

CCES – College of Computer Engineering and Sciences

Prince Sattam Bin Abdulaziz University

CS600: Advanced AI

Propositional Logic

Chapter 4 — Syntax, Semantics, and Automated Inference

Dr. Khalil Chebil

Artificial Intelligence Course

Chapter Outline | What we will cover

Foundations & Syntax
Semantics & Normal Forms
Model Checking, Resolution & DPLL

Horn Clauses & Chaining
Summary

Learning Objectives

- ▶ Translate English statements into **propositional formulas** and evaluate them with **truth tables**
- ▶ Decide **satisfiability**, **validity** and **entailment**; convert formulas to **CNF**
- ▶ Prove entailment by **model checking**, **resolution** and **DPLL**, step by step
- ▶ Reason with **Horn clauses** using **forward** and **backward chaining**



Section 1

Foundations & Syntax

Why Logic in AI? | A formal language for knowledge and reasoning

A knowledge-based agent

TELL the knowledge base (KB) facts and rules;
ASK it questions. An *inference* procedure answers using only what follows logically.

Why logic?

Declarative: state *what* is true, not how to compute it.

Compositional: meaning of the whole from its parts.

Unambiguous and **explainable:** every conclusion has a proof.

Running example: the Wumpus world

A cave of squares. A *pit* makes the neighbouring squares *breezy*.

You stand in [1, 1] and feel **no breeze**.

You move to [2, 1] and feel **a breeze**.

Is [1, 2] safe? Is there a pit in [2, 2]?

Applications: expert systems and diagnosis, hardware and software verification, planning, scheduling, circuit design (SAT).

Syntax: Propositions & Connectives | Building well-formed formulas

Atomic propositions $P, Q, R, \dots$ stand for statements that are *true* or *false*.

Symbol	Name	Reads as
$\neg P$	negation	not P
$P \wedge Q$	conjunction	P and Q
$P \vee Q$	disjunction	P or Q (or both)
$P \rightarrow Q$	implication	if P then Q
$P \leftrightarrow Q$	biconditional	P if and only if Q

Well-formed formulas

Every atom is a formula. If α, β are formulas, so are $\neg\alpha$, $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $(\alpha \rightarrow \beta)$, $(\alpha \leftrightarrow \beta)$.

$\neg(P \wedge \neg Q)$ ✓ $P \wedge \wedge Q$ ✗

Precedence (tightest first)

$\neg \quad \wedge \quad \vee \quad \rightarrow \quad \leftrightarrow$
 $\neg P \wedge Q \rightarrow R$ means $((\neg P) \wedge Q) \rightarrow R$.

Translating English into Logic | Think first, then reveal

Symbols: R = it rains, U = I take an umbrella, W = I get wet, S = the sprinkler is on.

English	Formula
1 If it rains and I have no umbrella, I get wet.	
2 I get wet only if it rains or the sprinkler is on.	
3 I take an umbrella exactly when it rains.	
4 It is not the case that it rains and I stay dry.	
5 Unless it rains, the sprinkler is on.	

Translating English into Logic | Think first, then reveal

Symbols: R = it rains, U = I take an umbrella, W = I get wet, S = the sprinkler is on.

English	Formula
1 If it rains and I have no umbrella, I get wet.	$(R \wedge \neg U) \rightarrow W$
2 I get wet only if it rains or the sprinkler is on.	$W \rightarrow (R \vee S)$ (“only if” = then)
3 I take an umbrella exactly when it rains.	$U \leftrightarrow R$
4 It is not the case that it rains and I stay dry.	$\neg(R \wedge \neg W)$, equivalently $R \rightarrow W$
5 Unless it rains, the sprinkler is on.	$\neg R \rightarrow S$, equivalently $R \vee S$

Section 2

Semantics & Normal Forms

Semantics: Models & Truth Tables | A model assigns a truth value to every symbol

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \rightarrow Q$	$P \leftrightarrow Q$
T	T	F	T	T	T	T
T	F	F	F	T	F	F
F	T	T	F	T	T	F
F	F	T	F	F	T	T

n symbols give 2^n models (rows).

The puzzling row: $P \rightarrow Q$

$P \rightarrow Q$ is false *only* when P is true and Q false. “If pigs fly, I am king” is *true*: a promise is not broken when its condition never happens.

Useful identities

$$P \rightarrow Q \equiv \neg P \vee Q$$

$$P \leftrightarrow Q \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$$

Worked Truth Table | Is $(P \rightarrow Q) \wedge P \rightarrow Q$ always true? (modus ponens)

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

Conclusion

True in all 4 models: the formula is **valid** (a tautology). This is why **modus ponens** is a sound rule.

Try it: affirming the consequent

Is $(P \rightarrow Q) \wedge Q \rightarrow P$ valid?

Worked Truth Table | Is $(P \rightarrow Q) \wedge P \rightarrow Q$ always true? (modus ponens)

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

Conclusion

True in all 4 models: the formula is **valid** (a tautology). This is why **modus ponens** is a sound rule.

Try it: affirming the consequent

Is $(P \rightarrow Q) \wedge Q \rightarrow P$ valid?

No: row $P = F, Q = T$ makes it false.
“Wet ground, so it rained” is a fallacy (the sprinkler).

Logical Equivalences | Same truth value in every model: $\alpha \equiv \beta$

Law	Equivalence
Double negation	$\neg\neg P \equiv P$
De Morgan	$\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
De Morgan	$\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
Implication	$P \rightarrow Q \equiv \neg P \vee Q$
Contrapositive	$P \rightarrow Q \equiv \neg Q \rightarrow \neg P$
Distributivity	$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$
Distributivity	$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$

Worked: simplify $\neg(P \rightarrow Q)$

$$\neg(P \rightarrow Q) \equiv \neg(\neg P \vee Q) \equiv \neg\neg P \wedge \neg Q \equiv P \wedge \neg Q$$

“It’s false that rain implies wet” means: it rains and I’m dry.

Not equivalent!

Converse $Q \rightarrow P$ and inverse $\neg P \rightarrow \neg Q$ differ from $P \rightarrow Q$ (they agree with each other).

Three properties of a formula

Satisfiable: true in *some* model ($P \wedge Q$).

Unsatisfiable: true in *no* model ($P \wedge \neg P$).

Valid: true in *every* model ($P \vee \neg P$).

Entailment $KB \models \alpha$

α is true in **every model in which KB is true.**

$\{P \rightarrow Q, P\} \models Q$

Refutation principle

$KB \models \alpha$ iff $KB \wedge \neg \alpha$ is **unsatisfiable**.

Classify each formula:

Formula	Answer
$P \rightarrow P$	
$(P \vee Q) \wedge \neg P$	
$(P \rightarrow Q) \wedge P \wedge \neg Q$	
$(P \rightarrow Q) \vee (Q \rightarrow P)$	

Satisfiability, Validity & Entailment | The vocabulary of reasoning

Three properties of a formula

Satisfiable: true in *some* model ($P \wedge Q$).

Unsatisfiable: true in *no* model ($P \wedge \neg P$).

Valid: true in *every* model ($P \vee \neg P$).

Entailment $KB \models \alpha$

α is true in **every model in which KB is true.**

$\{P \rightarrow Q, P\} \models Q$

Refutation principle

$KB \models \alpha$ iff $KB \wedge \neg \alpha$ is **unsatisfiable**.

Classify each formula:

Formula	Answer
$P \rightarrow P$	valid
$(P \vee Q) \wedge \neg P$	satisfiable ($P=F, Q=T$)
$(P \rightarrow Q) \wedge P \wedge \neg Q$	unsatisfiable
$(P \rightarrow Q) \vee (Q \rightarrow P)$	valid (surprisingly)

Entailment in the Wumpus World | Check every model

Knowledge base

$\neg B_{11}$ (no breeze in [1,1])
 $B_{11} \leftrightarrow (P_{12} \vee P_{21})$
 B_{21} (breeze in [2,1])
 $B_{21} \leftrightarrow (P_{11} \vee P_{22} \vee P_{31})$
 $\neg P_{11}$ (start square is safe)

7 symbols $\Rightarrow 2^7 = 128$ models. Only **3** make the KB true.

P_{11}	P_{12}	P_{21}	P_{22}	P_{31}
F	F	F	T	F
F	F	F	F	T
F	F	F	T	T

Answers

P_{12} is false in all 3: $KB \models \neg P_{12}$, so [1,2] is **safe**.

P_{22} is true in 2, false in 1: **not entailed** either way. We don't know.

Conjunctive Normal Form (CNF) | AND of ORs: the input to resolution and SAT

Vocabulary

Literal: P or $\neg P$.

Clause: a disjunction of literals.

CNF: a conjunction of clauses.

$(P \vee Q) \wedge (\neg P \vee R)$ ✓ $(P \wedge Q) \vee R$ ✗

Conversion recipe

1. Eliminate $\leftrightarrow$ then $\rightarrow$
2. Push $\neg$ inward (De Morgan, $\neg\neg$)
3. Distribute $\vee$ over $\wedge$

Worked: $B_{11} \leftrightarrow (P_{12} \vee P_{21})$

1a. $(B_{11} \rightarrow (P_{12} \vee P_{21})) \wedge ((P_{12} \vee P_{21}) \rightarrow B_{11})$

1b. $(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg(P_{12} \vee P_{21}) \vee B_{11})$

2. $(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge ((\neg P_{12} \wedge \neg P_{21}) \vee B_{11})$

3. $(\neg B_{11} \vee P_{12} \vee P_{21}) \wedge (\neg P_{12} \vee B_{11}) \wedge (\neg P_{21} \vee B_{11})$

Three clauses. In words: a breeze needs a pit nearby, and each nearby pit causes a breeze.

Section 3

Model Checking, Resolution & DPLL

Model Checking | Enumerate every model

TT-Entails?

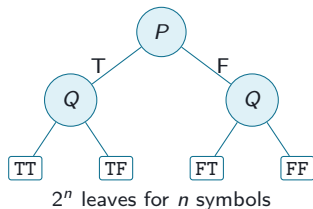
To check $KB \models \alpha$, try every assignment of the n symbols: whenever KB is true, α must be true too.

Properties

Sound and **complete**: it always gives the right answer.

Cost: $\mathcal{O}(2^n)$

Wumpus: $2^7 = 128$ models, easy. 30 symbols: 10^9 models. 100 symbols: 10^{30} , hopeless.



Next: methods that *prove* entailment by manipulating formulas (resolution), or search models cleverly (DPLL).

Resolution | One sound inference rule, refutation-complete

The rule

$$\frac{\ell \vee A \quad \neg \ell \vee B}{A \vee B}$$

Cancel a complementary pair; keep everything else. Remove duplicate literals.

Examples

$(P \vee Q), (\neg P \vee R) \vdash (Q \vee R)$

$(P), (\neg P \vee Q) \vdash (Q)$

modus ponens

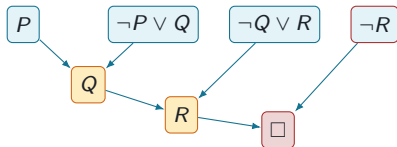
$(P), (\neg P) \vdash \square$

contradiction

Proof by refutation

Convert $KB \wedge \neg \alpha$ to CNF; resolve until the **empty clause** $\square$ appears.

Prove $KB \models R$ for
 $KB = \{P \rightarrow Q, Q \rightarrow R, P\}$:



$\neg R$ is the negated goal (red border). Reaching $\square$ means $KB \wedge \neg R$ is unsatisfiable, so $KB \models R$.

Worked Resolution: Is [1,2] Safe? | Prove $KB \models \neg P_{12}$

Clauses

From the CNF slide:

(1) $\neg B_{11} \vee P_{12} \vee P_{21}$

(2) $\neg P_{12} \vee B_{11}$

(3) $\neg P_{21} \vee B_{11}$

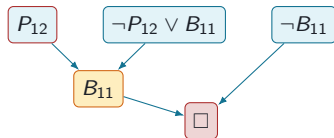
(4) $\neg B_{11}$ (percept)

Negated goal: (5) P_{12}

Derivation

(5) + (2) $\Rightarrow$ (6) B_{11}

(6) + (4) $\Rightarrow \square$



Intuition

"A pit in [1,2] would cause a breeze in [1,1], but there is none." Resolution found the two-step argument automatically; clauses (1) and (3) were not needed.

DPLL: Smarter Satisfiability Search | Backtracking with three shortcuts

Shortcuts

Early termination: a clause is true once one literal is; a clause with all literals false fails the branch.

Pure symbol: a symbol with only one polarity left: set it to make those literals true.

Unit clause: one unassigned literal left: it must be true.

Split: otherwise try $X = T$, then $X = F$.

Still exponential in the worst case, but the basis of modern **CDCL** solvers that handle millions of variables.

Worked: $\{A \vee B, \neg A \vee C, \neg B \vee \neg C\}$

Pure? A, B, C all appear with both signs: none.

Unit? none. **Split** $A = T$:

$A \vee B$ satisfied; $\neg A \vee C$ becomes unit C

$C = T$: $\neg B \vee \neg C$ becomes unit $\neg B$

$B = F$: every clause satisfied.

SAT with $A = T, B = F, C = T$ (one split, no backtracking).

Check yourself

Is there another model?

DPLL: Smarter Satisfiability Search | Backtracking with three shortcuts

Shortcuts

Early termination: a clause is true once one literal is; a clause with all literals false fails the branch.

Pure symbol: a symbol with only one polarity left: set it to make those literals true.

Unit clause: one unassigned literal left: it must be true.

Split: otherwise try $X = T$, then $X = F$.

Still exponential in the worst case, but the basis of modern **CDCL** solvers that handle millions of variables.

Worked: $\{A \vee B, \neg A \vee C, \neg B \vee \neg C\}$

Pure? A, B, C all appear with both signs: none.

Unit? none. **Split** $A = T$:

$A \vee B$ satisfied; $\neg A \vee C$ becomes unit C

$C = T$: $\neg B \vee \neg C$ becomes unit $\neg B$

$B = F$: every clause satisfied.

SAT with $A = T, B = F, C = T$ (one split, no backtracking).

Check yourself

Is there another model? Yes: $A = F, B = T, C = F$.

Section 4

Horn Clauses & Chaining

Horn Clauses | Restricted, but inference becomes linear-time

Definition

A **Horn clause** has *at most one* positive literal.

Definite clause (exactly one): a rule

$$\neg P \vee \neg Q \vee R \equiv (P \wedge Q) \rightarrow R$$

Fact: a single positive literal P

Goal clause (none): $\neg P \vee \neg Q$

Why they matter

Most rule bases are naturally Horn (“if conditions then conclusion”). Entailment is **linear** in the KB size. Basis of **Prolog**.

Horn or not?

Clause	Answer
$\neg A \vee \neg B \vee C$	
$A \vee B$	
$\neg A \vee \neg B$	
$A \rightarrow (B \vee C)$	
$(A \vee B) \rightarrow C$	

Horn Clauses | Restricted, but inference becomes linear-time

Definition

A **Horn clause** has *at most one* positive literal.

Definite clause (exactly one): a rule

$$\neg P \vee \neg Q \vee R \equiv (P \wedge Q) \rightarrow R$$

Fact: a single positive literal P

Goal clause (none): $\neg P \vee \neg Q$

Why they matter

Most rule bases are naturally Horn (“if conditions then conclusion”). Entailment is **linear** in the KB size. Basis of **Prolog**.

Horn or not?

Clause	Answer
$\neg A \vee \neg B \vee C$	Horn (rule $A \wedge B \rightarrow C$)
$A \vee B$	not Horn (two positive)
$\neg A \vee \neg B$	Horn (goal clause)
$A \rightarrow (B \vee C)$	not Horn: $\neg A \vee B \vee C$
$(A \vee B) \rightarrow C$	Horn: two clauses $\neg A \vee C, \neg B \vee C$

Forward Chaining, Step by Step | Data-driven: fire every rule whose premises are known

KB (classic AIMA example)

$P \rightarrow Q$

$L \wedge M \rightarrow P$

$B \wedge L \rightarrow M$

$A \wedge P \rightarrow L$

$A \wedge B \rightarrow L$

Facts: A, B . Query: Q ?

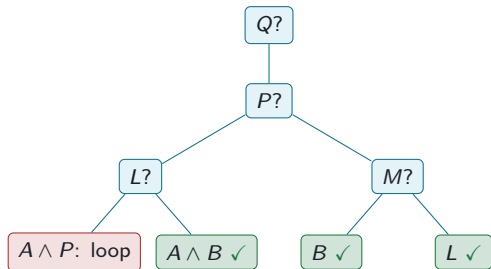
Each rule keeps a *count* of unproved premises; it fires when the count reaches 0.

Step	Rule fired	New fact
1	$A \wedge B \rightarrow L$	L
2	$B \wedge L \rightarrow M$	M
3	$L \wedge M \rightarrow P$	P
4	$P \rightarrow Q$	Q ✓ query proved

Observations

$A \wedge P \rightarrow L$ becomes ready at step 3 but adds nothing (L is known). Forward chaining derives *everything* entailed, relevant or not.

Backward Chaining, Same KB | Goal-driven: work back from the query



Trace

Q needs P ; P needs L and M .

L via $A \wedge P$? P is already a pending goal:

loop, skip.

L via $A \wedge B$: both facts. ✓

M via $B \wedge L$: both known. ✓ So Q is proved.

Two safeguards

Keep the current goal stack to avoid infinite loops; cache proved and failed subgoals to avoid repeated work.

Animal classification (notebook)

Facts: hasFur, produceMilk
 $\text{hasFur} \wedge \text{produceMilk} \rightarrow \text{mammal}$
 $\text{mammal} \wedge \text{eatsMeat} \rightarrow \text{carnivore}$

mammal?
carnivore?

Bird KB (lab)

Facts: has_feathers, lays_eggs
 $\text{has_feathers} \rightarrow \text{is_bird}$
 $\text{is_bird} \wedge \text{lays_eggs} \rightarrow \text{can_fly}$
 $\text{is_bird} \wedge \text{swims} \rightarrow \text{is_penguin}$

is_bird? can_fly? is_penguin?

Animal classification (notebook)

Facts: hasFur, produceMilk
 $\text{hasFur} \wedge \text{produceMilk} \rightarrow \text{mammal}$
 $\text{mammal} \wedge \text{eatsMeat} \rightarrow \text{carnivore}$

mammal? ✓ derivable

carnivore? ✗ eatsMeat is not known

Bird KB (lab)

Facts: has_feathers, lays_eggs
 $\text{has_feathers} \rightarrow \text{is_bird}$
 $\text{is_bird} \wedge \text{lays_eggs} \rightarrow \text{can_fly}$
 $\text{is_bird} \wedge \text{swims} \rightarrow \text{is_penguin}$

is_bird? can_fly? is_penguin? ✓ ✓ ✗

Modelling lesson

“Not derivable” means *unknown*, not false. And the bird rules are wrong for penguins: add swims and both can_fly and is_penguin hold. Propositional logic is **monotonic**; it cannot express “birds fly *unless* they are penguins” without extra conditions.

Forward vs. Backward Chaining | Pick by the question you are asking

Aspect	Forward	Backward
Starts from	known facts	the query
Driven by	data	goals
Derives	everything entailed	only what the query needs
Cost	linear in KB size	often much less than linear
Typical use	monitoring, alarms	diagnosis, question answering

Which one?

A smart-home controller reacting to new sensor readings:

A help desk asking “why won’t my laptop boot?”:

Both are sound and complete for definite (Horn) clauses.

Forward vs. Backward Chaining | Pick by the question you are asking

Aspect	Forward	Backward
Starts from	known facts	the query
Driven by	data	goals
Derives	everything entailed	only what the query needs
Cost	linear in KB size	often much less than linear
Typical use	monitoring, alarms	diagnosis, question answering

Which one?

A smart-home controller reacting to new sensor readings: **forward**.

A help desk asking “why won’t my laptop boot?”: **backward**.

Both are sound and complete for definite (Horn) clauses.



Section 5 Summary

Inference Methods Compared | Choose by problem shape

Method	Complete?	Input	Cost	Best for
Truth tables	yes	any formula	$\mathcal{O}(2^n)$	small problems, teaching
Resolution	refutation-complete	CNF	exponential (worst)	proofs that explain why
DPLL / CDCL	yes	CNF	exponential (worst), fast in practice	large SAT problems
Forward chaining	yes, for Horn	Horn clauses	linear	deriving all consequences
Backward chaining	yes, for Horn	Horn clauses	linear	answering one query

The big picture

All methods are **sound** and decide the same question, $KB \models \alpha$. They trade generality (any formula) against speed (Horn clauses only).

1. “Only if” backwards

“ P only if Q ” is $P \rightarrow Q$.

2. Affirming the consequent

From $P \rightarrow Q$ and Q , nothing follows about P .

3. Forgetting to negate the goal

Resolution adds $\neg\alpha$, then looks for $\square$.

4. Resolving two literals at once

$(P \vee Q), (\neg P \vee \neg Q)$ do not give $\square$.

5. Unknown is not false

If chaining cannot derive α , the KB is silent about it.

6. Naming the rule wrong

$(P \vee Q) \wedge \neg P \Rightarrow Q$ is *disjunctive syllogism*, not modus tollens.

Practice Problems | Answers on the next click

Problem	Answer
1 Convert $P \rightarrow (Q \wedge R)$ to CNF.	
2 Resolve $(A \vee B \vee C)$ with $(\neg B \vee D)$.	
3 Does $\{P \vee Q, \neg P\} \models Q$? Prove by resolution.	
4 DPLL on $\{A \vee B, \neg A, \neg B \vee C\}$: first step?	
5 How many models does a KB with 10 symbols have?	

Practice Problems | Answers on the next click

Problem	Answer
1 Convert $P \rightarrow (Q \wedge R)$ to CNF.	$(\neg P \vee Q) \wedge (\neg P \vee R)$
2 Resolve $(A \vee B \vee C)$ with $(\neg B \vee D)$.	$A \vee C \vee D$
3 Does $\{P \vee Q, \neg P\} \models Q$? Prove by resolution.	Add $\neg Q$; $(P \vee Q) + (\neg P)$ gives Q ; $Q + \neg Q$ gives $\square$. Yes.
4 DPLL on $\{A \vee B, \neg A, \neg B \vee C\}$: first step?	Unit clause $\neg A$: $A = F$, then unit B , then unit C . SAT.
5 How many models does a KB with 10 symbols have?	$2^{10} = 1024$

Chapter Summary | Key takeaways

Represent

Atoms and five connectives; models are rows of the truth table; CNF is the standard input form.

Reason

$KB \models \alpha$ iff $KB \wedge \neg \alpha$ is unsatisfiable. Check models, resolve to $\square$, or run DPLL. Horn KBs: forward or backward chaining in linear time.

Limitation

No objects, no variables: “every pit makes its neighbours breezy” needs one sentence per square. Wumpus on a 4×4 board already needs dozens of rules.

Looking ahead

Ch. 5: **first-order logic** adds objects, predicates and quantifiers.

Thank you!

Questions & Discussion

Next: Chapter 5 — First-Order Logic