



Knowledge Representation and Logic

Symbolic and probabilistic AI 01 | Propositional and first-order logic, inference rules, CNF and resolution with a worked proof, Horn clauses, semantic networks, frames, description logics and OWL reasoning

#Logic #FOL #Resolution #DescriptionLogics #KR

Why represent knowledge explicitly

Knowledge representation and reasoning



Write the rules of a domain in a precise language ("every student who studies passes") so a computer can draw conclusions that were never written down.

Pick a formalism by the trade-off you need: propositional logic for SAT, Horn rules for fast rule engines, description logics (OWL) for ontologies, full first-order logic for theorem proving.



Every formalism trades expressiveness for decidability and complexity: propositional SAT is NP-complete, OWL 2 DL is decidable but N2EXPTIME-complete, full FOL is only semi-decidable.

Logic for AI

ADVANCED

Computability

why FOL is semi-decidable

HELPFUL

Recursion and trees

Learn: algorithms 03

Search, backtracking

proof search is search · Learn: algorithms 07-08

MUST KNOW

Boolean algebra: AND, OR, NOT

Learn: maths: algebra refresher

Sets and relations

Learn: maths: discrete maths

Formalism	Can say	Reasoning cost	Used in
Propositional logic	fixed facts: Rain $\rightarrow$ WetRoad	SAT: NP-complete	SAT solvers, hardware verification
Horn clauses	rules with one conclusion	linear time (propositional)	Prolog, Datalog, rule engines
Description logics	concepts, roles, restrictions	decidable; PTIME (EL) to N2EXPTIME (OWL 2 DL)	OWL ontologies, SNOMED CT
First-order logic	objects, quantifiers, functions	semi-decidable	theorem provers (Vampire, E)
Higher-order logic	quantify over functions and sets	undecidable, interactive	Lean, Isabelle, Coq/Rocq

The knowledge-representation family, from least to most expressive

Propositional logic

Connective	Symbol	True when	Python
negation	$\neg P$	P is false	not p
conjunction	$P \wedge Q$	both true	p and q
disjunction	$P \vee Q$	at least one true	p or q
implication	$P \rightarrow Q$	P false, or Q true	(not p) or q
biconditional	$P \leftrightarrow Q$	same truth value	p == q

P	Q	$P \rightarrow Q$	$(P \rightarrow Q) \wedge P$	$((P \rightarrow Q) \wedge P) \rightarrow Q$
T	T	T	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

Tautology: true in all 4 rows (valid)

DNF (one term per true row): $(P \wedge Q) \vee (P \wedge \neg Q) \vee (\neg P \wedge Q) \vee (\neg P \wedge \neg Q)$

CNF (one clause per false row): T (no false row)

Where: T: true, F: false; $\neg$ a: not a; $a \wedge b$: a and b; $a \vee b$: a or b (or both); $a \oplus b$: exactly one; $a \rightarrow b$: false only when a is true and b false; $a \leftrightarrow b$: same value

Modus ponens as a sentence: a tautology, true in all four rows, so the inference rule is sound

Name	Equivalence	Use
Implication elimination	$P \rightarrow Q \equiv \neg P \vee Q$	first CNF step
Biconditional elimination	$P \leftrightarrow Q \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$	before implications
De Morgan	$\neg(P \wedge Q) \equiv \neg P \vee \neg Q$	push negations inward
Contrapositive	$P \rightarrow Q \equiv \neg Q \rightarrow \neg P$	modus tollens
Distribution	$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$	last CNF step; can blow up size

where P, Q, R = any propositions (sentences that are true or false) (true / false); $\equiv$ = logically equivalent: same truth value in every model (relation)

Equivalences used to rewrite sentences into CNF

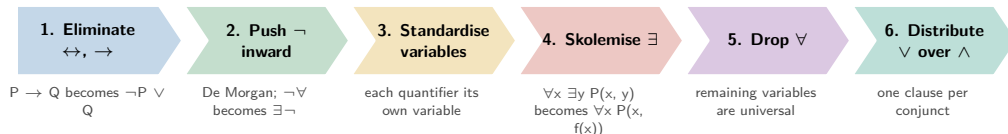
Rule	Premises	Conclusion
Modus ponens	$P \rightarrow Q, P$	Q
Modus tollens	$P \rightarrow Q, \neg Q$	$\neg P$
And-elimination	$P \wedge Q$	P
Resolution	$P \vee Q, \neg Q \vee R$	$P \vee R$
Universal instantiation (FOL)	$\forall x \text{ Human}(x) \rightarrow \text{Mortal}(x)$	$\text{Human}(\text{Ana}) \rightarrow \text{Mortal}(\text{Ana})$
Existential instantiation (FOL)	$\exists x \text{ Owns}(\text{Ana}, x)$	$\text{Owns}(\text{Ana}, C1), C1 \text{ a new constant (Skolem)}$

Inference rules: from the premises above the line, derive the conclusion

🔍 First-order logic and resolution

English	FOL	Trap
All students are smart	$\forall x \text{ Student}(x) \rightarrow \text{Smart}(x)$	not $\forall x \text{ Student}(x) \wedge \text{Smart}(x)$
Some student is smart	$\exists x \text{ Student}(x) \wedge \text{Smart}(x)$	not $\exists x \text{ Student}(x) \rightarrow \text{Smart}(x)$
Nobody likes Bob	$\neg \exists x \text{ Likes}(x, \text{Bob})$, i.e. $\forall x \neg \text{Likes}(x, \text{Bob})$	negation flips the quantifier
Everyone has a mother	$\forall x \exists y \text{ Mother}(y, x)$	order matters: $\exists y \forall x$ = one mother of all
Ana's father is a doctor	$\text{Doctor}(\text{father}(\text{Ana}))$	father is a function, not a predicate

Translating English into first-order logic



Converting first-order sentences to clauses (CNF) before resolution

PROBLEM

KB: every student who studies passes; whoever passes graduates; Ana is a student; Ana studies.
Prove $\text{Graduates}(\text{Ana})$ by resolution refutation.

1. Resolve C5 with C2

$\Rightarrow$ unify $\text{Graduates}(y)$ with $\text{Graduates}(\text{Ana})$: $\{y/\text{Ana}\} \rightarrow C6$:
 $\neg \text{Passes}(\text{Ana})$

why: complementary literals Graduates / $\neg \text{Graduates}$ cancel

2. Resolve C6 with C1

$\Rightarrow \{x/\text{Ana}\} \rightarrow C7$: $\neg \text{Student}(\text{Ana}) \vee \neg \text{Studies}(\text{Ana})$

why: $\text{Passes}(\text{Ana})$ cancels $\neg \text{Passes}(\text{Ana})$

3. Resolve C7 with C3

$\Rightarrow C8$: $\neg \text{Studies}(\text{Ana})$

why: $\text{Student}(\text{Ana})$ is a fact

4. Resolve C8 with C4

$\Rightarrow$ empty clause $\square$

why: $\text{Studies}(\text{Ana})$ and $\neg \text{Studies}(\text{Ana})$ contradict

GIVEN

- C1: $\neg \text{Student}(x) \vee \neg \text{Studies}(x) \vee \text{Passes}(x)$
- C2: $\neg \text{Passes}(y) \vee \text{Graduates}(y)$
- C3: $\text{Student}(\text{Ana})$ C4: $\text{Studies}(\text{Ana})$
- C5 (negated goal): $\neg \text{Graduates}(\text{Ana})$

ANSWER

KB $\wedge \neg \text{Graduates}(\text{Ana})$ is unsatisfiable, so KB $\models \text{Graduates}(\text{Ana})$

Check: Forward chaining reaches the same result: C3, C4 fire C1 $\rightarrow$ $\text{Passes}(\text{Ana})$, which fires C2 $\rightarrow$ $\text{Graduates}(\text{Ana})$.

The same KB as Horn clauses in Prolog (backward chaining, depth-first)

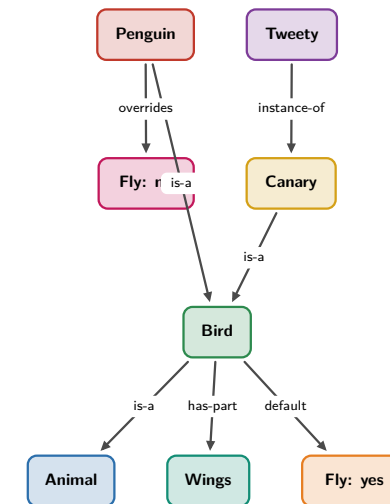
```

passes(X) :- student(X), studies(X).
graduates(X) :- passes(X).
student(ana).
studies(ana).
% ?- graduates(ana). -> true
% ?- graduates(bob). -> false (closed world: not provable = false)
  
```

💡 TIP

Resolution is **refutation-complete**: if $\text{KB} \models \alpha$, deriving the empty clause from $\text{KB} \wedge \neg \alpha$ always succeeds. If α does not follow, the search may run forever in FOL (semi-decidability). Practical provers (Vampire, E) add ordering, subsumption and set-of-support strategies.

📦 Structured representations: networks, frames, description logics



Semantic network with inheritance: Tweety inherits 'has wings' from Bird; Penguin overrides 'can fly' (a default with an exception, which classical logic cannot express without extra machinery)

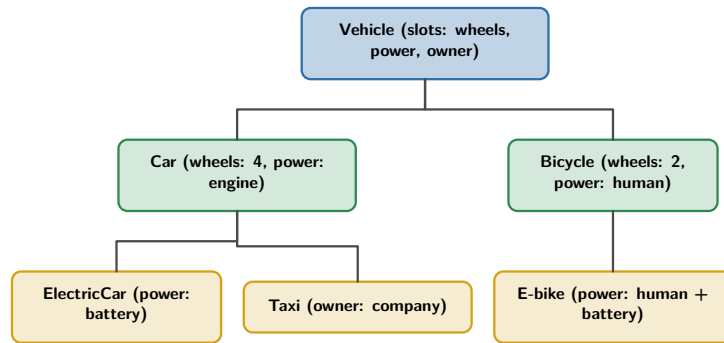
Check entailment with SymPy (propositional): KB entails q iff $\text{KB} \wedge \neg q$ is unsatisfiable

```

from sympy import symbols
from sympy.logic.boolalg import Implies, And, Not, to_cnf
from sympy.logic.inference import satisfiable
  
```

```

rain, wet, slip = symbols("rain wet slip")
kb = And(Implies(rain, wet), Implies(wet, slip), rain)
print(to_cnf(kb)) # clauses
print(satisfiable(And(kb, Not(slip)))) # False -> kb |= slip
  
```



A frame hierarchy: each frame has slots; children inherit slot values unless they redefine them (defaults)

DL syntax	OWL 2	Meaning
$C \sqcap D$	ObjectIntersectionOf	things in both C and D
$C \sqcup D$	ObjectUnionOf	things in C or D
$\neg C$	ObjectComplementOf	things not in C
$\exists r.C$	ObjectSomeValuesFrom	has at least one r-filler in C
$\forall r.C$	ObjectAllValuesFrom	all r-fillers (if any) are in C
$C \subseteq D$	SubClassOf	every C is a D (TBox axiom)
$a : C, (a, b) : r$	ClassAssertion, PropertyAssertion	facts about individuals (ABox)

Description logic ALC: building blocks, with the OWL 2 keyword and a reading

Worked example: classification by a DL reasoner

TBox: **Parent** = **Person** $\sqcap$ $\exists$ hasChild.**Person**; **Mother** = **Parent** $\sqcap$ **Female** (= is DL equivalence, OWL EquivalentClasses). ABox: ana : **Person** $\sqcap$ **Female**, (ana, leo) : hasChild, leo : **Person**. The reasoner infers ana : **Parent** (she has a child who is a person), then ana : **Mother**, and **Mother** $\subseteq$ **Parent** (subsumption) without being told. Under the open-world assumption, 'leo has no children' is **unknown**, not false. Reasoners: HermiT and Pellet (OWL 2 DL), ELK (OWL 2 EL, millions of classes such as SNOMED CT).

Validity
true in every model (tautology)

vs

Satisfiability
true in at least one model

Tell: All models or some model? α valid $\Leftrightarrow \neg\alpha$ unsatisfiable

Entailment KB $\models \alpha$
semantic: truth in all models of KB

vs

Derivation KB $\vdash \alpha$
syntactic: a proof with inference rules

Tell: Sound + complete procedures make them coincide

Closed world
not provable = false (SQL, Prolog)

vs

Open world
not stated = unknown (OWL, the web)

Tell: Can missing data mean 'no'?

⚠ Traps and recap

MYTH

✗ $P \rightarrow Q$ means P causes Q

FACT

✓ It only rules out P true with Q false

'If the Moon is cheese then $2 + 2 = 4$ ' is true; material implication carries no causality.

✗ First-order logic is decidable, just slow

✓ It is semi-decidable

A prover finds every valid entailment but may never stop on a non-entailment (Church and Turing, 1936).

✗ Ontologies are just class hierarchies

✓ They carry axioms a reasoner uses

Definitions (EquivalentClasses), disjointness and restrictions let the reasoner infer classes and find contradictions.

✗ Logic cannot handle exceptions

✓ Non-monotonic logics and defaults can

Default logic, negation as failure (Prolog) and answer set programming retract conclusions when new facts arrive.

Key takeaways

01

Entailment = truth in all models

check via unsatisfiability of $KB \wedge \neg\alpha$

02

Resolution on CNF is refutation-complete

unification handles variables

03

Horn clauses buy speed

Prolog, Datalog, rule engines

04

Description logics = decidable FOL fragments

OWL, TBox + ABox, open world

05

Expressiveness costs decidability

propositional $\rightarrow$ DL $\rightarrow$ FOL $\rightarrow$ higher-order

Q1 How do you prove $KB \models \alpha$ by resolution?

A Add $\neg\alpha$ to the KB in CNF and derive the empty clause; a contradiction shows every model of KB satisfies α .

Q2 Correct FOL for 'some dogs bark'?

A $\exists x \text{ Dog}(x) \wedge \text{Barks}(x)$; with $\rightarrow$ it would be true whenever any non-dog exists.

Q3 What does Skolemisation do?

A Replaces an existential variable with a new constant or a function of the enclosing universal variables, e.g. $\forall x \exists y \text{ Loves}(x, y) \rightarrow \forall x \text{ Loves}(x, f(x))$.

Q4 What is a Horn clause and why care?

A A clause with at most one positive literal; forward and backward chaining on Horn KBs run in linear time (propositional).

Q5 Most general unifier of Knows(John, x) and Knows(y, Mother(y))?

A $\{y/\text{John}, x/\text{Mother}(\text{John})\}$.

Q6 TBox vs ABox?

A TBox: terminology, axioms between concepts ($\text{Mother} \subseteq \text{Parent}$). ABox: assertions about individuals ($\text{ana} : \text{Mother}$).

Q7 Which OWL 2 profile scales to medical ontologies with hundreds of thousands of classes?

A OWL 2 EL: polynomial-time classification, used for SNOMED CT with the ELK reasoner.

Q8 Why can semantic networks not be translated naively to FOL?

A Default inheritance with exceptions (penguins do not fly) is non-monotonic; classical FOL never retracts a conclusion.

Q9 Complexity of propositional satisfiability?

A NP-complete (Cook-Levin, 1971); still solved for millions of clauses in practice by CDCL solvers (sheet 02).