Propositional Logic

Reading: Chapter 7.1, 7.3 – 7.5

[Partially based on slides from Chuck Dyer, Jerry Zhu, Louis Oliphant and Andrew Moore]
Logic

• If a problem domain can be represented formally, then a decision maker can use **logical reasoning** to make rational decisions

• Several types of logic:
  - Propositional Logic (Boolean logic)
  - First-Order Logic (aka first-order predicate calculus)
  - Non-Monotonic Logic
  - Markov Logic

• A logic includes:
  - **syntax**: what is a correctly-formed sentence?
  - **semantics**: what is the meaning of a sentence?
  - **Inference procedure** (reasoning, entailment): what sentence logically follows given knowledge?
Propositional Logic

- A symbol in PL is a symbolic variable whose value must be either True or False, and which stands for a natural language statement that could be either true or false.
  - A = “Smith has chest pain”
  - B = “Smith is depressed”
  - C = “It is raining”
Propositional Logic Syntax

Sentence → AtomicSentence | ComplexSentence
AtomicSentence → True | False | Symbol
Symbol → P | Q | R | . . .
ComplexSentence → ¬ Sentence

| ( Sentence ∧ Sentence )
| ( Sentence ∨ Sentence )
| ( Sentence ⇒ Sentence )
| ( Sentence ⇔ Sentence )

BNF (Backus-Naur Form) grammar for Propositional Logic

(((¬P ∨ ((True ∧ R) ⇔ Q)) ⇒ S) well formed (“wff” or “sentence”)
(¬(P ∨ Q) ∧ ⇒ S) not well formed
Propositional Logic Syntax

((¬P ∨ ((True ∧ R) ⇔ Q)) ⇒ S)

- ¬: Means "Not"
- ∨: Means "Or" -- disjunction
- ∧: Means "And" -- conjunction
- ⇔: Means "iff" -- biconditional
- ⇒: Means "if-then" implication
- ( and ): control the order of operations

Propositional symbols must be specified.

Means True

Means "Not"
Propositional Logic Syntax

- Precedence (from highest to lowest):
  \( \leftrightarrow, \land, \lor, \Rightarrow, \Leftrightarrow \)

- If the order is clear, you can leave off parentheses

\[ \leftrightarrow P \lor \text{True} \land R \leftrightarrow Q \Rightarrow S \quad \text{ok (though not recommended)} \]
\[ P \Rightarrow Q \Rightarrow S \quad \text{not ok} \]
Semantics

• An **interpretation** is a complete True / False assignment to all propositional symbols
  - Example symbols: P means “It is hot”, Q means “It is humid”, R means “It is raining”
  - There are 8 interpretations (TTT, ..., FFF)

• The semantics (meaning) of a sentence is the set of interpretations in which the sentence evaluates to True

• Example: the semantics of the sentence $P \lor Q$ is the set of 6 interpretations:
  - $P=True, Q=True, R=True$ or False
  - $P=True, Q=False, R=True$ or False
  - $P=False, Q=True, R=True$ or False

• A **model** of a set of sentences is an interpretation in which *all* the sentences are true
Evaluating a Sentence under an Interpretation

• Calculated using the definitions of all the connectives, recursively

<table>
<thead>
<tr>
<th>$P$</th>
<th>$Q$</th>
<th>$\neg P$</th>
<th>$P \land Q$</th>
<th>$P \lor Q$</th>
<th>$P \Rightarrow Q$</th>
<th>$P \Leftrightarrow Q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td></td>
<td></td>
<td></td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

• Pay attention to $\Rightarrow$
  ▪ “5 is even implies 6 is odd” is True!
  ▪ If $P$ is False, regardless of $Q$, $P \Rightarrow Q$ is True
  ▪ No causality needed: “5 is odd implies the Sun is a star” is True
Understanding “⇒”

- This is an operator. Although we call it “implies” or “implication,” do not try to understand its semantic form from the name. We could have called it “foo” instead and still defined its semantics the same way.
- A ⇒ B “means” A is *sufficient* but not *necessary* to make B true
- Example:
  - Let A be “has a cold” and B be “drink water”
  - A ⇒ B can be interpreted as “should drink water” when “has a cold.”
  - However, you can drink water even when you do not have a cold. Thus A ⇒ B is still true when A is *not* true.
Example

(←P ∨ (Q ∧ R)) ⇒ Q
Example

\( \leftarrow (P \lor (Q \land R)) \Rightarrow Q \)

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>R</th>
<th>~P</th>
<th>Q^R</th>
<th>~PvQ^R</th>
<th>~PvQ^R-&gt;Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Satisfiable: a sentence that is true under some interpretation(s)

Deciding satisfiability of a sentence is NP-complete
Example

$$((P \land R) \Rightarrow Q) \land P \land R \land \leftarrow Q$$
Example

$$((P \land R) \Rightarrow Q) \land P \land R \land \leftarrow Q$$

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>R</th>
<th>~Q</th>
<th>R^~Q</th>
<th>P^R^~Q</th>
<th>P^R</th>
<th>P^R-&gt;Q</th>
<th>final</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

** Unsatisfiable:** a sentence that is **false** under **all** interpretations

Also called **inconsistent** or a **contradiction**
Example

\((P \Rightarrow Q) \lor (P \land \leftarrow Q)\)
Example

\[(P \Rightarrow Q) \lor (P \land \leftarrow Q)\]

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>R</th>
<th>\sim Q</th>
<th>P\rightarrow Q</th>
<th>P\land \sim Q</th>
<th>\right( P\rightarrow Q \lor P\land \sim Q \left)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Valid**: a sentence that is **true** under **all** interpretations

Also called a **tautology**
Knowledge Base (KB)

- A knowledge base, KB, is a *set of sentences*
  Example KB:
  - \( \text{ChuckGivingLecture} \iff (\text{TodayIsMonday} \lor \text{TodayIsWednesday} \lor \text{TodayIsFriday}) \)
  - \( \neg \text{ChuckGivingLecture} \)

- It is equivalent to a *single* long sentence: the *conjunction* of all sentences
  - \( (\text{ChuckGivingLecture} \iff (\text{TodayIsMonday} \lor \text{TodayIsWednesday} \lor \text{TodayIsFriday})) \land \neg \text{ChuckGivingLecture} \)

- A *model* of a KB is an interpretation in which *all* sentences in KB are *true*
Entailment

- **Entailment** is the relation of a sentence $\beta$ **logically following** from other sentences $\alpha$ (e.g., KB)
  \[
  \alpha \models \beta
  \]

- $\alpha \models \beta$ if and only if, in every interpretation in which $\alpha$ is true, $\beta$ is also true; i.e., whenever $\alpha$ is true, so is $\beta$; all models of $\alpha$ and also models of $\beta$

- Deduction theorem: $\alpha \models \beta$ if and only if $\alpha \Rightarrow \beta$ is **valid** (always true)

- Proof by contradiction (refutation, *reductio ad absurandum*): $\alpha \models \beta$ if and only if $\alpha \land \neg \beta$ is **unsatisfiable**

- There are $2^n$ interpretations to check, if KB has $n$ symbols
Entailment

• Entailment is the relation of a sentence $\beta$ logically following from other sentences $\alpha$ (e.g., the KB)

$$\alpha \models \beta$$

• $\alpha \models \beta$ if and only if, in every interpretation in which $\alpha$ is true, $\beta$ is also true

All interpretations

\[
\begin{array}{c}
\text{\(\beta\) is true} \\
\quad \text{\(\alpha\) is true}
\end{array}
\]
Deductive Inference

• Say you write a program that, according to you, proves whether a sentence $\beta$ is entailed by $\alpha$
• The thing your program does is called *deductive inference*
• We don’t trust your inference program (yet), so we write things your program finds as
  \[ \alpha \vdash \beta \]
  It reads “$\beta$ is derived from $\alpha$ by your program”
• What properties should your program have?
  - **Soundness**: the inference algorithm only derives entailed sentences. That is, if $\alpha \vdash \beta$ then $\alpha \models \beta$
  - **Completeness**: all entailment can be inferred. That is, if $\alpha \models \beta$ then $\alpha \vdash \beta$
Soundness and Completeness

- **Soundness** says that any wff that follows deductively from a set of axioms, KB, is valid (i.e., true in all models).

- **Completeness** says that all valid sentences (i.e., true in *all* models of KB), can be proved from KB and hence are theorems.
Method 1: Inference by Enumeration

Also called **Model Checking** or **Truth Table Enumeration**

**LET:** \( KB = A \lor C, B \lor \neg C \quad \beta = A \lor B \)

**QUERY:** \( KB \models \beta? \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>| \beta \models \beta ? |</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

**NOTE:** The computer doesn't know the meaning of the proposition symbols

So, *all* logically distinct cases must be checked to prove that a sentence *can* be derived from KB
Inference by Enumeration

LET: \( KB = A \lor C, B \lor \neg C \quad \beta = A \lor B \)

QUERY: \( KB \models \beta \) ?

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>AVC</th>
<th>BV</th>
<th>\neg C</th>
<th>KB</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>\false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>\false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>false</td>
<td>\false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>false</td>
<td>\false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

Rows where all of sentences in \( KB \) are true are the models of \( KB \)
Inference by Enumeration

LET: \( KB = A \lor C, \ B \lor \neg C \)

\( \beta = A \lor B \)

QUERY: \( KB \models \beta \) ? \( YES! \)

\( \beta \) is entailed by \( KB \) if \( all \) models of \( KB \) are models of \( \beta \), i.e., \( all \) rows where \( KB \) is true, \( \beta \) is also true

In other words: \( KB \Rightarrow \beta \) is valid
Inference by Enumeration

• Using inference by enumeration to build a complete truth table in order to determine if a sentence is entailed by KB is a **complete** inference algorithm for Propositional Logic

• But very slow: takes exponential time
Method 2: Natural Deduction using Sound Inference Rules

Goal: Define a more efficient algorithm than enumeration that uses a set of inference rules to \textit{incrementally deduce new sentences} that are true given the initial set of sentences in KB, plus uses all logical equivalences.
Logical Equivalences

\[(\alpha \land \beta) \equiv (\beta \land \alpha) \quad \text{commutativity of } \land\]
\[(\alpha \lor \beta) \equiv (\beta \lor \alpha) \quad \text{commutativity of } \lor\]
\[((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma)) \quad \text{associativity of } \land\]
\[((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma)) \quad \text{associativity of } \lor\]
\[\neg(\neg \alpha) \equiv \alpha \quad \text{double-negation elimination}\]
\[(\alpha \Rightarrow \beta) \equiv (\neg \beta \Rightarrow \neg \alpha) \quad \text{contraposition}\]
\[(\alpha \Rightarrow \beta) \equiv (\neg \alpha \lor \beta) \quad \text{implication elimination}\]
\[(\alpha \iff \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination}\]
\[\neg(\alpha \land \beta) \equiv (\neg \alpha \lor \neg \beta) \quad \text{de Morgan}\]
\[\neg(\alpha \lor \beta) \equiv (\neg \alpha \land \neg \beta) \quad \text{de Morgan}\]
\[(\alpha \land (\beta \lor \gamma)) \equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) \quad \text{distributivity of } \land \text{ over } \lor\]
\[(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma)) \quad \text{distributivity of } \lor \text{ over } \land\]

You can use these equivalences to derive or modify sentences
Proof

• Series of inference steps that leads from $\alpha$ (or KB) to $\beta$
• This is exactly a search problem

KB:
1. TomGivingLecture $\iff$ (TodayIsTuesday $\lor$ TodayIsThursday)
2. $\neg$ TomGivingLecture

$\beta$:
   $\neg$ TodayIsTuesday
Proof

KB:
1. TomGivingLecture $\iff$ (TodayIsTuesday $\lor$ TodayIsThursday)
2. $\neg$ TomGivingLecture

3. TomGivingLecture $\Rightarrow$ (TodayIsTuesday $\lor$ TodayIsThursday) $\land$
   (TodayIsTuesday $\lor$ TodayIsThursday) $\Rightarrow$ TomGivingLecture
   biconditional-elimination to 1.
4. (TodayIsTuesday $\lor$ TodayIsThursday) $\Rightarrow$ TomGivingLecture
   and-elimination to 3.
5. $\neg$ TomGivingLecture $\Rightarrow$ $\neg$ (TodayIsTuesday $\lor$
   TodayIsThursday) contraposition to 4.
6. $\neg$ (TodayIsTuesday $\lor$ TodayIsThursday) Modus Ponens 2,5.
7. $\neg$ TodayIsTuesday $\land$ $\neg$ TodayIsThursday de Morgan to 6.
8. $\neg$ TodayIsTuesday and-elimination to 7.
Method 3: Resolution

- Your algorithm can use all the logical equivalences, *Modus Ponens*, and-elimination to derive new sentences.

- **Resolution**: a single inference rule
  - **Sound**: only derives entailed sentences
  - **Complete**: can derive any entailed sentence
    - Resolution is only refutation complete: if $\text{KB} \models \beta$, then $\text{KB} \land \neg \beta \vdash \text{empty}$. It cannot derive $\text{empty} \vdash (P \lor \neg P)$
  - But the sentences need to be preprocessed into a special form
  - But all sentences can be converted into this form