Example: The Wumpus World

- KB:
  \[ \neg S_{1,1} \neg S_{2,1} S_{1,2} \]
  \[ \neg B_{1,1} B_{2,1} \neg B_{2,1} \]

- Knowledge about the environment:
  \[ \neg S_{1,1} \Rightarrow \neg W_{1,1} \land \neg W_{1,2} \land \neg W_{2,1} \]
  \[ \neg S_{2,1} \Rightarrow \neg W_{1,1} \land \neg W_{2,1} \land \neg W_{2,2} \land \neg W_{3,1} \]
  \[ S_{1,2} \Rightarrow W_{1,1} \lor W_{1,2} \lor W_{2,2} \lor W_{1,3} \]

Need to specify knowledge explicitly for each location on the grid!
Summary of propositional logic

- The good: propositional logic is very simple!
  - Inference is simple, few rules.
- The bad: propositional logic is very simple!
  - We cannot express things in a compact way.
  
  E.g. for the Wumpus world, we need propositions for ALL positions.
  - We cannot say things like "for all squares, try X".

What should we do?

First-order logic (FOL)

- Add a few new elements:
  1. **Predicates** are used to describe objects, properties, relationships.
  2. **Quantifiers** ($\forall = "for all", \exists = "there exists") are used for statements that apply to a class of objects.

E.g. $\forall x \text{On}(x, \text{Table}) \rightarrow \text{Fruit}(x)$

- $\forall$ is a quantifier
- $x$ is a variable
- $\text{Table}$ is a constant
- $\text{On}$ is a predicate

Translate to English?
First-order logic (FOL)

- Add a few **new elements**:
  1. **Predicates** are used to describe objects, properties, relationships.
  2. **Quantifiers** (\(\forall\) = “for all”, \(\exists\) = “there exists”) are used for statements that apply to a class of objects.

  E.g. \(\forall x \text{ On}(x, \text{Table}) \rightarrow \text{Fruit}(x)\)

  - \(\forall\) is a quantifier
  - \(x\) is a variable
  - \(\text{Table}\) is a constant
  - \(\text{On}\) is a predicate

  *Translate to English?*

  Use of quantifiers allows FOL to handle **infinite domains**, while propositional logic can only handle finite domains.

---

Syntax of FOL: Basic elements

- **Constants**  
  - *Wumpus*, 2, CS424, ...

- **Connectives** \(\land, \lor, \neg, \Rightarrow\)

- **Variables**  
  - \(x, y, \ldots\)

- **Equality**  
  - \(=\)

- **Quantifiers**  
  - \(\forall, \exists\)

- **Predicates**  
  - \(\text{At}(\text{Wumpus}, x, y), \text{IsPit}(x, y), \ldots\)
    - *Input* is a variable or constant, *output* is True or False.

- **Functions**  
  - \(y=\text{SonOf}(x)\)
    - *Input* is a variable or constant, *output* is a variable or constant.
Types of sentences

1. **Term**: \( \text{constant, variable, function}(\text{term}_1, \ldots, \text{term}_n) \)

2. **Atomic sentences**: \( \text{predicate}(\text{term}, \text{term}) \)
   \[ \text{term}_1 = \text{term}_2 \]
   E.g. \( \text{At}(\text{Wumpus}, 2, 1) \)

3. **Complex sentences**: Combine atomic sentences using connectives
   E.g. \( \text{At}(\text{Wumpus}, 2, 1) \rightarrow \neg \text{At}(\text{Wumpus}, 1, 2) \)

Universal quantification

- **Form**: \( \forall \text{<variables>} \text{<sentence>} \)
  
  “Everyone taking AI is smart” \( \forall x \text{Taking}(x,\text{AI}) \rightarrow \text{Smart}(x) \)

- This is equivalent to the conjunction of all variable instantiations
  \( (\text{Taking}(\text{Alice},\text{AI}) \rightarrow \text{Smart}(\text{Alice})) \land (\text{Taking}(\text{Bob},\text{AI}) \rightarrow \text{Smart}(\text{Bob})) \land \ldots \)

- Typically, \( \rightarrow \) is the main connector with \( \forall \)
Example

• What does this statement mean?

\[ \forall x \text{ Taking}(x,\text{AI}) \land \text{Smart}(x) \]

Problem!

Translation is: “Everyone is taking AI and everyone is smart.”

• Common mistake: Using \( \land \) as the main connective with \( \forall \).

Existential quantification

• Form: \( \exists <\text{variables}> <\text{sentence}> \)

“Someone taking AI is smart” \( \exists x \text{ Taking}(x,\text{AI}) \land \text{Smart}(x) \)

• This is equivalent to the disjunction of all variable instantiations

\( (\text{Taking}(\text{Alice},\text{AI}) \land \text{Smart(Alice)}) \lor (\text{Taking}(\text{Bob},\text{AI}) \land \text{Smart(Bob)}) \lor \ldots \)

• Typically, \( \land \) is the main connector with \( \exists \)
Example

- What does this statement mean?
  $\exists x \; \text{Taking}(x,\text{AI}) \rightarrow \text{Smart}(x)$

Problem! This sentence is true if there is anyone who is not taking AI.

Remember:
$\text{Taking}(x,\text{AI}) \rightarrow \text{Smart}(x)$ is equivalent to $\neg \text{Taking}(x,\text{AI}) \lor \text{Smart}(x)$

- Common mistake: Using $\rightarrow$ as the main connective with $\exists$.

Properties of quantifiers

Basic rules:
- $\forall x \; \forall y$ is the same as $\forall y \; \forall x$
- $\exists x \; \exists y$ is the same as $\exists y \; \exists x$
- $\exists x \; \forall y$ is not the same as $\forall y \; \exists x$

E.g. $\exists x \; \forall y \; \text{Loves}(x,y)$  $\text{Translate to English?}$

$\forall y \; \exists x \; \text{Loves}(x,y)$
Properties of quantifiers

Basic rules:

• \( \forall x \forall y \) is the same as \( \forall y \forall x \)
• \( \exists x \exists y \) is the same as \( \exists y \exists x \)
• \( \exists x \forall y \) is not the same as \( \forall y \exists x \)

E.g. \( \exists x \forall y \) Loves\((x,y)\)

“There is a person who loves everyone in the world.”

\( \forall y \exists x \) Loves\((x,y)\)

“Everyone in the world is loved by at least one person.”

Example

• Let \( x \) and \( y \) be real numbers.
• For each of the following sentences, say whether it is valid, satisfiable or unsatisfiable.

1. \( \forall x \exists y \) \( x > y \)

2. \( \exists x \forall y \) \( x > y \)

(Hint: Consider the “domain” of \( x \) and \( y \).)
Quantifier duality

- Each quantifier can be expressed by using the other quantifier and negation:

  \( \forall x \text{ Loves}(x, \text{IceCream}) \) equivalent to \( \exists x \neg \text{Loves}(x, \text{IceCream}) \)

  \( \exists x \text{ Loves}(x, \text{Broccoli}) \) equivalent to \( \forall x \neg \text{Loves}(x, \text{Broccoli}) \)

Fun with Sentences

Given predicates \( \text{Brother}(x, y) \), \( \text{Sibling}(x, y) \), \( \text{Mother}(x, y) \), \( \text{Female}(x) \), \( \text{Parent}(x, y) \), \( \text{FirstCousin}(x, y) \), translate these from English to FOL:

1. Brothers are siblings.

2. Sibling is reflexive.

3. One’s mother is one’s female parent.

4. A first cousin is a child of a parent’s sibling.
Fun with Sentences

Given predicates \( \text{Brother}(x,y) \), \( \text{Sibling}(x,y) \), \( \text{Mother}(x,y) \), \( \text{Female}(x) \), \( \text{Parent}(x,y) \), \( \text{FirstCousin}(x,y) \), translate these from English to FOL:

1. Brothers are siblings.
   \[
   \forall x \forall y \text{Brother}(x,y) \rightarrow \text{Sibling}(x,y)
   \]

2. Sibling is reflexive.
   \[
   \forall x \forall y \text{Sibling}(x,y) \leftrightarrow \text{Sibling}(y,x)
   \]

3. One’s mother is one’s female parent.
   \[
   \forall x \forall y \text{Mother}(x,y) \leftrightarrow (\text{Female}(x) \land \text{Parent}(x,y))
   \]

4. A first cousin is a child of a parent’s sibling.
   \[
   \forall x \forall y \text{FirstCousin}(x,y) \leftrightarrow \exists p \exists s \text{Parent}(p,x) \land \text{Sibling}(p,s) \land \text{Parent}(s,y)
   \]

Equality

- Expression \( \text{term}_1 = \text{term}_2 \) is true under a given interpretation if and only if \( \text{term}_1 \) and \( \text{term}_2 \) refer to the same object.

E.g. \( \text{Obj}_1 = \text{Obj}_2 \) is satisfiable.
\( 2 = 2 \) is valid.

Example: definition of the sibling predicate:

\[
\forall x \forall y \text{Sibling}(x,y) \leftrightarrow \neg (x = y) \land \\
\exists m \exists f (m = f) \land \text{Parent}(m,x) \land \text{Parent}(f,x) \land \\
\text{Parent}(m,y) \land \text{Parent}(f,y)
\]
Truth in first-order logic

Sentences are true with respect to a model and an interpretation.

- The model contains objects and relations among them.
- The interpretation specifies referents for:
  - Constant symbols → objects
  - Predicate symbols → relations
  - Function symbols → functional relations
- An atomic sentence: \(\text{predicate}(\text{term}_1, \ldots, \text{term}_n)\)
  is true if and only if the relation referred to by \(\text{predicate}\)
  holds for the objects \(\text{term}_1, \ldots, \text{term}_n\).

Proofs

- The proof process can be viewed as a search in which the operators are inference rules:
  - Modus Ponens (MP)
    \[
    \alpha, \alpha \rightarrow \beta \quad \frac{Takes(Joe,Al) \quad Takes(Joe,Al) \rightarrow Cool(Joe)}{Cool(Joe)}
    \]
  - And-Introduction (AI)
    \[
    \alpha \quad \beta \quad \frac{\text{Cool(Joe)} \quad \text{CSMajor(Joe)}}{\text{Cool(Joe)} \land \text{CSMajor(Joe)}}
    \]
  - Universal Elimination (UE):
    \[
    \forall x \alpha \rightarrow y \quad \frac{\forall x \text{Takes}(x,Al) \rightarrow Cool(x)}{\text{Takes(Pat,Al) \rightarrow Cool(Pat)}}
    \]
**Example Proof**

**KB:**

<table>
<thead>
<tr>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob is a buffalo</td>
</tr>
<tr>
<td>Pat is a pig</td>
</tr>
<tr>
<td>Buffaloes outrun pigs</td>
</tr>
<tr>
<td>$\forall x \forall y \text{Buffalo}(x) \land \text{Pig}(y) \rightarrow \text{Faster}(x,y)$</td>
</tr>
</tbody>
</table>

**Question:** Who is faster, Bob or Pat?

---

**Example Proof**

**KB:**

<table>
<thead>
<tr>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob is a buffalo</td>
</tr>
<tr>
<td>Pat is a pig</td>
</tr>
<tr>
<td>Buffaloes outrun pigs</td>
</tr>
<tr>
<td>$\forall x \forall y \text{Buffalo}(x) \land \text{Pig}(y) \rightarrow \text{Faster}(x,y)$</td>
</tr>
</tbody>
</table>

**Question:** Who is faster, Bob or Pat?

**Resolution:**

<table>
<thead>
<tr>
<th>Rule</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI 1 &amp; 2</td>
<td>4. Buffalo(Bob) $\land$ Pig(Pat)</td>
</tr>
<tr>
<td>UE 3, x/Bob, y/Pat</td>
<td>5. Buffalo(Bob) $\land$ Pig(Pat) $\rightarrow$ Faster(Bob, Pat)</td>
</tr>
<tr>
<td>MP 4 &amp; 5</td>
<td>6. Faster(Bob, Pat)</td>
</tr>
</tbody>
</table>
Search with Primitive Inference Rules

- Operators are inference rules.
  - MP, IA, UE
- States are sets of sentences.
- Goal test checks state to see if it contains query sentence.

Solution: Apply standard search techniques!

Problem: Branching factor is huge! Especially for UE.

Idea: Find a substitution that makes the rule premise match some known facts => A single more powerful inference rule!

Unification

We say a substitution $\sigma$ unifies atomic sentences $p$ and $q$ if $p\sigma = q\sigma$.

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knows(John, x)</td>
<td>Knows(John, Jane)</td>
<td>${x/Jane}$</td>
</tr>
<tr>
<td>Knows(John, x)</td>
<td>Knows(y, Mary)</td>
<td>${y/John, x/Mary}$</td>
</tr>
<tr>
<td>Knows(John, x)</td>
<td>Knows(y, Mother(y))</td>
<td>${y/John, x/Mother(John)}$</td>
</tr>
</tbody>
</table>

Idea: Unify complex sentences with known facts to draw conclusions.

E.g. If we know $q$ and the rule: $\text{Knows}(John, x) \rightarrow \text{Likes}(John, x)$

Then we can conclude:
- Likes(John, Jane)
- Likes(John, Mary)
- Likes(John, Mother(John))
### Generalized Modus Ponens (GMP)

\[
p_1', p_2', \ldots, p_n', (p_1 \land p_2 \land \ldots \land p_n \Rightarrow q) \quad \text{where } p_i' \sigma = p_i \sigma \text{ for all } i
\]

E.g.  
\begin{align*}
p_1' &= \text{Faster}(\text{Bob}, \text{Pat}) \\
p_2' &= \text{Faster}(\text{Pat}, \text{Steve}) \\
p_1 \land p_2 &\Rightarrow q = \text{Faster}(x, y) \land \text{Faster}(y, z) \Rightarrow \text{Faster}(x, z) \\
\sigma &= x/\text{Bob}, y/\text{Pat}, z/\text{Steve} \\
q\sigma &= \text{Faster}(\text{Bob}, \text{Steve})
\end{align*}

GMP used with KB of Horn clauses (=exactly 1 positive literal):
- a single atomic sentence;
- or a clause of the form: (conjunction of atomic sentences) \(\Rightarrow\) (atomic sentence).

- All variables assumed to be universally quantified.

### Completeness in FOL

- Procedure \(i\) is complete if and only if:
  \[\text{KB} \vdash_i \alpha \quad \text{whenever} \quad \text{KB} \models \alpha\]
- GMP is complete for KBs of universally quantified Horn clauses, but incomplete for general first-order logic.
- Entailment in FOL is only semi-decidable: can find a proof when \(\text{KB entails } \alpha\), but not always when KB does not entail \(\alpha\).
  - Reduces to Halting Problem: proof may be about to terminate with success or failure, or may go on forever.
- Second complete inference procedure: Resolution
Resolution

- Sound and complete inference method for FOL.

- **Proof by negation:** To prove that KB entails \( \alpha \), instead we prove that \((KB \land \neg \alpha)\) is unsatisfiable.

- **Method:**
  - The \( KB \) and \( \neg \alpha \) are expressed in universally quantified, conjunctive normal form.
  - Repeat: The resolution inference rule combines two clauses to make a new one.
  - Continue until an empty clause is derived (contradiction).

Conjunctive Normal Form in FOL

- Literal = (possibly negated) atomic sentence, e.g., \( \neg \text{Rich(Me)} \)
- Clause = disjunction of literals, e.g. \( \neg \text{Rich(Me)} \lor \text{Unhappy(Me)} \)
- The KB is a big conjunction of clauses.

E.g. 

\[ \neg \text{Rich}(x) \lor \text{Unhappy}(x) \]

\[ \text{Rich(Me)} \]

\[ \text{Unhappy(Me)} \]

with \( \sigma = \{x / Me\} \)
Converting a KB to CNF

1. Replace $P \Rightarrow Q$ by $\neg P \lor Q$
2. Move $\neg$ inwards, e.g. $\neg \forall x P$ becomes $\exists x \neg P$
3. Standardize variables apart, e.g. $\forall x P \lor \exists x Q$ becomes $\forall x P \lor \exists y Q$
4. Move quantifiers left in order, e.g. $\forall x P \lor \exists x Q$ becomes $\forall x \exists y P \lor Q$
5. Eliminate existential quantifiers by Skolemization

Skolemization

- We want to get ride of existentially quantified variables: $\exists x \text{Rich}(x)$ becomes $\text{Rich}(G1)$ where $G1$ is a new Skolem constant.
- It gets more tricky when $\exists$ is inside $\forall$

E.g. "Everyone has a heart" $\forall x \text{Person}(x) \Rightarrow \exists y \text{Heart}(y) \land \text{Has}(x,y)$

How should we replace $y$ here?
- Incorrect: $\forall x \text{Person}(x) \Rightarrow \text{Heart}(H1) \land \text{Has}(x,H1)$
- Correct: $\forall x \text{Person}(x) \Rightarrow \text{Heart}(H(x)) \land \text{Has}(x,H(x))$

where $H$ is a new symbol called a Skolem function.
Converting a KB to CNF

1. Replace $P \implies Q$ by $\neg P \lor Q$
2. Move $\neg$ inwards, e.g. $\neg \forall x P$ becomes $\exists x \neg P$
3. Standardize variables apart, e.g. $\forall x P \lor \exists x Q$ becomes $\forall x P \lor \exists y Q$
4. Move quantifiers left in order, e.g. $\forall x P \lor \exists x Q$ becomes $\forall x \exists y P \lor Q$
5. Eliminate existential quantifiers by Skolemization
6. Drop universal quantifiers
7. Distribute over $\lor$, e.g. $(P \land Q) \lor R$ becomes $(P \lor R) \land (Q \lor R)$

Example

Jack owns a dog.
Every dog owner is an animal lover.
No animal lover kills an animal.
Either Jack or Curiosity killed the cat, who is named Tuna.
Did Curiosity kill the cat?
Example: Sentences + background knowledge

1. \( \exists x : \text{Dog}(x) \land \text{Owns}(\text{Jack}, x) \)

2. \( \forall x ; (\exists y \ \text{Dog}(y) \land \text{Owns}(x, y)) \rightarrow \text{AnimalLover}(x) \)

3. \( \forall x ; \text{AnimalLover}(x) \rightarrow (\forall y \ \text{Animal}(y) \rightarrow \neg \text{Kills}(x, y)) \)

4. \( \text{Kills}(\text{Jack}, \text{Tuna}) \lor \text{Kills}(\text{Curiosity}, \text{Tuna}) \)

5. \( \text{Cat}(\text{Tuna}) \)

6. \( \forall x : \text{Cat}(x) \rightarrow \text{Animal}(x) \)

Example: Conjunctive Normal Form

\( \text{Dog}(D) \) (\( D \) is a placeholder for the dog's unknown name \( \text{i.e.} \) Skolem symbol/function. Think of \( D \) like "JohnDoe")

\( \text{Owns}(\text{Jack}, D) \)

\( \neg \text{Dog}(y) \lor \neg \text{Owns}(x, y) \lor \text{AnimalLover}(x) \)

\( \neg \text{AnimalLover}(w) \lor \neg \text{Animal}(y) \lor \neg \text{Kills}(w, y) \)

\( \text{Kills}(\text{Jack}, \text{Tuna}) \lor \text{Kills}(\text{Curiosity}, \text{Tuna}) \)

\( \text{Cat}(\text{Tuna}) \)

\( \neg \text{Cat}(z) \lor \text{Animal}(z) \)

\( \neg \text{Kills}(\text{Curiosity}, \text{Tuna}) \)
Example: Find the mistake!

Resolution Strategies

Heuristics that impose a sensible order on the resolutions we attempt:

• **Unit resolution**: prefer to perform resolution if one clause is just a literal – yields shorter sentences.

• **Set of support**: identify a subset of the KB (hopefully small); every resolution will take a clause from the set and resolve it with another sentence, then add the result to the set of support.
  – Can make inference incomplete!

• **Input resolution**: always combine a sentence from the query or KB with another sentences. Not complete in general.
More resolution strategies

- **Linear resolution**: resolve P and Q if P is in the original KB or is an ancestor of Q in the proof tree.
- **Subsumption**: eliminate all sentences more specific than a sentence already in the KB.
- **Demodulation and paramodulation**: special extra inference rules to allow treatment of equality.

Simple problem

- **Schubert Steamroller**:
  - Wolves, foxes, birds, caterpillars, and snails are animals and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants.
  - Prove: there is an animal that likes to eat a grain-eating animal
  - Some of the necessary logical forms:
    - $\forall x (\text{Wolf}(x) \rightarrow \text{animal}(x))$
    - $\forall x \forall y ((\text{Caterpillar}(x) \lor \text{Bird}(y)) \rightarrow \text{Smaller}(x,y)$
    - $\exists x \text{bird}(x)$
  - Requires almost 150 resolution steps (minimal)
  - **Proofs can be lengthy!**
Successes in Rule-Based Reasoning

- Expert systems
- Dendral (Buchanan et al., 1969)
- MYCIN (Feigenbaum, Buchanan, Shortliffe)
- PROSPECTOR (Duda et al., 1979)
- R1 (McDermott, 1982)

Properties of knowledge-based systems

**Advantages**
1. Expressibility*: Human readable
2. Simplicity of inference procedures*: Rules/knowledge in same form
3. Modifiability*: Easy to change knowledge
4. Explainability: Answer “how” and “why” questions.
5. Machine readability
6. Parallelism*

**Disadvantages**
1. Difficulties in expressibility
2. Undesirable interactions among rules
3. Non-transparent behavior
4. Difficult debugging
5. Slow
6. Where does the knowledge base come from???
Other applications of FOL

• Prolog: a logic programming language
• Production systems
• Semantic nets
• Automated theory proving
• Planning (next week)

Important news

• Homework 2 posted. Due Feb.19. Next tutorial on Feb.16.

• Midterm is confirmed for Feb.23 at the usual class time.
  • Last name A-O go to MCMED 504.
  • Last name P-Z go to MCMED 521.