete axiomatic has the property "if $S \models X$ then $S \vdash X$"

$S \models X$ iff $S \cup \{\neg X\}$ is not satisfiable.

Let's show that $S \models X$ iff $S \cup \{\neg X\}$ is not satisfiable.

This is an important result, used in theorem proving. It turns the problem of proving an implication into proving that some formula is a tautology.

$\boxed{\Rightarrow}$ Let v be a valuation.

- If there is a formula $A \in S$ such that $v(A) = \text{False}$, then v does not satisfy S and hence does not satisfy $S \cup \{\neg X\}$.
- If for all $A \in S$, $v(A) = \text{True}$, then by $S \models X$ we have that $v(X) = \text{True}$, and so $v(\neg X) = \text{Not } v(X) = \textit{False}$. So v does not satisfy X and does not satisfy $S \cup \{\neg X\}$.

$S \models X$ iff $S \cup \{\neg X\}$ is not satisfiable.

Let's show that $S \models X$ iff $S \cup \{\neg X\}$ is not satisfiable.

This is an important result, used in theorem proving. It turns the problem of proving an implication into proving that some formula is a tautology.

$\boxed{\Leftarrow}$ Let v be a valuation that satisfies S, i.e. $\forall A \in S$, $v(A) = \text{True}$.
$S \cup \{\neg X\}$ is not satisfiable, so $v(\neg X) = \text{False}$, so $\text{Not}\, v(X) = \text{False}$ and $v(X) = \text{True}$.

Goal

We want to prove $S \models X$. This is the same as proving that the set $S \cup \{\neg X\}$ is not satisfiable.

Let $S \cup \{\neg X\} = \{A_1, A_2, ..., A_n\}$

We want to show that $\{A_1, A_2, ..., A_n\}$ is not satisfiable i.e. that regardless of the valuation that I pick, one of the A_i will evaluate to False.

Procedure

- A *sequence* is a conjunction of *lines*
- Each *line* is a generalized disjunction (a clause)
- A proof of X by resolution is a sequence starting with the $[\neg X]$ line (goal) and ending with an empty clause []
- *Growth* of the sequence:
  - if a clause reads as $[...(\beta_1 \vee \beta_2)...]$, insert a new line: $[...\beta_1, \beta_2...]$
  - if a clause reads as $[...(\alpha_1 \wedge \alpha_2)...]$, insert two new lines: $[...\alpha_1...]$ and $[...\alpha_2...]$
  - when adding new lines, replace $\neg\neg X$ by X, $\neg \mathrm{T}$ by $\bot$ and $\neg \bot$ by T
- *Resolution*: from lines $[A, X, B]$ and $[C, \neg X, D]$ create the line $[A, B, C, D]$, i.e. concatenate the lines leaving aside all occurrences of X and of $\neg X$

Note that in practice, we are trying to prove $S \vdash X$, so we start with $[S_1], [S_2], ..., [S_n], [\neg X]$ in the sequence.

Forward chaining *vs* backward chaining

- *Backward chaining*
  - Start from goal, try to prove it using assumptions
  - Proof by resolution (1965) by John Alan Robinson
- *Forward chaining*
  - Start from assumptions, combine them to prove new theorems
  - Logic theorist (1956) by Allen Newell
- Can backward chaining solve any problem?
  - The number of statements in the proof by resolution can explode combinatorially
  - It's not always clear how to translate a problem into logic, eg natural language problems

Outline

From language to logic

How would you encode this into logic?

- Socrates is mortal
- All human beings are mortal
- Only adults may drink alcohol

Gottlob Frege

4 is preceded by 3 and 3 precedes 4 are equivalent

But Every instant is preceded by an instant and
An instant precedes every instant are not equivalent

This was solved by Gottlob Frege around 1878 using quantifiers

Negation

If there is an interpreter, then any two individuals communicate

$$(\forall x)(\forall y)(\forall l_1)(\forall l_2)((p(x,l_1) \wedge (p(y,l_2) \wedge ((\exists z)p(z,l_1) \wedge p(z,l_2)))) \supset c(x,y))$$

There are two individuals who cannot communicate, despite the existence of an interpreter

$$(\exists x)(\exists y)(\exists l_1)(\exists l_2)((p(x,l_1) \wedge (p(y,l_2) \wedge ((\exists z)p(z,l_1) \wedge p(z,l_2)))) \wedge \neg c(x,y))$$

What is new?

Predicate logic = propositional logic with variables

Propositional *vs* predicate logic

Propositional logic
Zeroth-order logic
Atomic formulas
Formulas

Predicate logic
First-order logic
Atomic formulas
Formulas

Outline

Syntax

- Terms
  - Variables $x_1, x_2, ...$
  - Constants $c_1, c_2, ...$
  - Functors $f(t_1, .., t_n)$ where f is a functor of arity n and each t_i is a term
  - The sets $\{x_i\}_i$, $\{c_i\}_i$, $\{p_i\}_i$, $\{f_i\}_i$ are given once and form the language signature
- Atomic formula
  - $p(t_1, ..., t_n)$
  - p is a predicate of arity n and each t_i is a term
- Formula
  - Atomic formulas
  - $\neg F$ where F is a formula
  - $(F \circ G)$ where F and G are formulas and $\circ$ a binary connective
  - $(\forall x)F$ where x is a variable, F is a formula
  - $(\exists x)F$ where x is a variable, F is a formula

Syntax: free and closed

- A term is *closed* iff it contains no variable
- A variable v is *free* in an atomic formula F iff v has an occurrence in F
- Let $\diamond$ be a quantifier. A variable v is *free* in $(\diamond v')F$ iff v is free in F and v is different from v'
- A variable v is *free* in $\neg F$ iff v is free in F
- Let $\circ$ be a 2-place connective. A variable v is *free* in $F_1 \circ F_2$ iff v is free in F_1 or in F_2
- A variable v is *bound* in a formula F iff it has no free occurrence
- A formula is *closed* iff it contains no free variable.

Propositional *vs* predicate logic

Propositional logic
Zeroth-order logic
Atomic formulas
Formulas

Predicate logic
First-order logic
Atomic formulas
Formulas
Terms: Variables, Constants, Functors
Quantifiers

Models

- We have defined a syntax for propositional logic
- How do we interpret formulas? How do we compute if they are True or False?
- These questions are answered by *semantics*
- ⚠ *Semantics* means slightly different things between predicate logic and propositional logic.
  - In propositional logic (last class) semantics was the truth value of a formula, as defined by a valuation
  - In predicate logic (this class) semantics is the *model*, which gives meaning to the symbols. The truth value of a formula is called an *attitude*.

Semantics

- We are given a non-empty set D called the *domain*
- An *interpretation* I associates
  - each constant c of the language with an element c^I of D (c^I is the *interpretation* of c)
  - each functor f of arity n to a function $f^I : D^n \to D$
  - each predicate p of arity n to an n-ary relation p^I in D
- An *assignment* A instantiates each variable v by giving it a value v^A taken from D
- Terms are interpreted recursively from the interpretation of their elements. For each term t, its interpretation $t^{I,A}$ is defined as:
  - c^I for a constant c,
  - v^A for a variable v,
  - $f^I(t_1^{I,A}, t_2^{I,A}, \ldots t_n^{I,A})$ for a functional term $f(t_1, , \ldots, t_n)$
- A *model* $M(D, I)$ is defined by the domain D and the interpretation I

Semantics: example

- Domain D is the set of first-year students at Télécom Paris with their birthdates and the classes they are taking this academic year.
  $D = S \cup B \cup C$
  - $S = \{\text{Chahinez}, \text{Souhail}, \text{Judith}...\}$
  - $B = \{\text{2006-02-21}, \text{2006-06-04}, \text{2007-12-12}...\}$
  - $C = \{\text{INF104}, \text{IA101}, \text{COM103}...\}$
- Interpretation I associates:
  - constants with elements of D: $c_{53} = \text{Souhail}$, $c_{12} = \text{2006-06-04}$
  - functors with functions $D^n \rightarrow D$: mapping from student to birthdate
  - predicates to a n-ary relations, like entries in a database:
    - whether student is signed up for IA101
    - whether two students are in the same class
- Assignment A instantiates variables with values from D: $x_{97} = \text{INF104}$, $x_{19} = \text{COM103}$.

Semantics: example #2

Find models for

$(\forall x)p(a,x)$

Attitudes (aka truth values)

- Predicates (atomic formulas):
  - $p(t_1, t_2, ..., t_n)^{I,A}$ = True if and only if $(t_1^{I,A}, t_2^{I,A}, ... t_n^{I,A}) \in p^I$
  - $\rightarrow$ *is* $(t_1^{I,A}, t_2^{I,A}, ... t_n^{I,A})$ *in table* p*?*
- Consistency (see valuations in propositional logic)
  - $\mathrm{T}^{I,A}$ = True
  - $\bot^{I,A}$ = False
  - $(\neg X^{I,A})$ = Not $X^{I,A}$
  - $(X \bullet Y)^{I,A} = X^{I,A} \blacksquare Y^{I,A}$ for coupled operators $\bullet$ and $\blacksquare$
- Quantifiers
  - $((\forall x)\ F)^{I,A}$ = True if and only if $F^{I,B}$ = True for all assignments B equal to A save for x
  - $((\exists x)\ F)^{I,A}$ = True if and only if $F^{I,B}$ = True for (at least) one assignment B equal to A save for x

Propositional *vs* predicate logic

Propositional logic
Zeroth-order logic
Atomic formulas
Formulas
Valuation

Predicate logic
First-order logic
Atomic formulas
Formulas
Terms: Variables, Constants, Functors
Quantifiers
Model: Interpretation + Domain
Assignments
Attitudes

Valid formulas (= tautologies in predicate logic)

- Validity
  - A formula F is true in model $M(D, I)$ if $F^{I,A} = \text{True}$ for all assignments A
  - A formula F is valid if F is true in any model
- Satisfiability
  - A set S of formulas is satisfiable in model $M(D, I)$ if there is (at least) an assignment A such that $F^{I,A} = \text{True}$ for all $F \in S$
  - S is satisfiable if S is satisfiable in at least one model.
- Consequence: a formula F is valid iff $\{\neg F\}$ is not satisfiable.

Examples

- $((\forall x)p(a,x)) \supset p(a,a)$ — valid
- $(\forall x)(p(a,x) \supset p(a,a))$ — not valid, satisfiable
- $(\forall x)(\forall y)(p(x,y) \supset p(y,x))$ — not valid, satisfiable
- $(\forall x)(\exists y)p(x,y)$ — not valid, satisfiable
- $(\exists x)(\exists y)(p(x) \wedge \neg p(y))$ — not valid, satisfiable
- $(\forall x(p(x) \wedge q(x))) \equiv (\forall x\, p(x) \wedge \forall x\, q(x))$ — valid
- $(\exists x(\neg p(x) \wedge p(x)))$ — not satisfiable

Propositional *vs* predicate logic

Propositional logic
Zeroth-order logic
Atomic formulas
Formulas
Valuation
Tautologies
Satisfiability

Predicate logic
First-order logic
Atomic formulas
Formulas
Terms: Variables, Constants, Functors
Quantifiers
Model: Interpretation + Domain
Assignments
Attitudes
Valid formulas
Satisfiability

Outline

Summary so far

- We have defined syntax (constants, variables, predicates, formulas, quantifiers...)
- We have defined semantics (interpretation, domain, assignment, attitude...)
- We have defined interesting types of formulas (valid formulas)
- Q: How do we use find valid formulas without semantics?
- A: Proof by resolution (again)

Example

Prove $((\exists x)(A \wedge \neg P(x)) \supset \neg(\forall x)P(x))$ by resolution:

$\neg((\exists x)(A \wedge \neg P(x)) \supset \neg(\forall x)P(x))$

---

1. $(\exists x)(A \wedge \neg P(x))$
2. $(\forall x)P(x)$

---

1 →3: skolemization

3. $(A \wedge \neg P(c))$
4. $(\forall x)P(x)$

---

5. A
6. $\neg P(c)$
7. $(\forall x)P(x)$

7 →8: dropping universal quantifiers

5. A
6. $\neg P(c)$
8. $P(x)$

---

6 and 8 →9 and 10: unification

5. A
9. $x = c$
10. []

Success!

→the formula is valid

Example #2

Proof by resolution of $((\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x)))$

1. $[\neg((\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x)))]$

---

renaming variables

2. $[\neg((\forall x)(P(x) \vee Q(x)) \supset ((\exists y)P(y) \vee (\forall z)Q(z)))]$

---

rewriting $\neg(A \supset B)$ as a conjunction

3. $[(\forall x)(P(x) \vee Q(x))]$
4. $[\neg((\exists y)P(y) \vee (\forall z)Q(z))]$

---

3 →5: dropping universal quantifier

5. $[P(x) \vee Q(x)]$

4 →6 and 7: rewriting a conjunction

6. $[\neg((\exists y)P(y))]$
7. $[\neg((\forall z)Q(z))]$

---

5 →8: rewriting a disjunction

8. $[P(x), Q(x)]$

4 →9 and 10: rewriting quantifiers

9. $[(\forall y)\neg P(y)]$
10. $[(\exists z)\neg Q(z)]$

8. $[P(x), Q(x)]$

9 →11 : dropping universal quantifier

11. $[\neg P(y)]$

10 →12 : skolemization

12. $[\neg Q(c)]$

---

12. $[\neg Q(c)]$

8 and 11 →13 and 14 : unification

13. $[Q(y)]$
14. $x = y$

---

12 and 13 →15 and 16 : unification

14. $x = y$
15. $y = c$
16. []

Success!

→the formula is valid

Tools for proof by resolution

- Prenex form
- Multiple quantifiers
- Skolemization

Prenex form

In Prenex form, quantifiers are outside the formulas

⚠ The following equivalences may only be used after variables have been renamed

$$\neg\,(\exists v)\;A \equiv (\forall v)\;\neg\,A$$
$$\neg\,(\forall v)\;A \equiv (\exists v)\;\neg\,A$$
$$((\forall v)\;A\;\wedge\;B) \equiv (\forall v)\;(A\;\wedge\;B)$$
$$(A\;\wedge\;(\forall v)\;B) \equiv (\forall v)\;(A\;\wedge\;B)$$
$$((\exists v)\;A\;\wedge\;B) \equiv (\exists v)\;(A\;\wedge\;B)$$

$$(A\;\wedge\;(\exists v)\;B) \equiv (\exists v)\;(A\;\wedge\;B)$$
$$((\forall v)\;A\;\supset\;B) \equiv (\exists v)\;(A\;\supset\;B)$$
$$(A\;\supset\;(\forall v)\;B) \equiv (\forall v)\;(A\;\supset\;B)$$
$$((\exists v)\;A\;\supset\;B) \equiv (\forall v)\;(A\;\supset\;B)$$
$$(A\;\supset\;(\exists v)\;B) \equiv (\exists v)\;(A\;\supset\;B)$$

Multiple quantifiers

Put the following in prenex form: $((\exists x)(\forall y)R(x,y) \supset (\forall y)(\exists x)R(x,y))$

$((\forall v)\ A \supset B) \equiv (\exists v)\ (A \supset B)$ $\quad$ $((\exists v)\ A \supset B) \equiv (\forall v)\ (A \supset B)$

$(A \supset (\forall v)\ B) \equiv (\forall v)\ (A \supset B)$ $\quad$ $(A \supset (\exists v)\ B) \equiv (\exists v)\ (A \supset B)$

$((\exists x)(\forall y)R(x,y) \supset (\forall u)(\exists v)R(v,u))$

$(\forall x)(\exists y)(\forall u)(\exists v)(R(x,y) \supset R(v,u))$

$(\forall u)(\forall x)(\exists v)(\exists y)(R(x,y) \supset R(v,u))$

Multiple orderings possible. Some better than others? See skolemization.

Skolemization: definition

A formula in prenex form:
$(Q_1 x_1)(Q_2 x_2)...(Q_{k-1} x_{k-1})(\exists x_k)(Q_{k+1} x_{k+1})...(Q_n x_n)F$

Is skolemized as $(Q_1 x_1)(Q_2 x_2)...(Q_n x_n)F(x_k/f(x_1, x_2, ...x_{k-1})$

f is a new functor that does not belong to the language. f is a *Skolem function*.

Both formulas have the same truth conditions.

Note that f is of arity $k-1$

The use of Skolem functions ensures that any formula F with no free variables (i.e. without unquantified variables) can be mapped to a formula G in prenex form with only universal quantifiers such that F is satisfiable if and only if G is satisfiable.

Skolemization: example

Everyone who is married has a spouse.

$$(\forall x)(\text{married}(x) \supset (\exists y)\text{spouse}(x, y))$$

Move quantifiers:

$$(\forall x)(\exists y)(\text{married}(x) \supset \text{spouse}(x, y))$$

Skolemize:

$$(\forall x)(\text{married}(x) \supset \text{spouse}(x, f(x)))$$

f maps an individual to his/her spouse (if that spouse exists)

Skolemization: example #2

From previous example, skolemize

$$(\forall x)(\exists y)(\forall u)(\exists v)(R(x,y) \supset R(v,u))$$

$$(\forall x)(\cancel{\exists y})(\forall u)(\exists v)(R(x,f(x)) \supset R(v,u))$$

From previous example, skolemize the equivalent

$$(\forall u)(\forall x)(\exists v)(\exists y)(R(x,y) \supset R(v,u))$$

$$(\forall u)(\forall x)(\cancel{\exists v})(\exists y)(R(x,y) \supset R(f(u,x),u))$$

→ Ideally, put existential quantifiers at the beginning.

Proof by resolution

To prove the validity of formula F:

1. Transform $\neg F$ in prenex form, renaming variables if necessary

2. Remove existential quantifiers through skolemization

3. Remove universal quantifiers

(this amounts to transforming the associated quantified variables into free variables)

4. Transform the formula into conjunctive normal form (optional)

5. Use the resolution algorithm with unification.

6. If the empty clause is obtained through this process starting from $\neg F$, then $\{\neg F\}$ is not satisfiable, and F is valid. Otherwise F is not valid.

Propositional *vs* predicate logic

Propositional logic	**Predicate logic**	*Unification*
Zeroth-order logic	First-order logic	*Dropping universal quantifiers*
Atomic formulas	Atomic formulas	
Formulas	Formulas	
Valuation	Terms: Variables, Constants, Functors	
Tautologies	Quantifiers	
Satisfiability	Model: Interpretation + Domain	
Proof by resolution	Assignments	
	Attitudes	
	Valid formulas	
	Satisfiability	
	Proof by resolution	
	Skolemization	

Soundness, completeness, decidability

Soundness: Anything I can prove using syntax (← eg with resolution), I can prove using models and interpretations.

Completeness: Anything I can prove using models and interpretations, I can prove using syntax.

NEW Decidability: I can decide in finite time whether a formula is valid.

Kurt Gödel

1929, PhD thesis:

There are sound and complete proof methods for predicate logic

Examples: Axiomatic systems, Tableaux, Resolution, Natural deduction

1931, *On Formally Undecidable Propositions of Principia Mathematica and Related Systems I*:

There are no complete methods for the axioms of arithmetics

The problem comes from recurrence.

Decidability

Predicate logic is sound and complete for proof by resolution (good news)

Predicate logic is semi-decidable (bad news):

- Let's say we have a formula in predicate logic. We know there is a proof for it (because of completeness).
- If the formula is valid, the proof can be found in finite time. But if the formula is not valid, there is no guarantee to find out in finite time.

Propositional *vs* predicate logic

Propositional logic	**Predicate logic**	Unification
Zeroth-order logic	First-order logic	Dropping universal quantifiers
Atomic formulas	Atomic formulas	*Sound*
Formulas	Formulas	*Complete*
Valuation	Terms: Variables, Constants, Functors	*Semi-decidable*
Tautologies	Quantifiers	
Satisfiability	Model: Interpretation + Domain	
Proof by resolution	Assignments	
Sound	Attitudes	
Complete	Valid formulas	
Decidable (SAT)	Satisfiability	
	Proof by resolution	
	Skolemization	