Inference in first-order logic

Our goal is to prove that KB entails a fact, $\alpha$

- We use logical inference
  - Forward chaining
  - Backward chaining
  - Resolution

All three logical inference systems rely on search to find a sequence of actions that derive the empty clause
Search and forward chaining

Start with KB full of first-order definite clauses

- Disjunction of literals with exactly one positive
  - Equivalent to implication with conjunction of positive literals on left (antecedent / body / premise) and one positive literal on right (consequent / head / conclusion)
  - Propositional logic used Horn clauses, which permit zero or one to be positive
- Look for rules with premises that are satisfied (use substitution to make matches) and add conclusions to KB
Search and forward chaining

Breadth First

- A, B, D, G, H
- A ^ E => C
- B ^ D => E
- E ^ C ^ G ^ H => I

Which rules have premises that are satisfied (modus ponens)?

- A ^ E => C... nope
- B ^ D => E... yes
- E ^ C ^ G ^ H => I... nope
  - A ^ E = C... yes
  - E ^ C ^ G ^ H ^ I... nope
  one more try... yes
Search and forward chaining

Would other search methods work?

- Yes, this technique falls in standard domain of all searches
Search and backward chaining

**Start with KB full of implications**

- Find all implications with conclusion matching the query
- Add to fringe list the unknown premises
  - Adding could be to front or rear of fringe (depth or breadth)
Search and backward chaining

Depth First

- A, B, D, G, H
- A \land E \Rightarrow C
- B \land D \Rightarrow E
- C \land E \land G \land H \Rightarrow I

- Are all the premises of I satisfied? No
  - For each (C E G H) are each of their premises satisfied?
    - C? no, put its premises on fringe
      - For each (A and E) are their premises satisfied?
        - A... yes
        - E... no, add premises for each B and D
          - B... yes
          - D... yes
  - E, G, H... yes
Search and backward chaining

Breadth First
- A, B, D, G, H
- A \land E \Rightarrow C
- B \land D \Rightarrow E
- C \land E \land G \land H \Rightarrow I

- Are all the premises of I satisfied? No
  - For each (C E G H) are each of their premises satisfied?
    - C? no, put its premises on fringe end
    - E? no, put its premises on fringe end
    - G, H... yes
      - Are C’s premises (A E) satisfied?
        - A... yes
        - E... no, add premises
      - Are E’s premises (B D) satisfied?
        - Yes
  - Return to C and I
Inference with resolution

- We put each first-order sentence into conjunctive normal form
  - We remove quantifiers
  - We make each sentence a disjunction of literals (each literal is universally quantified)
- We show $KB \wedge \{\neg \alpha\}$ is unsatisfiable by deriving the empty clause
  - Resolution inference rule is our method
    - Keep resolving until the empty clause is reached
Resolution

*Look for matching sentences*

- Shared literal with opposite sign
  - Substitution may be required
- \([\text{Animal } (F(x)) \lor \text{Loves } (G(x), x)]\) and \([\sim \text{Loves } (u,v) \lor \sim \text{Kills } (u, v)]\)
  - \(F(x) = \text{animal unloved by } x\)
  - \(G(x) = \text{someone who loves } x\)
Resolution

What does this mean in English?

- \([\text{Animal (} F(x) \text{)} \lor \text{Loves (} G(x), x \text{)}]\]
  - \(F(x) = \text{animal unloved by } x\)
  - \(G(x) = \text{someone who loves } x\)

- \([\neg \text{Loves (} u, v \text{)} \lor \neg \text{Kills (} u, v \text{)}]\]

- For all people, either a person doesn’t love an animal or someone loves the person
- Nobody loves anybody or nobody kills anybody
Resolution

- \([\text{Animal (} F(x) \text{)} \lor \text{Loves (} G(x), x \text{)}\)] \text{ and} \quad \[\sim\text{Loves (} u,v \text{)} \lor \sim\text{Kills (} u,v \text{)}\]
  - Loves and ~Loves cancel with substitution
    - \(u/G(x)\) and \(v/x\)

- Resolvent clause
  - \([\text{Animal (} F(x) \text{)} \lor \sim\text{Kills (} G(x), x \text{)}\)]
Example

\[ \neg \text{American}(x) \lor \neg \text{Weapon}(y) \lor \neg \text{Sells}(x, y, z) \lor \neg \text{Hostile}(z) \lor \text{Criminal}(x) . \]

\[ \neg \text{Missile}(x) \lor \neg \text{Owns}(\text{Nono}, x) \lor \text{Sells}(\text{West}, x, \text{Nono}) . \]

\[ \neg \text{Enemy}(x, \text{America}) \lor \text{Hostile}(x) . \]

\[ \neg \text{Missile}(x) \lor \text{Weapon}(x) . \]

\text{Owns}(\text{Nono}, M_1) . \quad \text{Missile}(M_1) .

\text{American}(\text{West}) . \quad \text{Enemy}(\text{Nono}, \text{America}) . \]
Resolution example

\[ \neg \text{American}(x) \lor \neg \text{Weapon}(y) \lor \neg \text{Sells}(x,y,z) \lor \neg \text{Hostile}(z) \lor \text{Criminal}(x) \]

\[ \neg \text{Criminal}(\text{West}) \]

\[ \text{American}(\text{West}) \]

\[ \neg \text{American}(\text{West}) \lor \neg \text{Weapon}(y) \lor \neg \text{Sells}(\text{West},y,z) \lor \neg \text{Hostile}(z) \]

\[ \neg \text{Missile}(x) \lor \text{Weapon}(x) \]

\[ \neg \text{Weapon}(y) \lor \neg \text{Sells}(\text{West},y,z) \lor \neg \text{Hostile}(z) \]

\[ \text{Missile}(M_1) \]

\[ \neg \text{Missile}(y) \lor \neg \text{Sells}(\text{West},y,z) \lor \neg \text{Hostile}(z) \]

\[ \neg \text{Missile}(x) \lor \neg \text{Owns}(\text{Nono},x) \lor \text{Sells}(\text{West},x,\text{Nono}) \]

\[ \neg \text{Sells}(\text{West},M_1,z) \lor \neg \text{Hostile}(z) \]

\[ \text{Missile}(M_1) \]

\[ \neg \text{Missile}(M_1) \lor \neg \text{Owns}(\text{Nono},M_1) \lor \neg \text{Hostile}(\text{Nono}) \]

\[ \text{Owns}(\text{Nono},M_1) \]

\[ \neg \text{Owns}(\text{Nono},M_1) \lor \neg \text{Hostile}(\text{Nono}) \]

\[ \neg \text{Enemy}(x,\text{America}) \lor \text{Hostile}(x) \]

\[ \neg \text{Hostile}(\text{Nono}) \]

\[ \text{Enemy}(\text{Nono},\text{America}) \]

\[ \neg \text{Enemy}(\text{Nono},\text{America}) \]
Inference with resolution

What resolves with what for proof?

- Unit preference
  - Start with single-literal sentences and resolve them with more complicated sentences
  - Every resolution reduces the size of the sentence by one
    - Consistent with our goal to find a sentence of size 0
  - Resembles forward chaining
Inference with resolution

**What resolves with what for proof?**

- Set of support
  - Build a special set of sentences
  - Every resolution includes one sentence from set
    - New resolvent is added to set
  - Resembles backward chaining if set of support initialized with negated query
Theorem provers

Logical inference is a powerful way to “reason” automatically

- Prover should be independent of KB syntax
- Prover should use control strategy that is fast
- Prover can support a human by
  - Checking a proof by filling in voids
Rational agents

**Up until now**

- Many rules were available and rationality was piecing rules together to accomplish a goal
  - Inference and deduction

**Now**

- Lots of data available (cause/effect pairs) and rationality is improving performance with data
  - Model building, generalization, prediction