

CSE 592 - Discussion 3

Propositional & First-Order Logic

Seung Hyun Lee / Fall 26

Logistics

Assignment 1 is due September 23 at 11:59 p.m.

Project proposal is due September 30 at 11:59 p.m.

Today: propositional logic and first-order logic.

We will practice inference, SAT reasoning, and quantifiers.

Recap: propositions and connectives

A proposition is a statement with a True or False value.

P: the door is open. Q: the light is on.

$\neg P$: NOT P. $P \wedge Q$: both P and Q.

$P \vee Q$: at least one is true, including both.

$P \rightarrow Q$: if P is true, Q must be true.

$P \leftrightarrow Q$: P and Q have the same truth value.

A knowledge base (KB) collects statements we assume true.

Possible worlds and incomplete knowledge

L: a laptop is in the bag. T: a tablet is in the bag.

KB = $\{L \vee T\}$ allows laptop only, tablet only, or both.

It rules out a bag with neither device.

Q: Can we conclude L? Can we conclude $\neg L$?

A: Neither follows. Both possibilities remain.

L=False, T=True is a countermodel to $KB \models L$.

L=True, T=False is a countermodel to $KB \models \neg L$.

Recap: models and entailment

A model is an assignment that makes every statement in KB true.

SAT: at least one model exists.

UNSAT: no assignment satisfies all the requirements.

$KB \models q$: q is true in every model of KB.

Example: $KB = \{L \vee T\}$ is SAT, but it does not entail L .

The model $L=False, T=True$ satisfies KB and makes L false.

One model proves SAT. One countermodel disproves entailment.

Entailment through counterexamples

$KB = \{L \vee T, \neg T\}$. Could the laptop also be absent?

$(L \vee T) \wedge \neg T \wedge \neg L$ is UNSAT, so $KB \models L$.

Neither device would be present, violating $L \vee T$.

In general: $KB \models q \Leftrightarrow KB \wedge \neg q$ is UNSAT.

Q: If $KB \wedge \neg q$ is SAT, what have we learned?

A: KB does not entail q. A model makes KB true and q false.

Other models may make q true. This does not establish $KB \models \neg q$.

Recap: literals, clauses, and CNF

Literal: an atom or its negation, such as P or $\neg Q$.

Clause: an OR of literals, such as $P \vee \neg Q$.

CNF: an AND of clauses, such as $(P \vee \neg Q) \wedge R$.

Conversion: $P \rightarrow (Q \wedge R)$

Eliminate $\rightarrow$: $\neg P \vee (Q \wedge R)$

Distribute OR over AND: $(\neg P \vee Q) \wedge (\neg P \vee R)$

When P is true, both Q and R are required.

CNF as a checklist of requirements

An event needs a drink, an activity, and an indoor location.

$$(C \vee T) \wedge (M \vee G) \wedge \neg O$$

C/T: coffee/tea. M/G: music/game. O: outdoors.

Q: Does one satisfied clause make the whole CNF true?

A: No. Every clause must hold.

Coffee indoors, with no music or game, gives $\text{True} \wedge \text{False} \wedge \text{True}$.

One true literal satisfies its clause. Every clause must pass.

Resolution: a conclusion in either case

A, B, C: Alice, Bob, and Carol attend.

We know $A \vee B$ and $\neg A \vee C$.

Q: Why must $B \vee C$ hold without knowing A?

A: If A=False, the first clause requires B=True.

If A=True, the second clause requires C=True.

Both cases imply $B \vee C$. This is the resolution inference.

If B=C=False, the premises would require both A and $\neg A$.

A complete resolution refutation

$KB = \{A, A \rightarrow B, B \rightarrow C\}$. Goal: prove $KB \models C$.

CNF: 1. A 2. $\neg A \vee B$ 3. $\neg B \vee C$

4. $\neg C$ (temporarily negate the goal)

5. B (from 1 and 2)

6. C (from 3 and 5)

Q: What follows from clauses 4 and 6?

A: 7. (from 4 and 6). No countermodel exists, so $KB \models C$.

Recap: SAT search with DPLL

DPLL searches for an assignment satisfying every clause.

1. Remove satisfied clauses and false literals.

2. Empty clause: the current branch fails.

 No clauses left: a satisfying assignment exists.

3. Unit clause: its literal is forced in this branch.

 Pure literal: choose the value that makes it true.

4. Branch if needed. If it fails, undo it and try the other value.

After each new assignment, simplify again.

Simplification and unit propagation

$(P \vee Q) \wedge (\neg P \vee R) \wedge (\neg R \vee S)$, with $P=\text{True}$.

Remove $P \vee Q$. Remove the false literal $\neg P$.

Remaining formula: $R \wedge (\neg R \vee S)$.

Q: The clause R has one literal. What happens next?

A: Set $R=\text{True}$. Then $\neg R \vee S$ becomes the unit clause S .

Set $S=\text{True}$. No clauses remain, so this branch succeeds.

$P=R=S=\text{True}$ satisfies the formula. Q may take either value.

Unit propagation and pure literals

Unit example: $(P) \wedge (\neg P \vee Q)$. $P=\text{True}$ forces $Q=\text{True}$.

Pure example: $(U \vee V) \wedge (\neg V \vee W)$. U occurs only positively.

Choosing $U=\text{True}$ preserves satisfiability.

Q: Could U be False? Which operation finds a forced value?

A: Yes. $U=\text{False}$, $V=\text{True}$, $W=\text{True}$ satisfies both clauses.

Unit propagation finds a value required in the current branch.

A pure-literal choice need not hold in every model.

DPLL backtracking on a small formula

$$F = (X \vee Y) \wedge (X \vee \neg Y) \wedge (\neg X \vee Y)$$

No unit clauses. Neither variable is pure.

Try $X=\text{False}$: the remaining clauses are Y and $\neg Y$. This fails.

Undo the choice. Try $X=\text{True}$: only the unit clause Y remains.

Set $Y=\text{True}$. All clauses are satisfied, so F is SAT.

Q: Why does the first contradiction not prove UNSAT?

A: It rules out only $X=\text{False}$. $X=\text{True}$, $Y=\text{True}$ is a model.

Testing an encoding against allowed states

Requirement: exactly one of two lights is on.

At least one: $U \vee V$. At most one: $\neg(U \wedge V)$.

CNF: $(U \vee V) \wedge (\neg U \vee \neg V)$

Q: Which unwanted state does each condition exclude?

A: $U \vee V$ excludes (False, False), where both lights are off.

$\neg(U \wedge V)$ excludes (True, True), where both lights are on.

Only (True, False) and (False, True) remain.

CNF formulas as lists of integer literals

Variable IDs: P is 1, Q is 2, and R is 3.

Positive integers name atoms. Negative integers negate them.

$(P \vee \neg Q) \wedge R$ becomes $[[1, -2], [3]]$.

Each inner list is an OR clause. The outer list joins them by AND.

Q: How do the formulas $[[3]]$, $[[[]]]$, and $[]$ differ?

A: $[[3]]$ forces R. $[[[]]]$ has an empty clause and is UNSAT.

$[]$ has no clauses. It has no unmet requirements and is SAT.

Recap: objects, predicates, and quantifiers

Domain D: the nonempty set of objects we talk about.

A constant names an object. A variable ranges over objects.

Robot(x): a property. Near(x, y): a relation.

$\forall x$: for every object. $\exists x$: for at least one object.

$x = y$: x and y refer to the same object.

$\exists x \text{ Robot}(x)$ means that at least one object is a robot.

Quantifiers range over the whole domain.

Quantifiers range over the whole domain

$D = \{r_1, r_2, b\}$. The robots r_1 and r_2 are powered. b is a box.

1. $\forall x (\text{Robot}(x) \rightarrow \text{Powered}(x))$: every robot is powered.
2. $\forall x (\text{Robot}(x) \wedge \text{Powered}(x))$: everything is a powered robot.

Q: What extra claim does formula 2 make about the box?

A: It requires $\text{Robot}(b)$ and $\text{Powered}(b)$. $\text{Robot}(b)$ is false.

Formula 2 therefore fails, regardless of the box's power.

Formula 1 imposes no power requirement on the non-robot box.

Building a formula with two object types

“Some robot uses every tool.” Domain: robots and tools.

1. Choose an object: $\exists x \dots$

2. Require a robot: $\exists x (\text{Robot}(x) \wedge \dots)$

3. For each tool: $\forall y (\text{Tool}(y) \rightarrow \text{Uses}(x, y))$

Together: $\exists x (\text{Robot}(x) \wedge \forall y (\text{Tool}(y) \rightarrow \text{Uses}(x, y)))$

Q: Why AND for x, but implication for y?

A: x must be a robot. The rule for y applies only to tools.

“Only if” and the direction of implication

“The door opens only if a valid card is present.”

O: door opens. K: valid card is present. Rule: $O \rightarrow K$.

The only forbidden case is $O=\text{True}$, $K=\text{False}$.

Q: Does a valid card guarantee that the door opens?

A: No. $O=\text{False}$, $K=\text{True}$ still satisfies the rule.

A card is necessary. The rule does not make it sufficient.

“The door opens if a card is present” would mean $K \rightarrow O$.

Quantifier order and a shared witness

Only these pairs hold: Ava admires Bo, Bo admires Cy, and Cy admires Ava.

1. $\forall x \exists y \text{Admires}(x, y)$ 2. $\exists y \forall x \text{Admires}(x, y)$

Q: Which formula is true? What witnesses make it true?

A: Formula 1 is true. Choose Bo for Ava, Cy for Bo, Ava for Cy.

Formula 2 is false. No single person receives everyone's admiration.

In 1, y can depend on x . In 2, the same y must work for all x .

Different variables can name the same object

$D = \{\text{Nori}\}$. Nori is a cat.

$\exists x \exists y (\text{Cat}(x) \wedge \text{Cat}(y))$ is true with $x=y=\text{Nori}$.

Q: What condition forces x and y to refer to different cats?

A: Add $x \neq y$: $\exists x \exists y (\text{Cat}(x) \wedge \text{Cat}(y) \wedge x \neq y)$.

This requires at least two cats. Additional cats remain allowed.

At least one allows 1 or more. At most one allows 0 or 1.

Exactly one requires existence and a limit of one object.

Exactly one: existence and uniqueness

Domain: lamps. $\text{Lit}(x)$ means that lamp x is on.

“Exactly one lamp is lit” requires existence and uniqueness.

Existence: $\exists x \text{ Lit}(x)$

At most one: $\forall x \forall y ((\text{Lit}(x) \wedge \text{Lit}(y)) \rightarrow x = y)$

Together: $\exists x (\text{Lit}(x) \wedge \forall y (\text{Lit}(y) \rightarrow y = x))$

Q: Why is the existence requirement necessary?

A: With zero lit lamps, “at most one” holds but “exactly one” fails.

Negating a quantified statement

Two of three packages arrived. $P(x)$ means “ x arrived.”

$\neg \forall x P(x) \equiv \exists x \neg P(x)$: at least one did not arrive.

$\neg \exists x P(x) \equiv \forall x \neg P(x)$: none arrived.

Q: How many missing packages refute “every package arrived”?

A: One is enough. That package witnesses $\exists x \neg P(x)$.

Here, “not every package arrived” is true.

“No package arrived” is false because two arrived.

A checkable countermodel

Claim 1: every cat is an animal. Claim 2: every animal is a cat.

$D = \{a, b\}$. $Cat = \{a\}$. $Animal = \{a, b\}$.

Claim 1 is true. Claim 2 is false: b is an animal but not a cat.

Q: What makes a countermodel checkable?

A: Specify the domain and the predicate interpretations.

For relations, give ordered pairs. Interpret any constants too.

Evaluate both claims in the same world and show the difference.

A countermodel with an explicit relation

Claim: "R sees every beacon." Proposed formula:

$$\forall y (\text{Beacon}(y) \rightarrow \text{Sees}(y, R))$$

$D = \{r, b_1, b_2\}$. R names r. Beacon = $\{b_1, b_2\}$.

Sees = $\{(r, b_1), (r, b_2)\}$. All other pairs are false.

Q: Does this formula correctly translate the claim?

A: No. In this world, the claim is true and the formula is false.

For $y=b_1$, Sees(b_1 , R) is false. Argument order matters.