CAS CS 440
Lec 15

Logic IV

1. Review
   a. All we need is Resolution
   b. Convert $\boldsymbol{KB \cap \neg\alpha}$ to CNF
   c. Resolution on CNF form
   d.

$$\frac{\alpha \Rightarrow \beta, \quad \alpha}{\beta}.$$

$$\frac{\alpha \land \beta}{\alpha}.$$

$$\begin{aligned}
(\alpha \land \beta) &\equiv (\beta \land \alpha) && \text{commutativity of } \land \\
(\alpha \lor \beta) &\equiv (\beta \lor \alpha) && \text{commutativity of } \lor \\
((\alpha \land \beta) \land \gamma) &\equiv (\alpha \land (\beta \land \gamma)) && \text{associativity of } \land \\
((\alpha \lor \beta) \lor \gamma) &\equiv (\alpha \lor (\beta \lor \gamma)) && \text{associativity of } \lor \\
\neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \lor \beta) && \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\
\neg(\alpha \land \beta) &\equiv (\neg\alpha \lor \neg\beta) && \text{De Morgan} \\
\neg(\alpha \lor \beta) &\equiv (\neg\alpha \land \neg\beta) && \text{De Morgan} \\
(\alpha \land (\beta \lor \gamma)) &\equiv ((\alpha \land \beta) \lor (\alpha \land \gamma)) && \text{distributivity of } \land \text{ over } \lor \\
(\alpha \lor (\beta \land \gamma)) &\equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma)) && \text{distributivity of } \lor \text{ over } \land
\end{aligned}$$

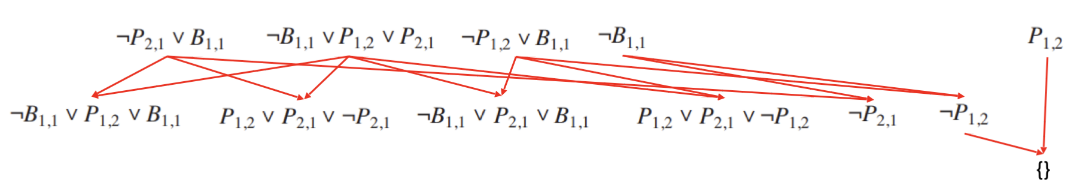
$$\frac{P_{1,1} \lor P_{2,2} \lor P_{3,1} \qquad \neg P_{2,2}}{P_{1,1} \lor P_{3,1}}$$

$$\frac{P_{1,1} \lor P_{3,1}, \qquad \neg P_{1,1} \lor \neg P_{2,2}}{P_{3,1} \lor \neg P_{2,2}}$$

$$\frac{\ell_1 \lor \cdots \lor \ell_k, \qquad m_1 \lor \cdots \lor m_n}{\ell_1 \lor \cdots \lor \ell_{i-1} \lor \ell_{i+1} \lor \cdots \lor \ell_k \lor m_1 \lor \cdots \lor m_{j-1} \lor m_{j+1} \lor \cdots \lor m_n}$$

- KB = $(B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})) \land \neg B_{1,1}$
- $\boldsymbol{\alpha}$ = $\neg P_{1,2}$,
- $\boldsymbol{KB \cap \neg\alpha}$:
  $(B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})) \land \neg B_{1,1} \land P_{1,2}$
- CNF of $\boldsymbol{KB \cap \neg\alpha}$:
  $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1}) \land \neg B_{1,1} \land P_{1,2}$

e.



2. Further Optimization
    a. In many KBs
        i. All sentences have the same template
        ii. Definite clause(s) appear all the time
            1. Disjunction where only one literal is positive

$$(\neg L_{1,1} \vee \neg Breeze \vee B_{1,1})$$

        iii. Horn clause(s) appear all the time
            1. Disjunction where at most one literal is positive
            2. Definite clauses are Horn clauses
            3. Clauses with no positives = goal clauses
    b. If you resolve two Horn clauses, the result is a Horn clause!
3. Horn clauses
    a. Why are these so popular?
        i. Remember logical equivalency!
        ii.

$$\underbrace{(L_{1,1} \wedge Breeze)}_{\text{premise}} \Rightarrow \underbrace{B_{1,1}}_{\text{head}} \xrightarrow{\text{To CNF}} \overset{\text{Horn clause!}}{(\neg L_{1,1} \vee \neg Breeze \vee B_{1,1})}$$

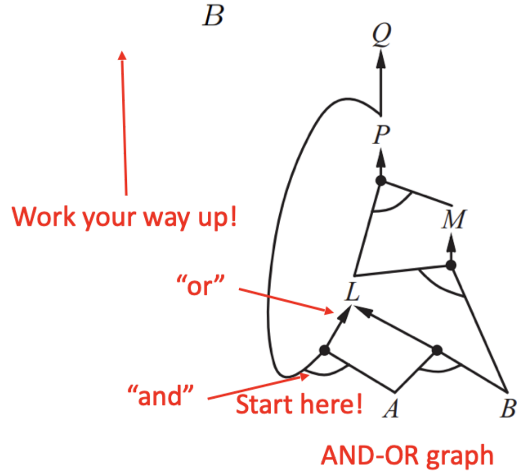
    b. Horn clauses are implication sentences!
        i. Axioms are typically implications!
    c. Sentence that is just a literal?
        i. Perceptions typically written this way
        ii. Called a "fact"

$$True \Rightarrow L_{1,1}$$

        iii. Can also be written as a Horn clause!
4. Inference with Horn Clauses
    a. Can be crazy efficient
        i. Forward-chaining & backward-chaining algorithms
        ii. If everything is a Horn clause (implies)
            1. Don't need to do resolution at all!
    b. Forward chaining (try to derive query sentence 'q')
        i. Begin with known facts
        ii. For every implication where the premise is met:
            1. Add conclusion (head) of implication as a fact to the KB
        iii. Repeat until 'q' is added to KB or no further implications!
        iv. Runs in linear time!
        v. Data-driver Reasoning
            1. Start with what you know: infer what you don't

$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow P$$
$$B \wedge L \Rightarrow M$$
$$A \wedge P \Rightarrow L$$
$$A \wedge B \Rightarrow L$$
$$A$$
$$B$$

KB

c. 

5. Forward-Chaining
    a. Sound & Complete

    b.

```
function PL-FC-ENTAILS?(KB, q) returns true or false
   inputs: KB, the knowledge base, a set of propositional definite clauses
           q, the query, a proposition symbol
   count ← a table, where count[c] is the number of symbols in c's premise
   inferred ← a table, where inferred[s] is initially false for all symbols
   agenda ← a queue of symbols, initially symbols known to be true in KB
   while agenda is not empty do
      p ← POP(agenda)
      if p = q then return true
      if inferred[p] = false then
         inferred[p] ← true
         for each clause c in KB where p is in c.PREMISE do
            decrement count[c]
            if count[c] = 0 then add c.CONCLUSION to agenda
   return false
```

Rules can by cyclic!
Don't add variables back to the Queue if we've processed them!

$P \Rightarrow Q$ and $Q \Rightarrow P$

6. Forward-Chaining in the Wumpus World
    a. Whenever we get some new perceptions:
        i. Engineer the KB to write axioms as implications
        ii. Add variables for the location of the agent!
        iii. Add the facts to the KB
        iv. Run Forward-Chaining to derive all new facts!

b. Humans do Forward-Chaining
    i. Keep it under control though
    ii. Don't need to try to derive every new fact
7. Backward-Chaining
    a. (try to derive query sentence 'q')
    b. Work backwards (instead of forwards)
    c. Find implications in KB whose conclusion is 'q'
        i. Try to prove premises are true (via backward chaining)
        ii. If one premise is true, then 'q' is true!
    d. Also runs in linear time!
    e. Goal-directed reasoning:
        i. Start with what you want: derive what you know
        ii. Often sublinear: only touches relevant facts

    $P \Rightarrow Q$

    $L \wedge M \Rightarrow P$

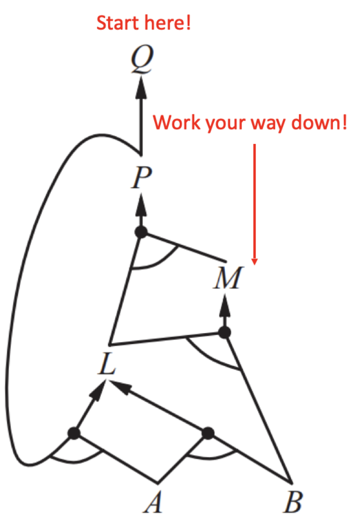
    $B \wedge L \Rightarrow M$

    $A \wedge P \Rightarrow L$

    $A \wedge B \Rightarrow L$

    A

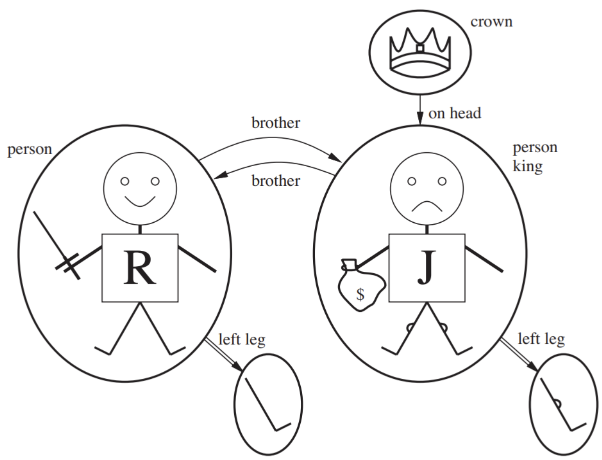
    f. B

    g. 

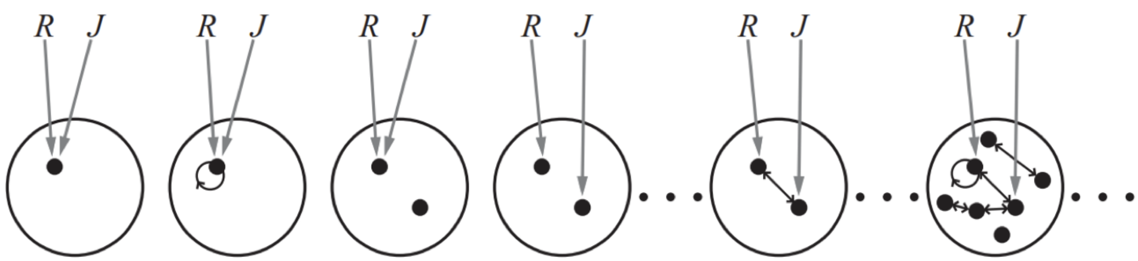
8. First Order Logic
    a. Prepositional logic has some nice properties
        i. Declarative
            1. Knowledge and inference are separate things
            2. Knowledge (i.e. sentences) declare things to be true/false
            3. Inference is domain independent!
        ii. Compositionality
            1. Sentence meaning is a function of its parts

b. Prepositional logic is not concise
    i. Lots
    ii. And lots
    iii. And lots of sentences
9. FOL Models
    a. Logical languages have models:
        i. Hypothetical worlds
        ii. Links the vocabulary to elements
        iii. Determine the truth of sentence(s)
    b. Models in FOL:
        i. Have objects in them
        ii. Domain of a model = set of objects (domain elements) it contains
        iii. Objects can be related:
            1. Relation is the set of tuples of objects that are related

Brotherhood = {(Richard, John), (John, Richard)}   OnHead = {(crown, John)}

    c. 

    d. Some Relations are Functions
        i. An object is related to exactly one other object in a certain way
    e. People only have one left leg
        i. LeftLeg:
            1. (Richard) -> Richard's left leg
            2. (John) -> John's left leg
    f. Functions in FOL must be total
        i. Must be a value for every input tuple
            1. Must be a left leg for the crown!
            2. Each left leg has a left leg!
        ii. Add an "invisible" object for base case
            1. Left leg for everything that doesn't have one
            2. Left leg of it is itself

10. FOL Syntax
    a. Types of symbols in the language:
        i. Constants (objects)
        ii. Predicates (relations)
        iii. Functions
    b. Convention in FOL is to start symbols with capital letters
    c. Predicate/Function symbols have arity
        i. Number of arguments
11. FOL Syntax & Models
    a. Every model must map symbols to objects (in the model)
        i. Called an “interpretation” of the symbols
        ii. Don’t need to name all objects in the model
        iii. Can assign multiple symbols to the same object
    b. Remember, it’s the KBs job to rule out inconsistent models!
        i. Lots of models are garbage
    c. 

12. Good News
    a. Can bring lots over from prepositional logic
        i. Entailment
        ii. Validity
        iii. Etc.
    b. Term
        i. Logical expression that refers to an object
        ii. Constant symbols are terms
            1. John
            2. *LeftLeg*(John)
        iii. This is not a function call, it’s a name!
            1. Can reason about left legs without defining *LeftLeg*

13. FOL Syntax

a. Atomic sentences (atom)

$$\forall x \;\; \forall y \;\; Brother(x, y) \;\Rightarrow\; Sibling(x, y)$$

i.

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

ii. State facts

iii. Predicate symbol (optionally w arguments)

iv. Can be true/false in a given model

1. If relation referred by predicate holds among the objects referred by the arguments

$$Brother(Richard, John)$$

v. Is $(Richard,\ John) \in Brother$?

b. Complex sentences

i.

variable

$$\forall x \;\; King(x) \;\Rightarrow\; Person(x)$$

ii. Logical operators

1. Same as prepositional logic!

iii. Quantifiers

1. $\exists x \;\; Crown(x) \land OnHead(x, John)$
2. Express properties of collections (rather than enumerating)
3. $\forall$ (forall)
4. $\exists$ (exists)