CS440/ECE448: Intro to Artificial Intelligence

Lecture 8: First-order predicate logic

Prof. Julia Hockenmaier
juliahmr@illinois.edu

http://cs.illinois.edu/fa11/cs440

Back to resolution in propositional logic

The resolution rule

Unit resolution:

$$\frac{p_1 \vee \ldots \vee p_{i-1} \vee \mathbf{p_i} \vee p_{i+1} \vee \ldots \vee p_n \qquad \neg \mathbf{p_i}}{p_1 \vee \ldots \vee p_{i-1} \vee p_{i+1} \vee \ldots \vee p_n}$$

Full resolution:

$$\frac{\mathbf{p_1} \vee \ldots \vee \mathbf{p_i} \vee \ldots \vee \mathbf{p_n} \qquad \mathbf{q_1} \vee \ldots \vee \neg \mathbf{p_i} \vee \ldots \vee \mathbf{q_m}}{\mathbf{p_1} \vee \ldots \vee \mathbf{p_n} \vee \mathbf{q_1} \vee \ldots \vee \mathbf{q_m}}$$

Final step: factoring (remove any duplicate literals from the result $A \vee A \equiv A$)

Proof by contradiction

How do we prove that $\alpha \models \beta$?

α entails β ('$\alpha \models \beta$') *iff* $\alpha \wedge \neg \beta$ not satisfiable.

Proof:

$\alpha \wedge \neg \beta$ not satisfiable *iff* $\models \neg (\alpha \wedge \neg \beta)$

Assume $\models \neg (\alpha \wedge \neg \beta)$.

$\models \neg \alpha \vee \beta)$

$\models \alpha \rightarrow \beta$.

Thus, $\neg (\alpha \wedge \neg \beta) \equiv \alpha \rightarrow \beta$.

A resolution algorithm

Goal: prove $\alpha \models \beta$' by showing that $\alpha \wedge \neg \beta$ is not satisfiable (*false*)

Observation:

Resolution derives a contradiction (*false*) if it derives the empty clause:

$$\frac{\mathbf{p_i} \qquad \neg \mathbf{p_i}}{\varnothing}$$

```
function PLresolution(α, β)
    input: formula α, // knowledge base
           formula β // query
    clauses := CNF(α ∧¬ β)
    new := {}
    while true:
        for each c1, c2 in clauses do
            resolvents := resolve(c1, c2)
            if ∅ in resolvents then return true;
            new := new ∪ resolvents
        if new ⊆ clauses then return false;
        clauses := clauses ∪ new
```

Completeness of Resolution

Resolution closure RC(S):
The set of all clauses that can be derived by resolution from a set of clauses S.
If S is finite, RC(S) is finite.

Ground resolution theorem:
If RC(S) contains ∅, S is not satisfiable.
If RC(S) does not contain ∅, S is satisfiable.

If RC(S) doesn't contain ∅…

…S is satisfiable, because we can build a model for its variables $p_1 \ldots p_n$:

For i from 1….n:
    if a clause in RC(S) contains $\neg p_i$
        and all its other literals are false,
    then assign false to p_i
    otherwise assign true to p_i

A little exercise…

Meet your neighbors!

Task 1: Give a WFF that is

1. Valid
2. Satisfiable
3. Not valid
4. Not satisfiable
5. Satisfiable but not valid
6. Valid but not satisfiable

Axiom set Δ:

1. $P \Rightarrow Q$
2. $L \Rightarrow R$
3. $Q \Rightarrow R$
4. $\neg L \Rightarrow Z$
5. $S \Rightarrow L$
6. $P \Rightarrow G$
7. $\neg L$
8. A
9. P
10. G

Task 2:

Is **R** true?

Can we prove it using Modus Ponens?

$$\frac{\Theta \Rightarrow \Psi \qquad \Theta}{\Psi}$$

Derivation of R

1.	$P \Rightarrow Q$	Δ
2.	$L \Rightarrow R$	Δ
3.	$Q \Rightarrow R$	Δ
4.	$\neg L \Rightarrow Z$	Δ
5.	$S \Rightarrow L$	Δ
6.	$P \Rightarrow G$	Δ
7.	$\neg L$	Δ
8.	A	Δ
9.	P	Δ
10.	G	Δ
11.	Q	MP: 1,9
12.	R	MP: 3,11

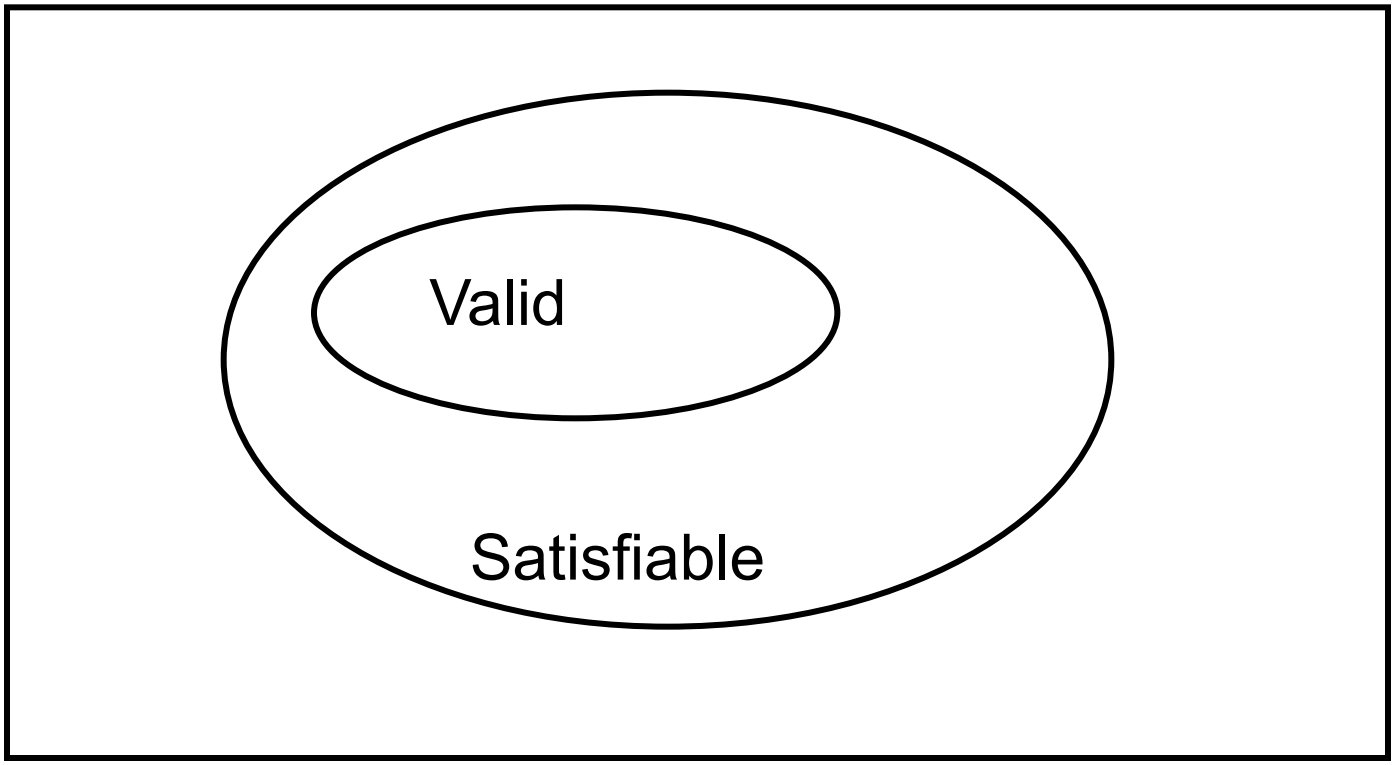
Give a WFF that is

1. Valid — $P \vee \neg P \quad R \Rightarrow R$
2. Satisfiable — P
3. Not valid — P
4. Not satisfiable — $P \wedge \neg P$
5. Satisfiable but not valid — P
6. Valid but not satisfiable — Not possible

Derivation of R

1.	$P \Rightarrow Q$	Δ
2.	$L \Rightarrow R$	Δ
3.	$Q \Rightarrow R$	Δ
4.	$\neg L \Rightarrow Z$	Δ
5.	$S \Rightarrow L$	Δ
6.	$P \Rightarrow G$	Δ
7.	$\neg L$	Δ
8.	A	Δ
9.	P	Δ
10.	G	Δ
11.	Q	MP: 1,9
12.	R	MP: 3,11

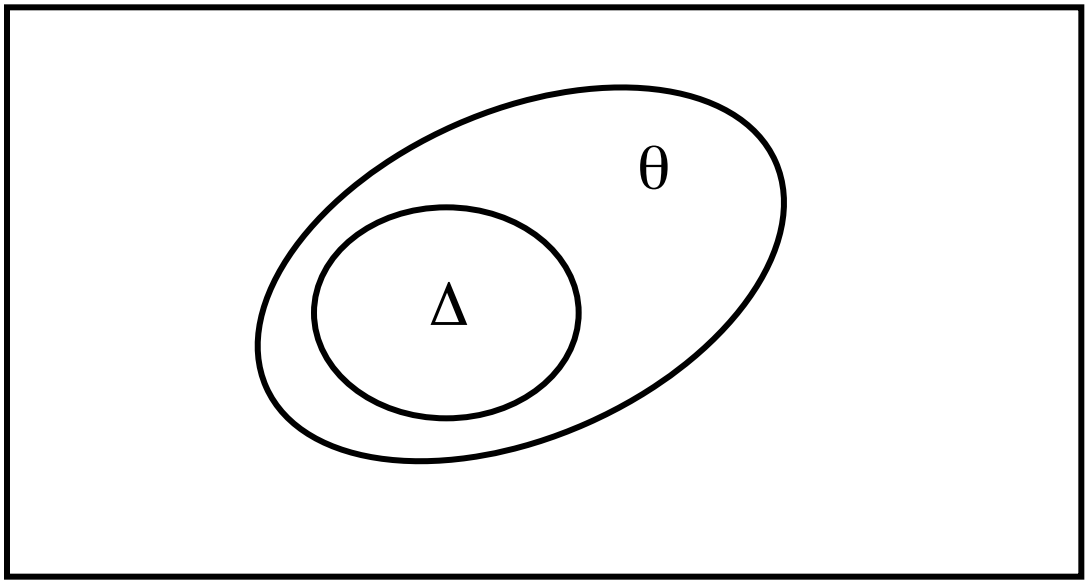
Validity and Satisfiability



All WFFs

(**not** universe of possible worlds!)

Suppose Δ is Satisfiable and Δ Entails θ



All possible worlds (models)

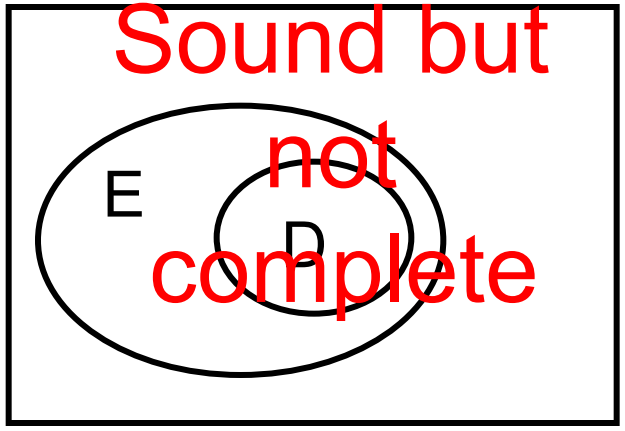
What are the possible worlds of θ?

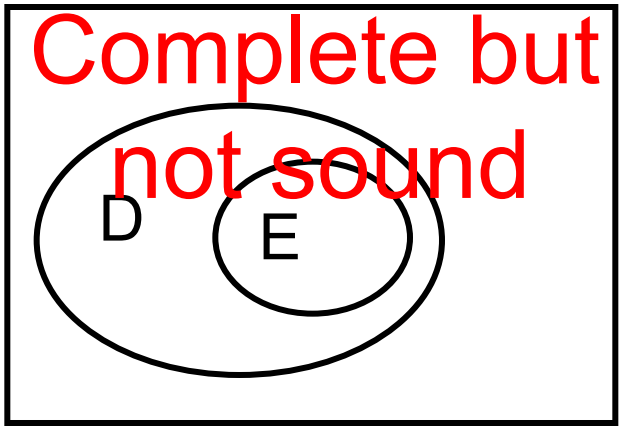
What are the possible worlds of $\neg\theta$?

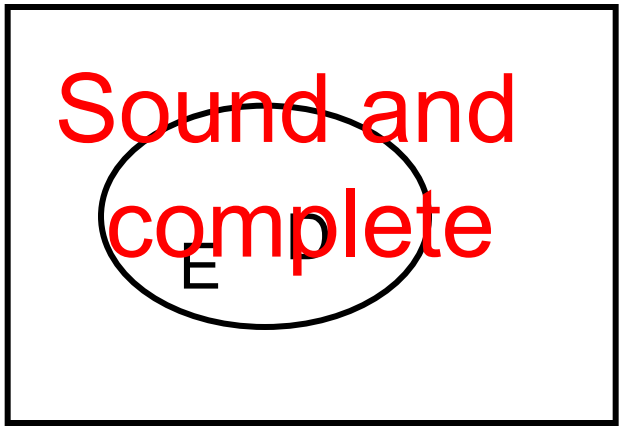
What are the possible worlds of $(\Delta \wedge \neg\theta)$?

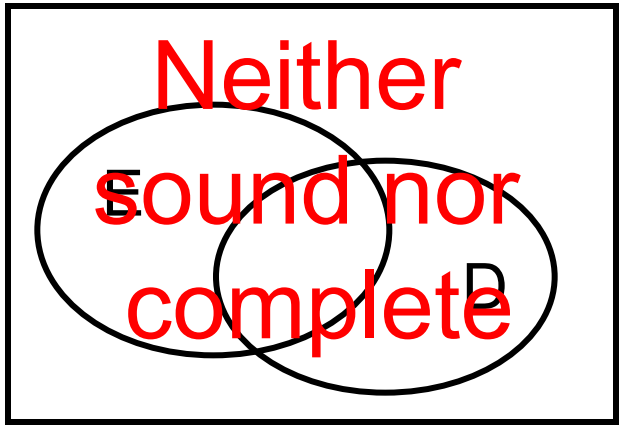
For all axiom sets we have these relationships

E: entailed WFFs D: derivable WFFs







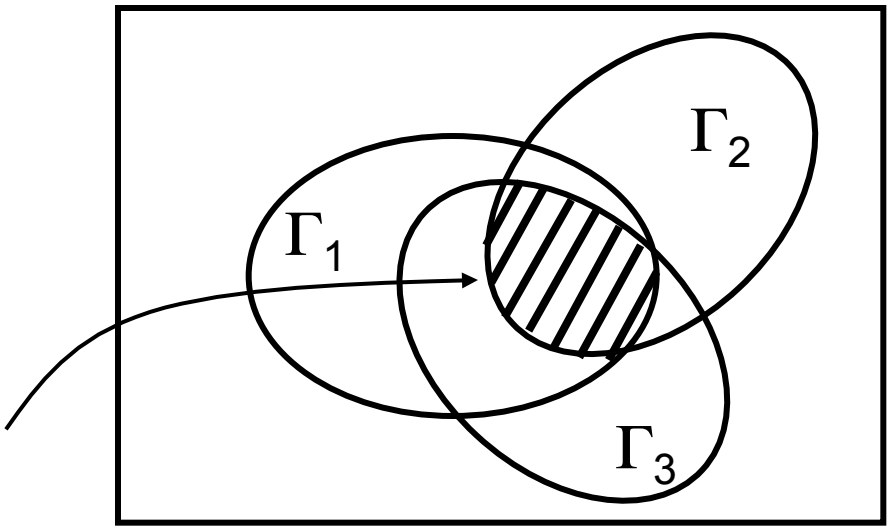


Entailment

An axiom set Δ contains 3 axioms:

$$\Delta \equiv \bigwedge_{i=1,3} \Gamma_i$$

$$(\Delta \equiv \Gamma_1 \wedge \Gamma_2 \wedge \Gamma_3)$$



Possible Worlds

Any WFF that *completely includes* this intersection is logically entailed

Shortcomings of propositional logic

We want to make generic statements:

All humans are mortal.

human and *mortal* are **properties**

x is smaller than x+1

x,1 and *x+1* refer to **entities.**

There may be an infinite number of them.

+ is a **function**

Shortcomings of propositional logic

There is at least one goal state.
(but we don't know which one….)

A queen on square X attacks a piece on square Y if…

We need a richer model

Domain: a set of entities *{ann, peter, ...,book1,...}*

Entities can have **properties**.
A property defines a (sub)set of entities:
blue= {book5, mug3, ...}

There may be **relations** between entities.
An *n*-ary relation defines a set of *n*-ary tuples of entities: *belongsTo ={<book5, peter>, <mug3, ann>,...}*

We need a richer language

Terms: refer to entities

– Variables, constants, functions,

Predicates: refer to properties of entities or relations between entities

Sentences (= propositions)
can be true or false

So…

We need a more expressive language:

- We need to talk about entities, their properties, and their relations to other entities

We need richer models:

- We need to represent entities, properties, and their relations to other entities

First-order predicate logic

Predicate logic

Syntax: What is the language of well-formed formulas of predicate logic?

Semantics: What is the interpretation of a well-formed formula in predicate logic?

Inference rules and algorithms: How can we reason with predicate logic?

Some examples

John is a student: $\text{student}(\text{john})$

All students take at least one class:

$\forall x\ \text{student}(x) \rightarrow \exists y(\text{class}(y) \wedge \text{takes}(x,y))$

There is a class that all students take:

$\exists y(\text{class}(y) \wedge \forall x\ (\text{student}(x) \rightarrow \text{takes}(x,y))$

Terms

Terms refer to entities.

They can be constants, variables, or *n*-ary functions applied to *n* terms:

john’, x, fatherOf(john’),

Predicates

Unary predicates

define properties of entities:

student(john')

***N*-ary predicates**

define relations between *n* entities:

fatherOf(john', tom')

Formulas

Formulas have truth values.

Atomic formulas consist of an n-ary predicate applied to n terms:

fatherOf(john', tom')

Formulas can be negated or joined with the same connectives as in propositional logic:

¬fatherOf(john', max');

fatherOf(john', tom') ∧ fatherOf(ann', tom')

Quantified formulas

Universal quantifier ('for all x')

All x are P: $\forall x\, P(x)$

All P are Q: $\forall x\, P(x) \rightarrow Q(x)$

Existential quantifier ('there exists an x')

There is a P: $\exists x\, P(x)$

At least one P is a Q: $\exists x\, P(x) \wedge Q(x)$

The syntax of FOL expressions

Term ⇒ Constant |
Variable |
Function(Term,...,Term)

Formula ⇒ Predicate(Term, ...Term) |
¬ Formula |
∀ Variable Formula |
∃ Variable Formula |
Formula ∧ Formula |
Formula ∨ Formula |
Formula → Formula

Why 'first order'

In first order predicate logic, variables refer only to entities. We can only quantify over entities: $\forall x\,(P(x) \rightarrow Q(x))$

In second order predicate logic, variables can also refer to predicates. We can now quantify over predicates:

$\forall x\ \forall Q\,(P(x) \rightarrow Q(x))$

NB: zero-th order predicate logic = propositional logic

Quantifier scope

$\forall x\ (P(x,y) \rightarrow Q(x))$

scope of $\forall$

x is bound by $\forall$

y is free.

Variables that are in the scope of a quantifier are **bound.** Otherwise, they are **free**.

Only formulas without free variables have truth values (='sentences/propositions')

Quantifier scope equivalences

$\forall x\ P(x) \wedge \forall y\ Q(y) \equiv \forall x\ \forall y\ [P(x) \wedge Q(y)]$
$\forall x\ \forall y\ [P(x) \wedge Q(y)] \equiv \forall y\ \forall x\ [P(x) \wedge Q(y)]$
$\forall x\ P(x) \vee \forall y\ Q(y) \equiv \forall x\ \forall y\ [P(x) \vee Q(y)]$
$\forall x\ \forall y\ [P(x) \vee Q(y)] \equiv \forall y\ \forall x\ [P(x) \vee Q(y)]$

$\exists x\ P(x) \vee \exists y\ Q(y) \equiv \exists x\ \exists y\ [P(x) \vee Q(y)]$
$\exists x\ \exists y\ [P(x) \vee Q(y)] \equiv \exists y\ \exists x\ [P(x) \vee Q(y)]$

Negation and Quantifiers

Not all x are P $\Leftrightarrow$

There is at least one x that is not P

$\neg\ \forall x\ P(x) \equiv \exists x\ \neg P(x)$

There is no x that is P $\Leftrightarrow$

All x are not P

$\neg\ \exists x\ P(x) \equiv \forall x\ \neg P(x)$

Quantifier scope ambiguities

'for each x there is a y such that P(x,y)'
$\forall x \exists y\ P(x,y)$

is **NOT EQUIVALENT** to

'there is a y such that for each x P(x,y)'
$\exists y \forall x\ P(x,y)$