CS221 Problem Workout

Week 9

Agenda

- **Syntax and semantics**
- **Inference Rules**
- **First Order Logic**
- **Modus Ponens**
- **Additional Topics**

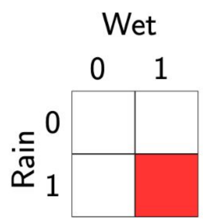
Ingredients of a Logic

Syntax: defines a set of valid **formulas** (Formulas)

Example: Rain ∧ Wet

Semantics: for each formula, specify a set of **models** (assignments / configurations of the world)

Example:



Inference rules: given f, what new formulas g can be added that are guaranteed to follow ($\frac{f}{g}$)?

Example: from Rain ∧ Wet, derive Rain

Syntax of Propositional Logic

Propositional symbols (atomic formulas): A, B, C

Logical connectives: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

Build up formulas recursively—if f and g are formulas, so are the following:

- Negation: $\neg f$
- Conjunction: $f \wedge g$
- Disjunction: $f \vee g$
- Implication: $f \rightarrow g$
- Biconditional: $f \leftrightarrow g$

Implication and Causality

Implication in propositional logic may express causality but not always:

Example 1: The Photosynthesis formula below expresses cause and effect.

$$Carbon\ dioxide + Water \rightarrow Glucose + Oxygen$$

Example 2: the following proposition does not express causality:

Raining → Doing well on the AI final

Properties/laws of Propositional Logic

1. **Identity law**:

   $p \wedge True \equiv p$

   $p \vee False \equiv p$

2. **Domination law**:

   $p \vee True \equiv True$

   $p \wedge False \equiv False$

3. **Idempotence law**:

   $p \vee p \equiv p$

   $p \wedge p \equiv p$

4. **Negation law**:

   $p \wedge (\neg p) \equiv False$

   $p \vee (\neg p) \equiv True$

5. **Double negation law**:

   $\neg\neg p \equiv p$

6. **Commutativity law**:

   $p \wedge q \equiv q \wedge p$

   $p \vee q \equiv q \vee p$

7. **Associativity law**:

   $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$

   $(p \vee q) \vee r \equiv p \vee (q \vee r)$

8. **Distributivity law**:

   $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$

   $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

   $p \vee (q \vee r) \equiv (p \vee q) \vee (p \vee r)$

   $p \wedge (q \wedge r) \equiv (p \wedge q) \wedge (p \wedge r)$

9. **Absorption law**:

   $p \vee (p \wedge q) \equiv p$

   $p \wedge (p \vee q) \equiv p$

10. **DeMorgan's law**:

    $\neg(p \wedge q) \equiv (\neg p) \vee (\neg q)$

    $\neg(p \vee q) \equiv (\neg p) \wedge (\neg q)$

11. **Implication to disjunction law**:

    $p \rightarrow q \equiv \neg p \vee q$

12. **IFF to implication law**:

    $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Conjunctive Normal Form

- **Clause**: disjunction of literals
  - … OR … OR …
- **CNF formula**: a conjunction of clauses
  - (… OR … OR …) AND (…OR …)…
- Every propositional formula can be converted to an equivalent CNF formula
- CNF is useful for resolution

Conversion rules:

- Eliminate $\leftrightarrow$: $\frac{f \leftrightarrow g}{(f \rightarrow g) \wedge (g \rightarrow f)}$
- Eliminate $\rightarrow$: $\frac{f \rightarrow g}{\neg f \vee g}$
- Move $\neg$ inwards: $\frac{\neg (f \wedge g)}{\neg f \vee \neg g}$
- Move $\neg$ inwards: $\frac{\neg (f \vee g)}{\neg f \wedge \neg g}$
- Eliminate double negation: $\frac{\neg \neg f}{f}$
- Distribute $\vee$ over $\wedge$: $\frac{f \vee (g \wedge h)}{(f \vee g) \wedge (f \vee h)}$

Problem 1

1) [CA session] Problem 1

Compute the conjunctive normal form (CNF) of the following two formulas and write every step of your computation:

(a) $\neg P \to \neg\neg(Q \vee (R \wedge \neg S))$

(b) $(P \to (Q \vee (R \wedge S))) \wedge (R \vee (S \to Q))$

Logical Inference and Modus Ponens

An inference in Propositional Logic is a sequence of propositions denoted as:

$$\frac{p_1 \; p_2 \cdots p_n}{q}$$

Definition: Modus ponens inference rule

For any propositional symbols p and q:

$$\frac{p, \quad p \to q}{q}$$

Example:

It is raining. (Rain)

If it is raining, then it is wet. (Rain $\to$ Wet)

Therefore, it is wet. (Wet)

$$\frac{\text{Rain}, \quad \text{Rain} \to \text{Wet}}{\text{Wet}} \qquad \frac{\text{(premises)}}{\text{(conclusion)}}$$

Resolution

Definition: resolution inference rule

$$\frac{f_1 \vee \cdots \vee f_n \vee p, \quad \neg p \vee g_1 \vee \cdots \vee g_m}{f_1 \vee \cdots \vee f_n \vee g_1 \vee \cdots \vee g_m}$$

Example:

$$\frac{\text{Rain} \vee \text{Snow}, \quad \neg\text{Snow} \vee \text{Traffic}}{\text{Rain} \vee \text{Traffic}}$$

First-Order Logic

- The expressive power of Propositional Logic is limited. For example, it cannot express expressions such as “for all” or “for some”. It is also difficult to express relationships.
- First-order logic, also known as predicate logic, combines quantifiers and predicates for a more powerful and compact formalism.

Terms (refer to objects):

- Constant symbol (e.g., arithmetic)
- Variable (e.g., x)
- Function of terms (e.g., $\text{Sum}(3, x)$)

Formulas (refer to truth values):

- Atomic formulas (atoms): predicate applied to terms (e.g., $\text{Knows}(x, \text{arithmetic})$)
- Connectives applied to formulas (e.g., $\text{Student}(x) \rightarrow \text{Knows}(x, \text{arithmetic})$)
- Quantifiers applied to formulas (e.g., $\forall x\, \text{Student}(x) \rightarrow \text{Knows}(x, \text{arithmetic})$)

Qualifiers

- **Universal quantifier**: denoted with the symbol $\forall$, expresses the statements: for all, for every, all of, for each, for any, any of, given any, for an arbitrary, etc.
  $\forall x\, P(x)$ asserts that the property/predicate $P(x)$ is true for every x in the domain.
- **Existential quantifier**: The existential quantifier, denoted with the symbol $\exists$, expresses the statements: there exist, for some, for at least one, there is, there is at least one, etc.
  $\exists x\, P(x)$ asserts that the property/predicate $P(x)$ is true for some element x in the domain.

Qualifiers

- **Universal quantifier**: denoted with the symbol $\forall$, expresses the statements: for all, for every, all of, for each, for any, any of, given any, for an arbitrary, etc.
  $\forall x\, P(x)$ asserts that the property/predicate $P(x)$ is true for every x in the domain.
- **Existential quantifier**: The existential quantifier, denoted with the symbol $\exists$, expresses the statements: there exist, for some, for at least one, there is, there is at least one, etc.
  $\exists x\, P(x)$ asserts that the property/predicate $P(x)$ is true for some element x in the domain.

Universal quantification ($\forall$):

Every student knows arithmetic.

$\forall x\, \text{Student}(x) \rightarrow \text{Knows}(x, \text{arithmetic})$

Existential quantification ($\exists$):

Some student knows arithmetic.

$\exists x\, \text{Student}(x) \wedge \text{Knows}(x, \text{arithmetic})$

First-Order Logic Examples

1. Everyone loves everyone.
   $\forall x \; \forall y \; love(x, y)$

First-Order Logic Examples

1. Everyone loves everyone.
   $\forall x \ \forall y \ love(x, y)$

2. If anyone cheats, everyone suffers.
   $\forall x \ (cheat(x) \rightarrow \forall y \ suffer(y))$

   wrong answer: $\forall y (\forall x \ cheat(x) \rightarrow suffer(y))$ (Order matters!)

First-Order Logic Examples

1. Everyone loves everyone.
   $\forall x\ \forall y$ love (x, y)

2. If anyone cheats, everyone suffers.
   $\forall x\ (\text{cheat}(x) \rightarrow \forall y\ \text{suffer}(y))$

   wrong answer: $\forall y (\forall x\ \text{cheat}(x) \rightarrow \text{suffer}(y))$ (Order matters!)
   This is one way of saying "If everyone cheats, then everyone suffers."

First-Order Logic Examples

1. Everyone loves everyone.
   $\forall x \ \forall y$ love (x, y)

2. If anyone cheats, everyone suffers.
   $\forall x$ (cheat(x) $\rightarrow$ $\forall y$ suffer(y))

3. Every startup that has a good product will have customers
   $\forall x$ ((startup(x) $\wedge$ good_product(x)) $\rightarrow$ has_customers(x)))

Problem 3

3) [CA Session] Problem 3

Translate the following English sentences into first-order logic formulas:

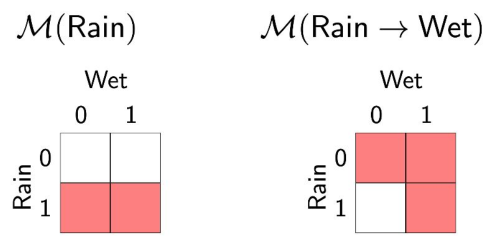
(a) Every student takes at least one course.

(b) Every student who takes Analysis also takes Geometry.

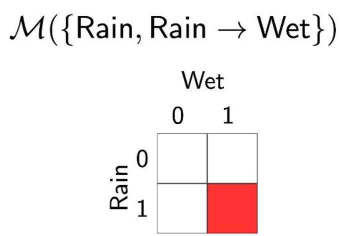
(c) No student failed Chemistry but at least one student failed History.

Quick Recap

- Knowledge base: set of formulae
- Model: an assignment to the world
- M(f): set of all satisfying models
- M(KB): models satisfy each formula in KB



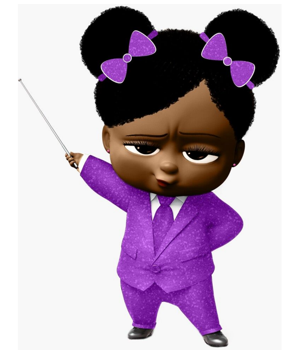
Intersection:



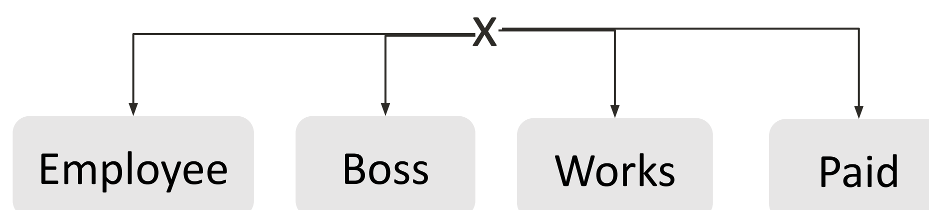
Office office - Problem Sheet P5

Alice

Bob



Carol



Office office - Problem Sheet P5

KB

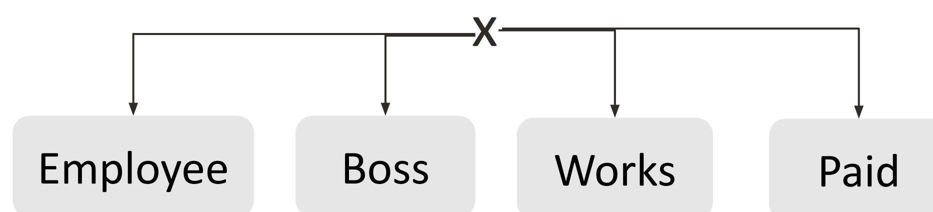
Boss(Carol)
Employee(Bob)
Paid(Carol) ∧ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) ⟷ ¬ Boss(x)
$\forall$ x Employee(x) ⟶ Works(x)
$\forall$ x Paid(x) ∧ ¬ Works(x) ⟶ Boss(x)


Alice


Bob


Carol



First Order to Propositional logic

$$\forall x\ (\text{Employee(x)} \rightarrow \text{Works(x)})$$

$$x \in \{\text{Alice}, \text{Bob}, \text{Carol}\}$$

$$(\text{Employee}(\text{Alice}) \rightarrow \text{Works}(\text{Alice})) \wedge$$

$$(\text{Empolyee}(\text{Bob}) \rightarrow \text{Works}(\text{Bob})) \wedge$$

$$(\text{Employee}(\text{Carol}) \rightarrow \text{Works}(\text{Carol}))$$

Adding a new fact to KB

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) ∧ Works(Carol)
Paid(Alice)

∀ x Employee(x) ⟷ ¬ Boss(x)
∀ x Employee(x) ⟶ Works(x)
∀ x Paid(x) ∧ ¬ Works(x) ⟶ Boss(x)

S = Anyone who is not a boss either works or does not get paid

Is M(KB) different from M(KB U S)?

$$F = \forall x \; \neg Boss(x) \rightarrow (Works(x) \vee \neg Paid(x))$$

$$\forall x \; Boss(x) \vee Works(x) \vee \neg Paid(x)$$

$$\forall x \; (Boss(x) \vee Employee(x)) \wedge (\neg Employee(x) \vee \neg Boss(x))$$

Adding a new fact to KB

$$\forall x \ Boss(x) \lor Works(x) \lor \neg Paid(x)$$

$$\forall x \ (Boss(x) \lor Employee(x)) \land (\neg Employee(x) \lor \neg Boss(x))$$

Case 1. If Boss(x) == T, S is satisfied for x

Case 2. If Boss(x) == F, Employee(x) must be T
Since Employee(x) -> Works(x)
S is again satisfied

Thus,
M(KB) $\subseteq$ M(KB U S)

By defn,
M(KB) $\supseteq$ M(KB U S)

Thus,
M(KB) = M(KB U S)

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) $\wedge$ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) $\leftrightarrow \neg$ Boss(x)
$\forall$ x Employee(x) $\longrightarrow$ Works(x)
$\forall$ x Paid(x) $\wedge \neg$ Works(x) $\longrightarrow$ Boss(x)

Does everyone work?

Not Everyone works

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) $\land$ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) $\leftrightarrow \neg$ Boss(x)
$\forall$ x Employee(x) $\longrightarrow$ Works(x)
$\forall$ x Paid(x) $\land \neg$ Works(x) $\longrightarrow$ Boss(x)

Does everyone work?

Not Everyone works

x	Employee	Boss	Works	Paid
Alice				
Bob				
Carol				

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) $\wedge$ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) $\leftrightarrow \neg$ Boss(x)
$\forall$ x Employee(x) $\longrightarrow$ Works(x)
$\forall$ x Paid(x) $\wedge \neg$ Works(x) $\longrightarrow$ Boss(x)

Does everyone work?

Not Everyone works

x	Employee	Boss	Works	Paid
Alice				T
Bob	T			
Carol		T	T	T

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) $\wedge$ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) $\leftrightarrow$ ¬ Boss(x)
$\forall$ x Employee(x) $\longrightarrow$ Works(x)
$\forall$ x Paid(x) $\wedge$ ¬ Works(x) $\longrightarrow$ Boss(x)

Does everyone work?

Not Everyone works

x	Employee	Boss	Works	Paid
Alice				T
Bob	T	F	T	
Carol	F	T	T	T

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) ∧ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) ⟷ ¬ Boss(x)
$\forall$ x Employee(x) ⟶ Works(x)
$\forall$ x Paid(x) ∧ ¬ Works(x) ⟶ Boss(x)

Does everyone work?

Not Everyone works

x	Employee	Boss	Works	Paid
Alice			F	T
Bob	T	F	T	
Carol	F	T	T	T

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) ∧ Works(Carol)
Paid(Alice)

$\forall$ x Employee(x) ⟷ ¬ Boss(x)
$\forall$ x Employee(x) ⟶ Works(x)
$\forall$ x Paid(x) ∧ ¬ Works(x) ⟶ Boss(x)

Does everyone work?

Not Everyone works

x	Employee	Boss	Works	Paid
Alice		T	F	T
Bob	T	F	T	F or T
Carol	F	T	T	T

Fact checking

KB

Boss(Carol)
Employee(Bob)
Paid(Carol) ⋀ Works(Carol)
Paid(Alice)

∀ x Employee(x) ⟷ ¬ Boss(x)
∀ x Employee(x) ⟶ Works(x)
∀ x Paid(x) ⋀ ¬ Works(x) ⟶ Boss(x)

Does everyone work?

Not Everyone works

x	Employee	Boss	Works	Paid
Alice	F	T	F	T
Bob	T	F	T	F or T
Carol	F	T	T	T

Entailment, Contingency and Contradiction

- We can examine a new formula *f* against *KB* by looking at M(f) ∩ M(KB)

$M(f) \cap M(KB) = M(KB)$	Intersection is *M(KB)*	*f* is **entailed** by *KB*	Already knew the info
$\emptyset \subsetneq M(f) \cap M(KB) \subsetneq M(KB)$	Intersection is smaller than *M(KB)*, but nonempty	*f* is **contingent** to *KB*	Learned new info
$M(f) \cap M(KB) = \emptyset$	Intersection is empty	*f* **contradicts** *KB*	Info contradicts what we know

Problem 2

2) [CA session] Problem 2: Proof by Resolution

In this question we practice proving by resolution on the following knowledge base: Either Heather attended the meeting or Heather was not invited. If the boss wanted Heather at the meeting, then she was invited. Heather did not attend the meeting. If the boss did not want Heather there, and the boss did not invite her there, then she is going to be fired. Prove Heather is going to be fired.

Proposition: contradiction and entailment

KB **contradicts** f iff KB **entails** $\neg f$.

Thank You