CSC242: Intro to AI
Lecture 11
Propositional Inference
Propositional Inference
Factored Representation

- Splits a state into variables (factors, attributes, features, “things you know”) that can have values
- Factored states can be more or less similar (unlike atomic states)
- Can also represent uncertainty (don’t know value of some attribute)
Constraint Satisfaction Problem (CSP)

X: Set of variables \{ X_1, ..., X_n \}
D: Set of domains \{ D_1, ..., D_n \}
  Each domain \( D_i = \) set of values \{ v_1, ..., v_k \}
C: Set of constraints \{ C_1, ..., C_m \}
Partially Observable

Don’t know everything about the state of the world

Have to represent uncertainty in our knowledge

Have to explore
Boolean[][] P =
    new Boolean[n][n];
Boolean[][] G =
    new Boolean[n][n];
Boolean[][] W =
    new Boolean[n][n];
Boolean[][] At =
    new Boolean[n][n];
Boolean FacingN, FacingS,
    FacingE, FacingW;
Boolean[][] S =
    new Boolean[n][n];
Boolean[][] B =
    new Boolean[n][n];
A Programming Language for Knowledge
if $A[i,j]$ and $S[i,j]$
then ($W[i-1,j]$ or $W[i+1,j]$ or $W[i,j-1]$ or $W[i,j+1]$)

if $A[i,j]$ and ($W[i-1,j]$ or $W[i+1,j]$ or $W[i,j-1]$ or $W[i,j+1]$)
then $S[i,j]$

Warning: not a real programming language
Syntax

• Propositional (boolean) variables

• Connectives:
  • Negation (not, \( \neg \))
  • Conjunction (and, \( \land \))
  • Disjunction (or, \( \lor \))
  • Implication (if-then, \( \Rightarrow \))
  • Biconditional (if and only if, \( \Leftrightarrow \))

• Precedence rules and parentheses
Semantics

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>¬P</th>
<th>P ∧ Q</th>
<th>P ∨ Q</th>
<th>P → Q</th>
<th>P ↔ Q</th>
</tr>
</thead>
<tbody>
<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>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</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>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>Hungry</td>
<td>Cranky</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Possible Worlds

Hungry=true, Cranky=false

Hungry=false, Cranky=true

Hungry=true, Cranky=true

Hungry=false, Cranky=false
Possible Worlds

Hungry \lor Cranky

Hungry=true, Cranky=false

Hungry=false, Cranky=true

Hungry=true, Cranky=true

Hungry=false, Cranky=false

Tuesday, February 26, 13
Possible Worlds

Hungry ⇒ Cranky

Hungry=true, Cranky=false

Hungry=false, Cranky=true

Hungry=true, Cranky=true

Hungry=false, Cranky=false
A sentence or set of sentences in propositional logic reduces the number of possible worlds by ruling out some of them.
## Satisfaction

<table>
<thead>
<tr>
<th>$B_{1,1}$</th>
<th>$P_{1,2}$</th>
<th>$P_{2,1}$</th>
<th>$P_{1,2} \lor P_{2,1}$</th>
<th>$B_{1,1} \iff (P_{1,2} \lor P_{2,1})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>
Model

• An assignment that satisfies a sentence or set of sentences

• A possible world where those sentences are true, and so is not ruled out by them

• May be how the world actually is
Unsatisfiable

• No complete, consistent assignment of truth values to the propositions that makes the sentence or set of sentences true

• Rules out all possible worlds

• Cannot describe the actual world
### Truth Table

<table>
<thead>
<tr>
<th>$B_{1,1}$</th>
<th>$P_{1,2}$</th>
<th>$P_{2,1}$</th>
<th>$P_{1,2} \lor P_{2,1}$</th>
<th>$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>
DFS for Boolean CSP

$O(2^n)$

Time Complexity

$O(n)$

Space Complexity
Exploration Strategy

Choose

Act

Observe

Update

Tuesday, February 26, 13
Inference

- Use our knowledge of the world to infer facts that we hadn't directly observed
Hungry ⇒ Cranky

Hungry

Cranky
\[
\begin{array}{cccc}
1,4 & 2,4 & 3,4 & 4,4 \\
1,3 & W! & 2,3 & 3,3 & 4,3 \\
1,2 & A & S & OK & 2,2 & OK & 3,2 & 4,2 \\
1,1 & V & OK & 2,1 & B & V & OK & 3,1 & P! & 4,1 \\
\end{array}
\]

<table>
<thead>
<tr>
<th>P</th>
<th>A</th>
<th>B</th>
<th>G</th>
<th>S</th>
<th>V</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>OK</td>
<td>Agent</td>
<td>Breeze</td>
<td>Glitter, Gold</td>
<td>Safe square</td>
<td>Pit</td>
<td>Stench</td>
</tr>
</tbody>
</table>

\[
P[1][1] = false; \quad B[2][1] = true; \\
W[1][1] = false; \quad S[2][1] = false; \\
B[1][1] = false; \quad B[1][2] = false; \\
S[1][1] = false; \quad S[1][2] = true; \\
W[1][2] = false; \quad W[1][3] = true; \\
P[1][2] = false; \quad P[2][2] = false; \\
W[2][1] = false; \quad P[3][1] = true; \\
P[1][2] = false; \\
\]
Inference

- What other things are we justified in believing, assuming our background knowledge and perceptions are accurate?
- What other sentences are true, given our background knowledge and perceptions?
Inference

- Infinite number of sentences of propositional logic, even over a finite set of propositional variables
- So ask whether a given sentence or set of sentences follows from our knowledge
What does it mean to “follow from” our knowledge?
$\alpha$, $\beta$ → $P[1,1] = \text{false}$, $B[1,2] = \text{false}$, $\ldots$
α → β
\[ P[1,1] = \text{true}, \quad B[1,2] = \text{false}, \]

\[ P[1,1] = \text{false}, \quad B[1,2] = \text{false}, \]

\[ P[1,1] = \text{false}, \quad B[1,2] = \text{true}, \]

\[ P[1,1] = \text{false}, \quad B[1,2] = \text{false}, \]

\[ \alpha \quad \rightarrow \quad \beta \]

\[ \alpha \rightarrow \beta \]

\[ \alpha \rightarrow \beta \rightarrow \ldots \]

\[ \alpha \rightarrow \beta \rightarrow \ldots \]

\[ \alpha \rightarrow \beta \rightarrow \ldots \]

\[ \alpha \rightarrow \beta \rightarrow \ldots \]

\[ \alpha \rightarrow \beta \rightarrow \ldots \]
Entailment

- $\alpha$ entails $\beta$ when:
  - $\beta$ is true in every world considered possible by $\alpha$
  - Every model of $\alpha$ is also a model of $\beta$
  - $\text{Models}(\alpha) \subseteq \text{Models}(\beta)$
Entailment is conservative

- Only accept a conclusion that is guaranteed to be true whenever the premises are true.
• Language for expressing knowledge: propositional logic

• Definition of entailment ("follows from")

• Need to be able to compute whether $\alpha$ entails $\beta$
## Model Checking

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>...</th>
<th>α</th>
<th>...</th>
<th>β</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>...</td>
<td>true</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>...</td>
<td>false</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>...</td>
<td>true</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>...</td>
<td>false</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>...</td>
<td>false</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>...</td>
<td>true</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>...</td>
<td>false</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>...</td>
<td>false</td>
<td>...</td>
<td>true</td>
</tr>
</tbody>
</table>
Propositional Inference
## Model Checking

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>α</th>
<th>β</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>...</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>...</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>...</td>
<td>true</td>
</tr>
</tbody>
</table>
Intuition: Math
Intuition: Math

\[
\begin{align*}
123 \\
+ & 456 \\
\hline
579
\end{align*}
\]
Intuition: Math

\[ x + 3 = 7 \]
\[ x + 3 - 3 = 7 - 3 \]
\[ x = 7 - 3 \]
\[ x = 4 \]
Mathematical Identities

- Allow us to rewrite equations
- Truth-preserving:
  - If the original equation holds, then so does the rewritten one
Inference Rules

• Look for rules that allow us to rewrite sentences in a truth-preserving way

• We’ll call these inference rules, since they will allow us to do inference (draw conclusions, make implicit knowledge explicit)
P ↔ Q
<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>P⇒Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>
Entailment

• $\alpha$ entails $\beta$ when:
  • $\beta$ is true in every world considered possible by $\alpha$
  • Every model of $\alpha$ is also a model of $\beta$
  • $\text{Models}(\alpha) \subseteq \text{Models}(\beta)$
\{ P \Rightarrow Q, P \} \models Q
\{ \alpha \Rightarrow \beta, \alpha \} \models \beta
Modus Ponens

\[ \alpha \Rightarrow \beta, \alpha \quad \frac{}{\beta} \]
Derivation

• $\beta$ can be derived from $\alpha$ using inference rules
• $\alpha \vdash \beta$
Properties of Inference
Rules
Soundness

- Derives only logically entailed sentences
- Truth-preserving

\[
\text{if } \alpha \vdash \beta \text{ then } \alpha \models \beta
\]
Completeness

- Derives all logically entailed sentences

\[
\text{if } \alpha \models \beta \text{ then } \alpha \vdash \beta
\]
Inference Rules

**And-elimination**
\[
\frac{\alpha \land \beta}{\alpha}
\]

**Double negation**
\[
\frac{\neg\neg\alpha}{\alpha}
\]

**DeMorgan’s Laws**
\[
\frac{\neg(\alpha \land \beta)}{\neg\alpha \lor \neg\beta}
\]
\[
\frac{\neg(\alpha \lor \beta)}{\neg\alpha \land \neg\beta}
\]

**Modus Ponens**
\[
\frac{\alpha \Rightarrow \beta, \alpha}{\beta}
\]

**Definition of biconditional**
\[
\frac{\alpha \leftrightarrow \beta}{(\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)}
\]
\[
\frac{(\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)}{\alpha \leftrightarrow \beta}
\]
Want to compute whether \( \alpha \vdash \beta \)

For a sound inference rule:

\[
\text{if } \alpha \vdash \beta \quad \text{then } \alpha \vdash \beta
\]

\[
\text{if } \alpha \vdash \gamma \text{ and } \gamma \vdash \beta \quad \text{then } \alpha \vdash \beta
\]
Proof

• Sequence of sound inference rule applications that lead from the premises to the desired conclusion
Background knowledge:

\[ R_1 : \neg P_{1,1} \]
\[ R_2 : B_{1,1} \iff (P_{1,2} \lor P_{2,1}) \]
\[ R_3 : B_{2,1} \iff (P_{1,1} \lor P_{2,2} \lor P_{3,1}) \]

Perceptions:

\[ R_4 : \neg B_{1,1} \]
\[ R_5 : B_{2,1} \]

Biconditional elimination on \( R_2 \):

\[ R_6 : ((B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})) \]

And-elimination on \( R_6 \):

\[ R_7 : (P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1} \]

Logical equivalence for contrapositives:

\[ R_8 : \neg B_{1,1} \Rightarrow \neg (P_{1,2} \lor P_{2,1}) \]

Modus Ponens on \( R_8 \) and \( R_4 \):

\[ R_9 : \neg (P_{1,2} \lor P_{2,1}) \]

DeMorgan’s Rule:

\[ R_{10} : \neg P_{1,2} \land \neg P_{2,1} \]

And-elimination on \( R_{10} \):

\[ R_{11} : \neg P_{1,2} \]
Propositional Inference As Search

- Initial state: Set of facts (initial knowledge base)
- Actions: Apply an inference rule to the sentences that match their premises
- Result: Add conclusions of inference rule to knowledge base
- Goal: The knowledge base contains the sentence we want to prove
Searching for proofs is an alternative to enumerating models

“In many practical cases, finding a proof can be more efficient because the proof can ignore irrelevant propositions, no matter how many of them there are.”
Propositional Inference

- Entailment: “follows from our knowledge”
- Model checking
- Inference rules: soundness, completeness
- Proof: search for sequence of sound inference rules from premises to conclusion
Propositional Inference As Search

- **Initial state:** Set of facts (initial knowledge base)
- **Actions:** Apply an inference rule to the sentences that match their premises
- **Result:** Add conclusions of inference rule to knowledge base
- **Goal:** The knowledge base contains the sentence we want to prove
Inference Rules

\[ \frac{\alpha \land \beta}{\alpha} \]
And-elimination

\[ \frac{\neg
\neg \alpha}{\alpha} \]
Double negation

\[ \frac{\neg(\alpha \land \beta)}{\neg\alpha \lor \neg\beta} \]
\[ \frac{\neg(\alpha \lor \beta)}{\neg\alpha \land \neg\beta} \]
DeMorgan’s Laws

\[ \frac{\alpha \Rightarrow \beta, \alpha}{\beta} \]
Modus Ponens

\[ \frac{\alpha \iff \beta}{(\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)} \]
Definition of biconditional

\[ (\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha) \]
\[ \alpha \iff \beta \]
Hungry ∨ Cranky

¬Hungry

Cranky
\[ P_{1,1} \lor P_{2,2} \lor P_{3,1} \]
\[ \neg P_{1,1} \]
\[ P_{2,2} \lor P_{3,1} \]
\[ \neg P_{2,2} \]
\[ P_{3,1} \]

**Legend:**
- **A** = Agent
- **B** = Breeze
- **G** = Glitter, Gold
- **OK** = Safe square
- **P** = Pit
- **S** = Stench
- **V** = Visited
- **W** = Wumpus

<table>
<thead>
<tr>
<th></th>
<th>1,1</th>
<th>2,1</th>
<th>3,1</th>
<th>4,1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,4</td>
<td>2,4</td>
<td>3,4</td>
<td>4,4</td>
<td></td>
</tr>
<tr>
<td>1,3</td>
<td>2,3</td>
<td>3,3</td>
<td>4,3</td>
<td></td>
</tr>
<tr>
<td>1,2</td>
<td>2,2</td>
<td>P?</td>
<td>3,2</td>
<td>4,2</td>
</tr>
<tr>
<td>1,1</td>
<td>2,1</td>
<td>B</td>
<td>3,1</td>
<td>P?</td>
</tr>
</tbody>
</table>

**Notes:**
- The table represents a grid with possible locations for various game elements.
- The symbols represent different game states and entities.
If A or B is true and you know it’s not A, then it must be B
Literals

• Literal: propositional variable (P) or negation of propositional variable (¬Q)

• Complementary literals: one literal is the negation of another (P, ¬P)
Clauses

- Clause: disjunction of literals:
  \[ P \lor \neg Q \lor \neg R \lor \neg P \]
- Unit clause: a single literal:
  \[ P \]
  \[ \neg Q \]
Unit Resolution

\[
\begin{align*}
  l_1 \lor \cdots \lor l_i \lor \cdots \lor l_k, & \quad m \\
  & \quad \frac{l_1 \lor \cdots \lor l_{i-1} \lor l_{i+1} \cdots \lor l_k}{l_1, \ldots, l_k \text{ and } m \text{ are literals}} \\
  & \quad \frac{l_i \text{ and } m \text{ are complementary}}{\text{Complementary literals:}} \\
  & \quad \text{Positive literal: } P \\
  & \quad \text{Negative literal: } \neg P
\end{align*}
\]
Hungry $\lor$ Cranky

$\neg$ Hungry

Cranky
<table>
<thead>
<tr>
<th></th>
<th>1,4</th>
<th>2,4</th>
<th>3,4</th>
<th>4,4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,3</td>
<td>2,3</td>
<td>3,3</td>
<td>4,3</td>
<td></td>
</tr>
<tr>
<td>1,2</td>
<td>2,2</td>
<td>P?</td>
<td>3,2</td>
<td>4,2</td>
</tr>
<tr>
<td>1,1</td>
<td>2,1</td>
<td>R</td>
<td>3,1</td>
<td>P?</td>
</tr>
<tr>
<td></td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
<td>OK</td>
</tr>
</tbody>
</table>

A = Agent  
B = Breeze  
G = Glitter, Gold  
OK = Safe square  
P = Pit  
S = Stench  
V = Visited  
W = Wumpus

\[
P_{1,1} \lor P_{2,2} \lor P_{3,1}
\]
\[
\neg P_{1,1}
\]
\[
P_{2,2} \lor P_{3,1}
\]
\[
\neg P_{2,2}
\]
\[
P_{3,1}
\]
Unit Resolution

• Sound: if $\alpha \vdash \beta$ then $\alpha \models \beta$

• Not complete:
  if $\alpha \models \beta$ then $\alpha \vdash \beta$ $\times$
Resolution

\[
\frac{l_1 \lor \cdots \lor l_i \lor \cdots \lor l_k, \quad m_1 \lor \cdots \lor m_j \lor \cdots \lor m_n}{l_1 \lor \cdots \lor l_{i-1} \lor l_{i+1} \cdots \lor l_k \lor m_1 \lor \cdots \lor m_{j-1} \lor m_{j+1} \cdots \lor m_n}
\]

\(l_1, \ldots, l_k, m_1, \ldots, m_n\) are literals
\(l_i\) and \(m_j\) are complementary

Technical note: Resulting clause must be factored to contain only one copy of each literal.
\[
\begin{align*}
P_{1,1} \lor P_{3,1}, & \quad \neg P_{1,1} \lor \neg P_{2,2} \\
\hline
\end{align*}
\]

\[
\begin{align*}
P_{3,1} \lor \neg P_{2,2}
\end{align*}
\]
1. Hungry ∨ Cranky
2. ¬Sleepy ∨ ¬Hungry
3. Cranky ∨ Sleepy
4. ¬Sleepy ∨ Cranky  (1,2)
5. Cranky ∨ Cranky  (3,4)
6. Cranky  (factoring)
Resolution

• Sound: if $\alpha \vdash \beta$ then $\alpha \models \beta$

• Easy to show

• Complete: if $\alpha \models \beta$ then $\alpha \vdash \beta$

• Proof by contradiction (see book)
Resolution

\[
\begin{align*}
    l_1 \lor \cdots \lor l_i \lor \cdots \lor l_k, & \quad m_1 \lor \cdots \lor m_j \lor \cdots \lor m_n \\
    \frac{l_1 \lor \cdots \lor l_{i-1} \lor l_{i+1} \cdots \lor l_k \lor m_1 \lor \cdots \lor m_{j-1} \lor m_{j+1} \cdots \lor m_n}{l_1, \ldots, l_k, m_1, \ldots, m_n \text{ are literals}} \\
    l_i \text{ and } m_j \text{ are complementary}
\end{align*}
\]

Technical note: Resulting clause must be factored to contain only one copy of each literal.
A \land B \quad P \Rightarrow Q \quad (A \lor B) \Rightarrow \neg (B \lor C)
Conjunctive Normal Form (CNF)

- Eliminate $\leftrightarrow$: $\alpha \leftrightarrow \beta \rightarrow \alpha \implies \beta \land \beta \implies \alpha$
- Eliminate $\implies$: $\alpha \implies \beta \rightarrow \neg \alpha \lor \beta$
- Move negation in:
  - $\neg \neg \alpha \rightarrow \alpha$
  - $\neg (\alpha \lor \beta) \rightarrow (\neg \alpha \land \neg \beta)$
  - $\neg (\alpha \land \beta) \rightarrow (\neg \alpha \lor \neg \beta)$
- Distribute $\lor$ over $\land$:
  - $(\alpha \lor (\beta \land \gamma)) \rightarrow ((\alpha \lor \beta) \land (\alpha \lor \gamma))$
Inference Using Resolution

• Convert sentences (KB) to CNF (set of clauses)

• Apply resolution inference rule to pairs of clauses with complementary literals

• Add resulting clause to set of clauses

• Until...
Proof by Contradiction

- $\alpha \models \beta$ if and only if $(\alpha \land \neg \beta)$ is unsatisfiable
- If negation of goal is inconsistent with our knowledge
- Then the goal itself is entailed by our knowledge
Resolution Refutation

- Convert \((KB \land \neg \alpha)\) to CNF
- Apply resolution rule until:
  - No new clauses can be added
    - KB does not entail \(\alpha\)
  - Two clauses resolve to yield the empty clause
    - KB entails \(\alpha\)
Proof Using Resolution

- Proof by contradiction
- Derive empty clause from \((KB \land \neg \alpha)\)
  (converted to CNF, of course)
Effective Resolution

• Definite clauses
  • Disjunction of literals with exactly one positive literal

• Horn clauses
  • Disjunction of literals with at most one positive literal

• Natural reading as facts and “if-then” rules
Forward Chaining

- Knowledge base of definite clauses: facts and rules
- If premises of a rule (conjunction of literals) are known
  - Add its conclusion (single literal) to set of known facts
- Until either query is added or no further inferences can be made
Backward Chaining

- Work backward from query $q$
- If $q$ is known to be true, we are done
- Otherwise find all rules whose conclusion (head) is $q$
  - If all the premises (body) of one of those rules can be proven true, then $q$ is true
Effective Resolution

- Definite clauses
  - Disjunction of literals with exactly one positive literal
- Horn clauses
  - Disjunction of literals with at most one positive literal
- Natural reading as facts and “if-then” rules
Propositional Inference

- Entailment: “follows from our knowledge”
- Model checking
  - Intractable
- But see also AIMA 7.6 & Project 2
Propositional Inference

- Inference rules: soundness, completeness
- Searching for proofs is an alternative to enumerating models
- May be faster in practice
Propositional Inference

• Resolution is a sound and complete inference rule

• Works on clauses (CNF)

• Special cases:
  • Definite & Horn clauses
  • Forward and backward chaining
Project 2: Logical Sudoku

- Formulate sudoku using logic
- Write a program that inputs a sudoku and outputs its logical representation
- Solve the puzzle using an inference engine
  - Which you don’t have to write...
- Write a program that outputs the solution of the sudoku nicely
- Due: Mon 4 Mar 23:00
For next time:

AIMA 8.0–8.3;
8.1.1–8.1.2 fyi

Quiz on PL