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 $\land$ Wet

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

Example:

```
<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

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

Example: from Rain $\land$ Wet, derive Rain
Syntax of Propositional Logic

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

Logical connectives: $\neg, \land, \lor, \to, \leftrightarrow$

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

- Negation: $\neg f$
- Conjunction: $f \land g$
- Disjunction: $f \lor g$
- Implication: $f \to g$
- 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.

\[ \text{Carbon dioxide} + \text{Water} \rightarrow \text{Glucose} + \text{Oxygen} \]

Example 2: the following proposition does not express causality:

Raining $\rightarrow$ Doing well on the AI final
Properties/laws of Propositional Logic

1. **Identity law:**
   
   \[ p \land \text{True} \equiv p \]
   
   \[ p \lor \text{False} \equiv p \]

2. **Domination law:**
   
   \[ p \lor \text{True} \equiv \text{True} \]
   
   \[ p \land \text{False} \equiv \text{False} \]

3. **Idempotence law:**
   
   \[ p \lor p \equiv p \]
   
   \[ p \land p \equiv p \]

4. **Negation law:**
   
   \[ p \land (\neg p) \equiv \text{False} \]
   
   \[ p \lor (\neg p) \equiv \text{True} \]

5. **Double negation law:**
   
   \[ \neg \neg p \equiv p \]

6. **Commutativity law:**
   
   \[ p \land q \equiv q \land p \]
   
   \[ p \lor q \equiv q \lor p \]

7. **Associativity law:**
   
   \[ (p \land q) \land r \equiv p \land (q \land r) \]
   
   \[ (p \lor q) \lor r \equiv p \lor (q \lor r) \]

8. **Distributivity law:**
   
   \[ p \land (q \lor r) \equiv (p \land q) \lor (p \land r) \]
   
   \[ p \lor (q \land r) \equiv (p \lor q) \land (p \lor r) \]

9. **Absorption law:**
   
   \[ p \lor (p \land q) \equiv p \]
   
   \[ p \land (p \lor q) \equiv p \]

10. **DeMorgan’s law:**
    
    \[ \neg (p \land q) \equiv (\neg p) \lor (\neg q) \]
    
    \[ \neg (p \lor q) \equiv (\neg p) \land (\neg q) \]

11. **Implication to disjunction law:**
    
    \[ p \rightarrow q \equiv \neg p \lor q \]

12. **IFF to implication law:**
    
    \[ p \leftrightarrow q \equiv (p \rightarrow q) \land (q \rightarrow p) \]
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 $\leftrightarrow$: $\frac{f \leftrightarrow g}{(f \rightarrow g) \land (g \rightarrow f)}$
- Eliminate $\rightarrow$: $\frac{f \rightarrow g}{\neg f \lor g}$
- Move $\neg$ inwards: $\frac{\neg(f \land g)}{\neg f \lor \neg g}$
- Move $\neg$ inwards: $\frac{\neg(f \lor g)}{\neg f \land \neg g}$
- Eliminate double negation: $\frac{\neg\neg f}{f}$
- Distribute $\lor$ over $\land$: $\frac{f \lor (g \land h)}{(f \lor g) \land (f \lor h)}$
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) $\neg P \rightarrow \neg \neg (Q \lor (R \land \neg S))$

(b) $(P \rightarrow (Q \lor (R \land S'))) \land (R \lor (S \rightarrow Q))$
Logical Inference and Modus Ponens

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

\[
\begin{array}{c}
p_1 \ p_2 \cdots p_n \\
\hline
q
\end{array}
\]

**Definition: Modus ponens inference rule**

For any propositional symbols \( p \) and \( q \):

\[
\begin{array}{c}
p, \ p \rightarrow q \\
\hline
q
\end{array}
\]

**Example:**

It is raining. (Rain)

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

Therefore, it is wet. (Wet)

\[
\begin{array}{c}
\text{Rain,} \quad \text{Rain} \rightarrow \text{Wet} \\
\hline
\text{Wet}
\end{array}
\]

(premises) (conclusion)
Resolution

**Definition: resolution inference rule**

\[
\begin{align*}
f_1 \lor \cdots \lor f_n \lor p, & \quad \neg p \lor g_1 \lor \cdots \lor g_m \\
\hline
f_1 \lor \cdots \lor f_n \lor g_1 \lor \cdots \lor g_m
\end{align*}
\]

**Example:**

Rain \lor \text{Snow}, \quad \neg \text{Snow} \lor \text{Traffic}

\[
\text{Rain} \lor \text{Traffic}
\]
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.*

$\exists x \text{ Student}(x) \land \text{Knows}(x, \text{arithmetic})$
First-Order Logic Examples

1. Everyone loves everyone.
   \[ \forall x \ \forall y \ \text{love} \ (x, y) \]
First-Order Logic Examples

1. Everyone loves everyone.
   \( \forall x \forall y \text{love}(x, y) \)

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!)
First-Order Logic Examples

1. Everyone loves everyone.
   \( \forall x \; \forall y \; \text{love}(x, y) \)

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.
   \( \forall x \ \forall y \ \text{love}(x, y) \)

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

3. Every startup that has a good product will have customers
   \( \forall x \ ((\text{startup}(x) \land \text{good\_product}(x)) \rightarrow \text{has\_customers}(x)) \)
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

![Diagram]

Intersection:
Office office - Problem Sheet P5

Alice  Bob  Carol

Employee  Boss  Works  Paid
Office office - Problem Sheet P5

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)
First Order to Propositional logic

\[ \forall x \ (\text{Employee}(x) \rightarrow \text{Works}(x)) \]

\[ x \in \{\text{Alice}, \text{Bob}, \text{Carol}\} \]

\[ (\text{Employee}(\text{Alice}) \rightarrow \text{Works}(\text{Alice})) \land \]

\[ (\text{Employee}(\text{Bob}) \rightarrow \text{Works}(\text{Bob})) \land \]

\[ (\text{Employee}(\text{Carol}) \rightarrow \text{Works}(\text{Carol})) \]
Adding a new fact to KB

KB

\[
\begin{align*}
&\text{Boss}(\text{Carol}) \\
&\text{Employee}(\text{Bob}) \\
&\text{Paid}(\text{Carol}) \land \text{Works}(\text{Carol}) \\
&\text{Paid}(\text{Alice})
\end{align*}
\]

\[
\begin{align*}
\forall x \times \text{Employee}(x) &\leftrightarrow \neg \text{Boss}(x) \\
\forall x \times \text{Employee}(x) &\rightarrow \text{Works}(x) \\
\forall x \times \text{Paid}(x) \land \neg \text{Works}(x) &\rightarrow \text{Boss}(x)
\end{align*}
\]

\[
S = \text{Anyone who is not a boss either works or does not get paid}
\]

Is $M(KB)$ different from $M(KB \cup S)$?

\[
\begin{align*}
F &= \forall x \neg \text{Boss}(x) \rightarrow (\text{Works}(x) \lor \neg \text{Paid}(x)) \\
&\lor x \neg \text{Boss}(x) \lor \text{Works}(x) \lor \neg \text{Paid}(x) \\
&\forall x \left( \text{Boss}(x) \lor \text{Employee}(x) \right) \land \\
&\left( \neg \text{Employee}(x) \lor \neg \text{Boss}(x) \right)
\end{align*}
\]
Adding a new fact to KB

\((\neg \text{boss}(x) \lor \text{works}(x) \lor \neg \text{paid}(x))\)

Case 1. If \(\text{Boss}(x) == T\), \(S\) is satisfied for \(x\)

Case 2. If \(\text{Boss}(x) == F\), \(\text{Employee}(x)\) must be \(T\)
Since \(\text{Employee}(x) \rightarrow \text{Works}(x)\)
\(S\) 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) ∧ 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
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

<table>
<thead>
<tr>
<th>x</th>
<th>Employee</th>
<th>Boss</th>
<th>Works</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Fact checking

**KB**
- $\text{Boss}(\text{Carol})$
- $\text{Employee}(\text{Bob})$
- $\text{Paid}(\text{Carol}) \land \neg \text{Works}(\text{Carol})$
- $\text{Paid}(\text{Alice})$

$\forall x \left( \text{Employee}(x) \leftrightarrow \neg \text{Boss}(x) \right)$
$\forall x \left( \text{Employee}(x) \rightarrow \text{Works}(x) \right)$
$\forall x \left( \text{Paid}(x) \land \neg \text{Works}(x) \rightarrow \text{Boss}(x) \right)$

**Does everyone work?**

**Not Everyone works**

<table>
<thead>
<tr>
<th></th>
<th>Employee</th>
<th>Boss</th>
<th>Works</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>T</td>
<td></td>
<td></td>
<td>T</td>
</tr>
<tr>
<td>Bob</td>
<td>T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>x</th>
<th>Employee</th>
<th>Boss</th>
<th>Works</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td></td>
<td></td>
<td></td>
<td>T</td>
</tr>
<tr>
<td>Bob</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>Carol</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>x</th>
<th>Employee</th>
<th>Boss</th>
<th>Works</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>T</td>
<td></td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>Bob</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>Carol</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>x</th>
<th>Employee</th>
<th>Boss</th>
<th>Works</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td></td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>Bob</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F or T</td>
</tr>
<tr>
<td>Carol</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Fact checking

Does everyone work?

Not Everyone works

<table>
<thead>
<tr>
<th>x</th>
<th>Employee</th>
<th>Boss</th>
<th>Works</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>Bob</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F or T</td>
</tr>
<tr>
<td>Carol</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Entailment, Contingency and Contradiction

We can examine a new formula $f$ against $KB$ by looking at $M(f) \cap M(KB)$

<table>
<thead>
<tr>
<th>$M(f) \cap M(KB) = M(KB)$</th>
<th>Intersection is $M(KB)$</th>
<th>$f$ is entailed by $KB$</th>
<th>Already knew the info</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\emptyset \subsetneq M(f) \cap M(KB) \subsetneq M(KB)$</td>
<td>Intersection is smaller than $M(KB)$, but nonempty</td>
<td>$f$ is contingent to $KB$</td>
<td>Learned new info</td>
</tr>
<tr>
<td>$M(f) \cap M(KB) = \emptyset$</td>
<td>Intersection is empty</td>
<td>$f$ contradicts $KB$</td>
<td>Info contradicts what we know</td>
</tr>
</tbody>
</table>
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

\[ \text{KB contradicts } f \text{ iff KB entails } \neg f. \]
Thank You