Knowledge
Representation and Reasoning

**Represent** knowledge about the world.
- Representation language.
- Knowledge base.
- Declarative - *facts* and *rules*.

**Reason** using that represented knowledge.
- Often *asking questions*.
- Inference procedure.
- Heavily dependent on *representation language*. 
Propositional Logic

Representation language and set of inference rules for reasoning about facts that are either **true** or **false**.

"that which is capable of being denied or affirmed as it is in itself"

Chrysippus of Soli, 3rd century BC
Knowledge Base

A list of *propositional logic* sentences that apply to the world.

For example:

\[
\begin{align*}
\text{Cold} \\
\neg \text{Raining} \\
(\text{Raining} \lor \text{Cloudy}) \\
\text{Cold} \iff \neg \text{Hot}
\end{align*}
\]

A knowledge base describes a set of worlds in which these facts and rules are true.
Knowledge Base

A *model* is a formalization of a “world”:
- Set the value of every variable in the KB to *True* or *False*.
- $2^n$ models possible for $n$ propositions.

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>False</td>
</tr>
<tr>
<td>Raining</td>
<td>False</td>
</tr>
<tr>
<td>Cloudy</td>
<td>False</td>
</tr>
<tr>
<td>Hot</td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>False</td>
</tr>
<tr>
<td>Cloudy</td>
<td>False</td>
</tr>
<tr>
<td>Hot</td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>True</td>
</tr>
<tr>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>True</td>
</tr>
</tbody>
</table>
Models and Sentences

Each sentence has a *truth value* in each model.

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>False</td>
</tr>
<tr>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>True</td>
</tr>
</tbody>
</table>

If sentence $a$ is true in model $m$, then $m$ **satisfies** (or **is a model of**) $a$.

$\text{Cold}$

$\neg \text{Raining}$

$(\text{Raining} \lor \text{Cloudy})$

$\text{Cold} \iff \neg \text{Hot}$

True

True

True

False
Models and Worlds

The KB specifies a subset of all possible models - those that satisfy all sentences in the KB.

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
<th>Proposition</th>
<th>Value</th>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>False</td>
<td>Cold</td>
<td>True</td>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>False</td>
<td>Raining</td>
<td>False</td>
<td>Raining</td>
<td>True</td>
</tr>
<tr>
<td>Cloudy</td>
<td>False</td>
<td>Cloudy</td>
<td>True</td>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>False</td>
<td>Hot</td>
<td>False</td>
<td>Hot</td>
<td>False</td>
</tr>
</tbody>
</table>

Each new piece of knowledge narrows down the set of possible models.

\[ \text{Cold} \iff \neg \text{Hot} \]

\[ \neg \text{Raining} \]

\[ (\text{Raining} \lor \text{Cloudy}) \]

\[ \text{Cold} \]
Summary

Knowledge Base

- Set of facts *asserted to be true* about the world.

Model

- Formalization of “the world”.
- An assignment to values to all variables.

Satisfaction

- Satisfies a sentence if that sentence is true in the model.
- Satisfies a KB if all sentences sure in model.
- Knowledge in the KB *narrows down* the set of possible world models.
Inference

So if we have a KB, then what?

Given:

\[
\begin{align*}
\text{Cold} \\
\neg \text{Raining} \\
(\text{Raining} \lor \text{Cloudy}) \\
\text{Cold} &\iff \neg \text{Hot}
\end{align*}
\]

We’d like to ask it questions.

... we can ask:  \text{Hot}?

**Inference:** process of deriving new facts from given facts.
Inference (Formally)

KB A **entails** sentence B

if and only if:

- every model which satisfies A, satisfies B.

In other words: if A is true then B **must be true**.

*Only conclusions you can make about the true world.*

Most frequent form of inference: $KB \models Q$

*That’s nice, but how do we compute?*
Logical Inference

Take a KB, and produce new sentences of knowledge.

Inference algorithms: methods for finding a proof of $Q$ using a set of inference rules.

Desirable properties:
- Don’t make any mistakes
- Be able to prove all possible true statements
Inference (formally)

Could just enumerate worlds …

Knowledge Base

Query Sentence

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>False</td>
</tr>
<tr>
<td>Raining</td>
<td>False</td>
</tr>
<tr>
<td>Cloudy</td>
<td>False</td>
</tr>
<tr>
<td>Hot</td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>True</td>
</tr>
<tr>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>True</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>False</td>
</tr>
<tr>
<td>Cloudy</td>
<td>False</td>
</tr>
<tr>
<td>Hot</td>
<td>False</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>True</td>
</tr>
<tr>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>True</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>True</td>
</tr>
<tr>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>True</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Proposition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cold</td>
<td>True</td>
</tr>
<tr>
<td>Raining</td>
<td>True</td>
</tr>
<tr>
<td>Cloudy</td>
<td>True</td>
</tr>
<tr>
<td>Hot</td>
<td>True</td>
</tr>
</tbody>
</table>

OK

Not OK
Inference Rules

Often written in form:

Start with $A \lor B, \neg B$

Given this knowledge can infer this $A$

can infer this
For example, given KB:

\[\begin{align*}
\text{Cold} \\
\lnot \text{Raining} \\
(\text{Raining} \lor \text{Cloudy}) \\
\text{Cold} \iff \lnot \text{Hot}
\end{align*}\]

We ask:

\[\text{Hot?}\]

**Inference:**

\[\begin{align*}
\text{Cold} &= \text{True} \\
\text{True} &\iff \lnot \text{Hot} \\
\lnot \text{Hot} &= \text{True} \\
\text{Hot} &= \text{False}
\end{align*}\]
Inference ...

We want to *start* somewhere (KB).
We’d like to *apply* some *rules*.
But there are lots of ways we *might* go.
… in order to reach some *goal* (sentence).

Does that sound familiar?

**Inference as search:**

<table>
<thead>
<tr>
<th>Set of states</th>
<th>True sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start state</td>
<td>KB</td>
</tr>
<tr>
<td>Set of actions and action rules</td>
<td>Inference rules</td>
</tr>
<tr>
<td>Goal test</td>
<td>Q in sentences?</td>
</tr>
<tr>
<td>Cost function</td>
<td>1 per rule</td>
</tr>
</tbody>
</table>
Resolution

The following inference rule is **both sound and complete**:

\[
\begin{align*}
  a_1 \lor \ldots \lor a_{i-1} \lor c \lor a_{i+1} \lor \ldots \lor a_n, & \quad b_1 \lor \ldots \lor b_{j-1} \lor \neg c \lor b_{j+1} \lor \ldots \lor b_m \\
  \hline
  a_1 \lor \ldots \lor a_{i-1} \lor a_{i+1} \lor \ldots \lor a_n \lor b_1 \lor \ldots \lor b_{j-1} \lor b_{j+1} \lor \ldots \lor b_m
\end{align*}
\]

This is called **resolution**. It is sound and complete when combined with a sound and complete search algorithm.
The World and the Model

KB → inference (syntactic) → true in the world

observation → (semantics)
Languages

Propositional logic isn’t very powerful.

How might we get more power?
First-Order Logic

More sophisticated representation language.

World can be described by:

- objects
- $\text{ColorOf}(\cdot)$
- $\text{Adjacent}(\cdot, \cdot)$
- $\text{IsApple}(\cdot)$
First-Order Logic

Objects:
• A “thing in the world”
  • Apples
  • Red
  • The Internet
  • Team Edward
  • Reddit
• A name that references something.
• Cf. a noun.
First-Order Logic

Functions:

• Operator that maps object(s) to single object.
  • \( \text{ColorOf}(\cdot) \)
  • \( \text{ObjectNextTo}(\cdot) \)
  • \( \text{SocialSecurityNumber}(\cdot) \)
  • \( \text{DateOfBirth}(\cdot) \)
  • \( \text{Spouse}(\cdot) \)

\[ \text{ColorOf(MyApple271)} = \text{Red} \]
First-Order Logic

Predicates - replaces proposition

Like a function, but returns True or False - holds or does not.

- IsApple(·)
- ParentOf(·, ·)
- BiggerThan(·, ·)
- HasA(·, ·)
First-Order Logic

We can build up complex sentences using logical connectives, as in propositional logic:

- \( \text{Fruit}(X) \implies \text{Sweet}(X) \)
- \( \text{Food}(X) \implies (\text{Savory}(X) \lor \text{Sweet}(X)) \)
- \( \text{ParentOf}(\text{Bob}, \text{Alice}) \land \text{ParentOf}(\text{Alice}, \text{Humphrey}) \)
- \( \text{Fruit}(X) \implies \text{Tasty}(X) \lor (\text{IsTomato}(X) \land \neg \text{Tasty}(X)) \)

Predicates can appear where a propositions appear in propositional logic, but functions cannot.
Models for First-Order Logic

Propositional logic: for a model:
  • Set the value of every variable in the KB to True or False.
  • $2^n$ models possible for $n$ propositions.

The situation is much more complex for FOL.

A model in FOL consists of:
  • A set of objects.
  • A set of functions + values for all inputs.
  • A set of predicates + values for all inputs.
Models for First-Order Logic

Consider:

**Objects**
- Orange
- Apple

**Predicates**
- IsRed(·)
- HasVitaminC(·)

**Functions**
- OppositeOf(·)

Example model:

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Argument</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IsRed</td>
<td>Orange</td>
<td>False</td>
</tr>
<tr>
<td>IsRed</td>
<td>Apple</td>
<td>True</td>
</tr>
<tr>
<td>HasVitaminC</td>
<td>Orange</td>
<td>True</td>
</tr>
<tr>
<td>HasVitaminC</td>
<td>Apple</td>
<td>True</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Function</th>
<th>Argument</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>OppositeOf</td>
<td>Orange</td>
<td>Apple</td>
</tr>
<tr>
<td>OppositeOf</td>
<td>Apple</td>
<td>Orange</td>
</tr>
</tbody>
</table>
Knowledge Bases in FOL

A KB is now:

- A set of objects.
- A set of predicates.
- A set of functions.
- A set of sentences using the predicates, functions, and objects, and **asserted to be true**.

<table>
<thead>
<tr>
<th><strong>Objects</strong></th>
<th><strong>Predicates</strong></th>
<th><strong>Functions</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Orange</td>
<td>$\text{IsRed}(\cdot)$</td>
<td>$\text{OppositeOf}(\cdot)$</td>
</tr>
<tr>
<td>Apple</td>
<td>$\text{HasVitaminC}(\cdot)$</td>
<td></td>
</tr>
</tbody>
</table>

$vocabulary$

$\text{IsRed(Apple)}$

$\text{HasVitaminC(Orange)}$
Knowledge Bases in FOL

Listing everything is tedious …
• Especially when general relationships hold.

We would like a way to say more general things about the world than explicitly listing truth values for each object.
Quantifiers

New weapon:

- **Quantifiers.**

Make generic statements about properties that hold for the entire collection of objects in our KB.

Natural way to say things like:

- All fish have fins.
- All books have pages.
- There is a textbook about AI.

Key idea: **variable + binding rule.**
Existential Quantifiers

There exists object(s) such that a sentence holds.

\[ \exists x, IsPresident(x) \]

“there exists”

temporary variable

sentence using variable
Universal Quantifiers

A sentence holds for all object(s).

\[ \forall x, \text{HasStudentNumber}(x) \implies \text{Person}(x) \]

“for every”

temporary variable

sentence using variable
Quantifiers

Difference in strength:

• Universal quantifier is very strong.
• So use weak sentence.

\[ \forall x, Bird(x) \implies Feathered(x) \]

• Existential quantifier is very weak.
• So use strong sentence.

\[ \exists x, Car(x) \land ParkedIn(x, E23) \]
Compound Quantifiers

\[ \forall x, \exists y, Person(x) \implies Name(x, y) \]

“every person has a name”
Common Pitfalls

\[ \forall x, \text{Bird}(x) \land \text{Feathered}(x) \]
Common Pitfalls

$$\exists x, \text{Car}(x) \implies \text{ParkedIn}(x, E23)$$
Inference in First-Order Logic

**Ground term**, or **literal** - an actual object:

\[
MyApple 12
\]

vs. a **variable**:

\[
x
\]

If you have only ground terms, you can convert to a propositional representation and proceed from there.

\[
IsTasty(Apple) : IsTastyApple
\]
Instantiation

Getting rid of variables: **instantiate** a variable to a literal.

**Why?**

Universally quantified:

\[ \forall x, \text{Fruit}(x) \implies \text{Tasty}(x) \]

\[ \text{Fruit(Apple)} \implies \text{Tasty(Apple)} \]

\[ \text{Fruit(Orange)} \implies \text{Tasty(Orange)} \]

\[ \text{Fruit(MyCar)} \implies \text{Tasty(MyCar)} \]

\[ \text{Fruit(TheSky)} \implies \text{Tasty(TheSky)} \]

For every object in the KB, just write out the rule with the variables substituted.
Instantiation

Existentially quantified:

- Invent a new name (Skolem constant)

\[ \exists x, \text{Car}(x) \land \text{ParkedIn}(x, E23) \]

\[ \text{Car}(C) \land \text{ParkedIn}(C, E23) \]

- Name cannot be one you’ve already used.
- Rule can then be discarded.
PROLOG

PROgramming in LOGic (Colmerauer, 1970s)

- General-purpose AI programming language
- Based on First-Order Logic
- Declarative

- Use centered in Europe and Japan
- Fifth-Generation Computer Project

- Some parts of Watson (pattern matching over NLP)
- Often used as component of a system.
DENDRAL and MYCIN


DENDRAL: (Feigenbaum et al. ~1965)
- Identify unknown organic molecules
- Eliminate most “chemically implausible” hypotheses.

MYCIN: (Shortliffe et al., 1970s)
- Identify bacteria causing severe infections.
- “research indicated that it proposed an acceptable therapy in about 69% of cases, which was better than the performance of infectious disease experts.”

Major issue: the Knowledge Bottleneck.