TDT4136 Introduction to Artificial Intelligence

Lecture 7 - Inference in First Order Logic

Chapter 9 in the textbook

Keith L. Downing

The Norwegian University of Science and Technology
Trondheim, Norway
keithd@idi.ntnu.no

October 2, 2024

Outline

- Two main types of inference in FOL
  - Reduction of FOL to propositional logic (and use propositional inference methods)
    - Substitution
  - Inference using FOL Inference rules
    - Unification
    - Standardizing apart
    - Conversion of FOL sentences to CNF
    - Inference using Resolution, on CNF representation
    - Forward/Backward Chaining, on Definite Clauses

Inference by Reduction to Propositionalized Sentences

Strategy: convert KB to propositional logic and then use PL inference

- Every FOL KB can be propositionalized so as to preserve entailment - A sentence is entailed by new KB iff it is entailed by the original KB

Inference by Reduction to Propositionalized Sentences

Strategy: convert KB to propositional logic and then use PL inference

- Every FOL KB can be propositionalized so as to preserve entailment - A sentence is entailed by new KB iff it is entailed by the original KB
- Ground atomic sentences become propositional symbols

Inference by Reduction to Propositionalized Sentences

Strategy: convert KB to propositional logic and then use PL inference

- Every FOL KB can be propositionalized so as to preserve entailment - A sentence is entailed by new KB iff it is entailed by the original KB
- Ground atomic sentences become propositional symbols
- What about the quantifiers?

Propositionalization

- Suppose the KB contains just the following:
  - $\forall x\ King(x) \land Greedy(x) \implies Evil(x)$
  - King(John)
  - Greedy(John)
  - Brother(Richard,John)
- The last 3 sentences can be symbols in PL
- Apply **Universal Instantiation** to the first sentence, to get rid of the *quantifier* and the *variable*.

Universal Instantiation

- UI says that from a universally quantified sentence, we can infer any sentence obtained by substituting a ground term[1] for the variable

[1] A term is used to denote an object in the world. It can be a constant, variable or a function. A ground term is a term with no variables.

Universal Instantiation

- UI says that from a universally quantified sentence, we can infer any sentence obtained by substituting a ground term[1] for the variable
- The **Substitution** rule for instantiation of variables

$$\frac{\forall x \;\; \alpha}{\textsc{Subst}(\{x/g\}, \alpha)}$$

variable x in the sentence α is substituted with a ground term g

[1] A term is used to denote an object in the world. It can be a constant, variable or a function. A ground term is a term with no variables.

Universal Instantiation

- UI says that from a universally quantified sentence, we can infer any sentence obtained by substituting a ground term[1] for the variable
- The **Substitution** rule for instantiation of variables

  $$\frac{\forall x \;\; \alpha}{\textsc{Subst}(\{x/g\}, \alpha)}$$

  variable x in the sentence α is substituted with a ground term g
- Variable x is *substituted* with the *ground terms* referring to the objects *John* and *Richard* in the model one by one:
  - King(Richard) $\wedge$ Greedy(Richard) $\implies$ Evil(Richard)
  - John King(x) $\wedge$ Greedy(John) $\implies$ Evil(John)

[1]A term is used to denote an object in the world. It can be a constant, variable or a function. A ground term is a term with no variables.

Existential Instantiation

- Through **Skolemization**: Each existentially quantified variable is replaced by a *Skolem constant* or a *Skolem function*.
- **Skolem Constant**: if the existential variable is not within the scope of any universaly quantified variable. Every instance of the existentially quantified variable is replaced with the same unique constant, a brand new one that does not appear elsewhere in the knowledge base.

$$\frac{\exists x \ \ \alpha}{\textsc{Subst}(\{x/k\}, \alpha)}$$

Example: $\exists y \ \ (P(y) \wedge Q(y))$ is converted to: $P(CC) \wedge Q(CC)$

Existential Instantiation

Skolem Function: If the existential quantifier is in the *scope* (i.e., "inside") of a (or more - n) universally quantified variable(s), then replace it with a unique n-ary function over these universally quantified variables. Remove then the existential quantifier.

E.g., $\forall x \exists y \ (P(x) \vee Q(y))$ converted to $\forall x P(x) \vee Q(F(x))$

Problems with UI

1. instantiation of functions leads to inifinite nr of instantiations
2. creates many irrelevant instantiations

Problems with UI

1. instantiation of functions leads to inifinite nr of instantiations
2. creates many irrelevant instantiations

Problem when a variable is substituted with a function:

- For example, suppose Father is a function in our KB:
- Then Father(John) is also an object/ground term

Problems with UI

1. instantiation of functions leads to inifinite nr of instantiations
2. creates many irrelevant instantiations

Problem when a variable is substituted with a function:

- For example, suppose Father is a function in our KB:
- Then Father(John) is also an object/ground term
- Also Father(Father(John)) and Father(Father(Father(John)))..... are ground terms
- Infinitely many ground terms/instantiations

Problems with UI

1. instantiation of functions leads to inifinite nr of instantiations
2. creates many irrelevant instantiations

Problem when a variable is substituted with a function:

- For example, suppose Father is a function in our KB:
- Then Father(John) is also an object/ground term
- Also Father(Father(John)) and Father(Father(Father(John)))..... are ground terms
- Infinitely many ground terms/instantiations
- Solution: Herbrand Theorem (1930). If a sentence α is entailed by an FOL KB, it is entailed by a finite subset of the propositionalized KB
  - Idea: For n = 0 to ∞ do
    create a propositional KB by instantiating with depth-*n* terms
    see if α is entailed by this KB

Propositionalization creates irrelevant instantiations

- Propositionalization seems to generate lots of "irrelevant sentences". E.g., from

$$\forall x \;\; King(x) \wedge Greedy(x) \implies Evil(x)$$
$$King(John)$$
$$Greedy(John)$$
$$Brother(Richard, John)$$

Propositionalization creates irrelevant instantiations

- Propositionalization seems to generate lots of "irrelevant sentences". E.g., from

  $\forall x \;\; King(x) \land Greedy(x) \implies Evil(x)$
  $King(John)$
  $Greedy(John)$
  $Brother(Richard, John)$

- It seems obvious that $Evil(John)$ will be inferred at the end, but propositionalization produces lots of facts such as $Greedy(Richard)$ that are irrelevant
- With p k-ary predicates and n constants, there are $p \cdot n^k$ instantiations

Problem with propositionalization

- Problem : Universal elimination gives us (too) many opportunities for substituting variables with ground terms
- Solution: avoid making blind substitution of ground terms
  - Make substitutions that help to advance inferences
  - i.e., use substitutions matching "similar" sentences in KB
- How? - Inference without propositionalization

Propositionalization is semi-decidable

- Propositionalization is complete if the query sentence is entailed.
- Works if α is entailed, loops if α is not entailed. Hence, it is semi-decidable.

Propositionalization is semi-decidable

- Propositionalization is complete if the query sentence is entailed.
- Works if α is entailed, loops if α is not entailed. Hence, it is semi-decidable.
- Semidecidable : algorithms exist that say yes to every entailed sentence, but no algorithm exists that also says no to every nonentailed sentence.

Inference in First-Order Logic - without propositionalization

- Inference rules for propositional logic: – Modus ponens, and-introduction, resolution, etc.
- These are valid for FOL also

Inference in First-Order Logic - without propositionalization

- Inference rules for propositional logic: – Modus ponens, and-introduction, resolution, etc.
- These are valid for FOL also
- But since these don't deal with quantifiers and variables, we need new rules, especially those that allow for substitution of variables with objects
- These are called lifted inference rules

Lifted Modus Ponens

Modus Ponens in propositional logic:

$$\begin{array}{l} \alpha \rightarrow \beta \\ \alpha \\ \hline \beta \end{array}$$

Generalized Modus Ponens - FOL

$$\frac{P_1{}',\ \ P_2{}',\ \ldots,\ P_n{}',\ \ (P_1 \wedge P_2 \wedge \ldots \wedge P_n \Rightarrow Q)}{Q\theta}$$

where $P_i{}'\theta = P_i\theta$ for all i , θ is Substitution

Lifted Inference Rules and Unification

- Generalized Modus ponens is the *lifted* version of Modus Ponens in PL(i.e., variable free version)
- Lifted Inferences need finding substitutions that make 2 sentences look identical.
- This is called **Unification**

Lifted Inference Rules and Unification

- Generalized Modus ponens is the *lifted* version of Modus Ponens in PL(i.e., variable free version)
- Lifted Inferences need finding substitutions that make 2 sentences look identical.
- This is called **Unification**
- UNIFICATION: takes two similar sentences and computes the **unifier** for them (a **substitution**) that makes them look the same, if it exists UNIFY(P,Q) $= \theta$; where SUBST (θ, P)= SUBST (θ, Q)

Motivating example for Unification

- Ground clauses are clauses with no variables in them. For ground clauses we can use syntactic identity to detect when we have a P and ¬P pair.
- What about variables? For example, can the following two clauses be resolved?
  - P(john) ∨ Q(fred) ∨ R(x))
  - ¬ P(y)∨ R(susan)∨ R(y)

Generalzed Modus Ponens in FOL

- Suppose this KB:

$$S1 : \forall x \ \ King(x) \land Greedy(x) \implies Evil(x)$$
$$S2 : King(John)$$
$$S3 : \forall y \ \ Greedy(y)$$

- We want to find if John is Evil.
- Make a **substitution** for values of variables that make the premise of the implication identical to the sentences in the knowledge base.
  - ${P_1}'$ is King(John) and P_1 is King(x)
  - ${P_2}'$ is Greedy(y) and P_2 is Greedy(x)
  - θ is $\{x/John, y/John\}$,
  - Q is Evil(x)
  - SUBS(θ,Q) is Evil(John)

Unification process

- Unify procedure: Unify(P,Q) takes two atomic (i.e. single predicates) sentences P and Q and returns a substitution that makes P and Q identical.
- The aim is be able to match literals even when they have variables.
- Rules for substitutions: Can replace a variable
  - by a constant.
  - by a variable.
  - by a function expression, as long as the function expression does not contain the variable.

Unifier: a substitution that makes two clauses resolvable. e.g., θ:$\{x_1/M; x_2/x_3; x_4/F(..)\}$

More on Unification

- Not all formulas can be unified - substitutions only affect variables.
  - Example: Consider the pair P(F(x),A) and P(y,F(w))
  - This pair cannot be unified as there is no way of making A = F(w) with a substitution.

More on Unification

- Not all formulas can be unified - substitutions only affect variables.
  - Example: Consider the pair P(F(x),A) and P(y,F(w))
  - This pair cannot be unified as there is no way of making A $=$ F(w) with a substitution.
- In θ, each variable is paired at most once

More on Unification

- Not all formulas can be unified - substitutions only affect variables.
  - Example: Consider the pair P(F(x),A) and P(y,F(w))
  - This pair cannot be unified as there is no way of making A = F(w) with a substitution.
- In θ, each variable is paired at most once
- A variable's pairing term may not contain the variable directly or indirectly.
  - e.g. can't have substitution { x/F(y), y/F(x) }

More on Unification

- Not all formulas can be unified - substitutions only affect variables.
  - Example: Consider the pair P(F(x),A) and P(y,F(w))
  - This pair cannot be unified as there is no way of making A = F(w) with a substitution.
- In θ, each variable is paired at most once
- A variable's pairing term may not contain the variable directly or indirectly.
  - e.g. can't have substitution { x/F(y), y/F(x) }

- When unifying expressions P and R, the variable names in P and the variable names in R should be disjoint.

More on Unification

- Not all formulas can be unified - substitutions only affect variables.
  - Example: Consider the pair P(F(x),A) and P(y,F(w))
  - This pair cannot be unified as there is no way of making A = F(w) with a substitution.
- In θ, each variable is paired at most once
- A variable's pairing term may not contain the variable directly or indirectly.
  - e.g. can't have substitution { x/F(y), y/F(x) }

- When unifying expressions P and R, the variable names in P and the variable names in R should be disjoint.
  - Yes:UNIFY(Loves (John, x), Loves (y, Jane)) θ={ x/Jane, y/John }
  - No:UNIFY(Loves (John, x), Loves (x, Jane)) – No unifier

  - Solution: Standardizing apart the variables - come back to this after a few slides

Most General Unifier

- Our aim is to be able to match conflicting literals (for the use of resolution), even when they have variables. Unification process determines whether there is a"specialization" that matches.
- However, we don't want to over specialize.

Most General Unifier-example

- Consider the two sentences:

  $\neg P(x) \vee S(x) \vee Q(Noah)$

  P(y)$\vee$ R(y)

Most General Unifier-example

- Consider the two sentences:

  $\neg P(x) \vee S(x) \vee Q(Noah)$

  P(y)∨ R(y)

- Possible unifications:

  (S(Arvid)∨ Q(Noah)∨ R(Arvid)) {y=x, x=Arvid}

Most General Unifier-example

- Consider the two sentences:

  $\neg P(x) \vee S(x) \vee Q(Noah)$

  P(y)∨ R(y)

- Possible unifications:

  (S(Arvid)∨ Q(Noah)∨ R(Arvid)) {y=x, x=Arvid}

  (S(Sophie) ∨ Q(Noah)∨ R(Sophie)) {y=x, x=Sophie}

Most General Unifier-example

- Consider the two sentences:

  $\neg P(x) \vee S(x) \vee Q(Noah)$

  P(y)∨ R(y)

- Possible unifications:

  (S(Arvid)∨ Q(Noah)∨ R(Arvid)) {y=x, x=Arvid}

  (S(Sophie) ∨ Q(Noah)∨ R(Sophie)) {y=x, x=Sophie}

  (S(x)∨ Q(Noah)∨ R(x)) {y=x}

Most General Unifier-example

- Consider the two sentences:

  $\neg P(x) \vee S(x) \vee Q(Noah)$

  P(y)$\vee$ R(y)

- Possible unifications:

  (S(Arvid)$\vee$ Q(Noah)$\vee$ R(Arvid)) {y=x, x=Arvid}

  (S(Sophie) $\vee$ Q(Noah)$\vee$ R(Sophie)) {y=x, x=Sophie}

  (S(x)$\vee$ Q(Noah)$\vee$ R(x)) {y=x}

- The last unifier is the "most-general" one, the other two are specializations of it.
- We want to keep the most general clause so that we can use it in future resolution steps.

Most General Unifier

- Informally, the most general unifier (MGU) imposes the fewest constraints on the terms (contains the most variables).
- Formally, a substitution θ is more general than β iff there is a substitution σ such that $\theta\ \sigma = \beta$.
  e.g. $\theta =$ z/F(w) is more general than $\beta =$ z/F(C) since $\sigma =$ w/C

One more example on Most General Unifier

Consider clauses P(f(x),z) and P(y,A)

- $\theta_1 = \{$y= f(A), x=A, z=A$\}$ is a unifier.

  p(f(x),z)θ_1 = p(f(A), A)
  p(y,A)θ_1 = p(f(A), A) but it is not MGU.

One more example on Most General Unifier

Consider clauses P(f(x),z) and P(y,A)

- $\theta_1 = \{$y= f(A), x=A, z=A$\}$ is a unifier.

  p(f(x),z)θ_1 = p(f(A), A)
  p(y,A)θ_1 = p(f(A), A) but it is not MGU.

- $\theta_2 = \{$y=f(x), z=A$\}$ is an MGU.

  p(f(x),z)θ_2 = p(f(x), A)
  p(y,A)θ_2 = p(f(x), A)

Motivation for Standardizing apart

P	Q	θ
Knows(John,x)	Knows(John,Jane)	
Knows(John,x)	Knows(y,OJ)	
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,OJ)	

Standardizing apart

P	Q	θ
Knows(John,x)	Knows(John,Jane)	{x/Jane}
Knows(John,x)	Knows(y,OJ)	{x/OJ, y/John}
Knows(John,x)	Knows(y,Mother(y))	{x/Mother(John),y/John}
Knows(John,x)	Knows(x,OJ)	No substitution possible yet (i.e., "fails")

Standardizing apart

Knows (John, x) and Knows (x, OJ) cannot be unified, i.e, unifications *fails*.

- Intuitively we know that if we know John hates everyone he knows, and that everyone knows OJ. So we should be able to infer that John hates OJ.

Standardizing apart

Knows (John, x) and Knows (x, OJ) cannot be unified, i.e, unifications *fails*.

- Intuitively we know that if we know John hates everyone he knows, and that everyone knows OJ. So we should be able to infer that John hates OJ.
- This is why we require that every variable has a separate name.
- Need **Standardizing apart** that eliminates overlap of variables.

Standardizing apart

Knows (John, x) and Knows (x, OJ) cannot be unified, i.e, unifications *fails*.

- Intuitively we know that if we know John hates everyone he knows, and that everyone knows OJ. So we should be able to infer that John hates OJ.
- This is why we require that every variable has a separate name.
- Need **Standardizing apart** that eliminates overlap of variables.
- UNIFY $Knows(John, z_{27})$ and $Knows(z_{17}, OJ)$ This works!!! $\{z_{17}/John, z_{27}/OJ\}$

Resolution Refutation in FOL

- The idea is the same as in Propositional Logic:
  Our goal is to determine if KB $\models \alpha$:
  1. Add $\neg\alpha$ to the KB
  2. Convert KB and α to Conjunctive Normal Form
  3. Use the lifted/generalized resolution rule and search to determine whether the system is satisfiable (SAT)

Lifted Resolution Rule

Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where $\textsc{Unify}(\ell_i, \neg m_j) = \theta$.

Two standardized clauses can be resolved if they contain complementary literals (one is the negation of the other).

FOL literals are complementary if one *unifies* with the negation of the other.

Lifted Resolution is refutation-complete.

Resolution Refutation - example

Suppose we have the following knowledge in the KB:

The law says that it is a crime for an American to sell weapons to hostile nations.
The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

Resolution Refutation - example

Suppose we have the following knowledge in the KB:

The law says that it is a crime for an American to sell weapons to hostile nations.
The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

We want to prove that **"Colonel West is a criminal".**

Conversion to CNF - example

Let us first look at the procedure for conversion of FOL sentences to CNF through an example:

"Everyone who loves all animals is loved by someone"

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \ [\forall y \ \ Animal(y) \implies Loves(x,y)] \implies [\exists y \ \ Loves(y,x)]$$

1. Eliminate biconditionals and implications

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \ [\forall y \ \ Animal(y) \implies Loves(x, y)] \implies [\exists y \ \ Loves(y, x)]$$

1. Eliminate biconditionals and implications

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \ [\forall y \ \ Animal(y) \implies Loves(x, y)] \implies [\exists y \ \ Loves(y, x)]$$

1. Eliminate biconditionals and implications

$$\forall x \ \neg[\forall y \ \ \neg Animal(y) \vee Loves(x, y)] \vee [\exists y \ \ Loves(y, x)]$$

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \; [\forall y \;\; Animal(y) \implies Loves(x,y)] \implies [\exists y \;\; Loves(y,x)]$$

1. Eliminate biconditionals and implications

$$\forall x \;\; \neg[\forall y \;\; \neg Animal(y) \vee Loves(x,y)] \vee [\exists y \;\; Loves(y,x)]$$

2. Move $\neg$ inwards: $\neg\forall x, p \;\; \equiv \exists x \;\; \neg p$, $\;\; \neg\exists x, p \;\; \equiv \forall x \;\; \neg p$:

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \ [\forall y \ \ Animal(y) \implies Loves(x,y)] \implies [\exists y \ \ Loves(y,x)]$$

1. Eliminate biconditionals and implications

$$\forall x \ \neg[\forall y \ \neg Animal(y) \lor Loves(x,y)] \lor [\exists y \ \ Loves(y,x)]$$

2. Move $\neg$ inwards: $\neg\forall x, p \ \equiv \exists x \ \neg p$, $\ \neg\exists x, p \ \equiv \forall x \ \neg p$:

$$\forall x \ [\exists y \ \neg(\neg Animal(y) \lor Loves(x,y))] \lor [\exists y \ \ Loves(y,x)]$$

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \ [\forall y \ \ Animal(y) \implies Loves(x,y)] \implies [\exists y \ \ Loves(y,x)]$$

1. Eliminate biconditionals and implications

$$\forall x \ \neg[\forall y \ \ \neg Animal(y) \lor Loves(x,y)] \lor [\exists y \ \ Loves(y,x)]$$

2. Move $\neg$ inwards: $\neg\forall x, p \ \equiv \exists x \ \neg p$, $\neg\exists x, p \ \equiv \forall x \ \neg p$:

$$\forall x \ [\exists y \ \ \neg(\neg Animal(y) \lor Loves(x,y))] \lor [\exists y \ \ Loves(y,x)]$$
$$\forall x \ [\exists y \ \ \neg\neg Animal(y) \land \neg Loves(x,y)] \lor [\exists y \ \ Loves(y,x)]$$

Conversion to CNF - example

Everyone who loves all animals is loved by someone:

$$\forall x \; [\forall y \;\; Animal(y) \implies Loves(x, y)] \implies [\exists y \;\; Loves(y, x)]$$

1. Eliminate biconditionals and implications

$$\forall x \;\; \neg[\forall y \;\; \neg Animal(y) \lor Loves(x, y)] \lor [\exists y \;\; Loves(y, x)]$$

2. Move $\neg$ inwards: $\neg\forall x, p \;\; \equiv \exists x \;\; \neg p$, $\;\; \neg\exists x, p \;\; \equiv \forall x \;\; \neg p$:

$$\forall x \;\; [\exists y \;\; \neg(\neg Animal(y) \lor Loves(x, y))] \lor [\exists y \;\; Loves(y, x)]$$
$$\forall x \;\; [\exists y \;\; \neg\neg Animal(y) \land \neg Loves(x, y)] \lor [\exists y \;\; Loves(y, x)]$$
$$\forall x \;\; [\exists y \;\; Animal(y) \land \neg Loves(x, y)] \lor [\exists y \;\; Loves(y, x)]$$

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different variable name

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different variable name

$$\forall x \;\; [\exists y \;\; Animal(y) \land \neg Loves(x, y)] \lor [\exists z \;\; Loves(z, x)]$$

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different variable name

$$\forall x \;\; [\exists y \;\; Animal(y) \land \neg Loves(x, y)] \lor [\exists z \;\; Loves(z, x)]$$

4. Skolemize: Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different variable name

$$\forall x \;\; [\exists y \;\; Animal(y) \land \neg Loves(x,y)] \lor [\exists z \;\; Loves(z,x)]$$

4. Skolemize: Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

$$\forall x \;\; [Animal(F(x)) \land \neg Loves(x,F(x))] \lor Loves(G(x),x)$$

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different variable name

$$\forall x \ \ [\exists y \ \ Animal(y) \land \neg Loves(x, y)] \lor [\exists z \ \ Loves(z, x)]$$

4. Skolemize: Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

$$\forall x \ \ [Animal(F(x)) \land \neg Loves(x, F(x))] \lor Loves(G(x), x)$$

5. Drop universal quantifiers:

$$[Animal(F(x)) \land \neg Loves(x, F(x))] \lor Loves(G(x), x)$$

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different variable name

$$\forall x \;\; [\exists y \;\; Animal(y) \wedge \neg Loves(x, y)] \vee [\exists z \;\; Loves(z, x)]$$

4. Skolemize: Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

$$\forall x \;\; [Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$$

5. Drop universal quantifiers:

$$[Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$$

6. Distribute $\vee$ over $\wedge$:

$$[Animal(f(x)) \vee Loves(G(x), x)] \wedge [\neg Loves(x, F(x)) \vee Loves(G(x), x)]$$

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z\ American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \implies Criminal(x)$

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z\ American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono . . . has some missiles

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists\, x \;\; Owns(Nono, x) \land Missile(x)$:

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists\, x \; Owns(Nono, x) \wedge Missile(x)$:

S3: ... all of its missiles were sold to it by Colonel West

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z\ American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists x\ Owns(Nono, x) \wedge Missile(x)$:

S3: ... all of its missiles were sold to it by Colonel West
$\forall x\ Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono . . . has some missiles

$\exists\, x \;\, Owns(Nono, x) \wedge Missile(x)$:

S3: . . . all of its missiles were sold to it by Colonel West

$\forall\, x \;\, Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S4: Missiles are weapons.

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists \, x \;\; Owns(Nono, x) \wedge Missile(x)$:

S3: ... all of its missiles were sold to it by Colonel West
$\forall \, x \;\; Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S4: Missiles are weapons.

$\forall \, x \;\; Missile(x) \Rightarrow Weapon(x)$

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono . . . has some missiles
$\exists x \; Owns(Nono, x) \wedge Missile(x)$:

S3: . . . all of its missiles were sold to it by Colonel West
$\forall x \; Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S4: Missiles are weapons.

$\forall x \; Missile(x) \Rightarrow Weapon(x)$

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists\, x \; Owns(Nono, x) \wedge Missile(x)$:

S3: ... all of its missiles were sold to it by Colonel West
$\forall\, x \; Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S4: Missiles are weapons.

$\forall\, x \; Missile(x) \Rightarrow Weapon(x)$

S5: An enemy of America counts as "hostile":

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists \, x \; Owns(Nono, x) \wedge Missile(x)$:

S3: ... all of its missiles were sold to it by Colonel West
$\forall \, x \; Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S4: Missiles are weapons.

$\forall \, x \; Missile(x) \Rightarrow Weapon(x)$

S5: An enemy of America counts as "hostile":

Colonel West example using Resolution Refutation

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles
$\exists\, x \; Owns(Nono, x) \wedge Missile(x)$:

S3: ... all of its missiles were sold to it by Colonel West
$\forall\, x \; Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S4: Missiles are weapons.

$\forall\, x \; Missile(x) \Rightarrow Weapon(x)$

S5: An enemy of America counts as "hostile":

S6: West, who is American ...
S6: $American(West)$

S7: The country Nono, an enemy of America ...
S7: $Enemy(Nono, America)$

Colonel West example using Resolution Refutation

- KB:
  $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$
  $Owns(Nono, M_1)$ and $Missile(M_1)$
  $Missile(x) \Rightarrow Weapon(x)$
  $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
  $Enemy(x, America) \implies Hostile(x)$
  $American(West)$
  $Enemy(Nono, America)$

- Question: Is colonel West criminal?

Colonel West example in CNF form

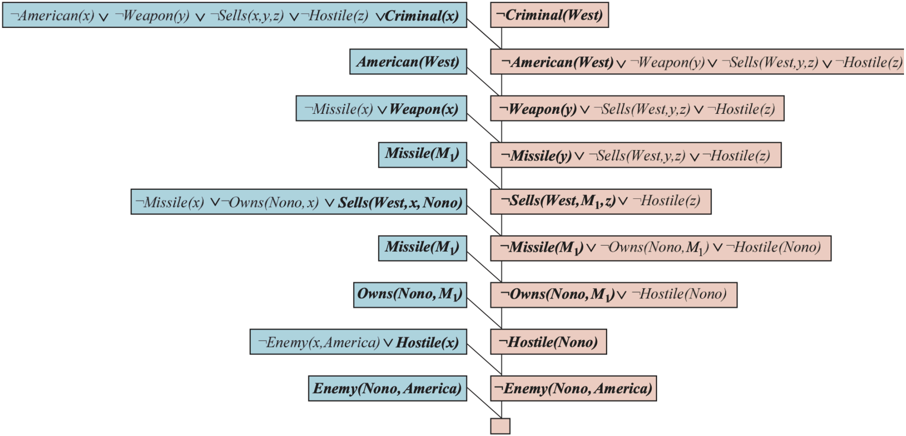
1. $\neg American(x) \vee \neg Weapon(y) \vee \neg Sells(x, y, z) \vee \neg Hostile(z) \vee Criminal(x)$
2. $\neg Missile(x) \vee \neg Owns(Nono, x) \vee Sells(West, x, Nono)$
3. $\neg Enemy(x, America) \vee Hostile(x)$
4. $\neg Missile(x) \vee Weapon(x)$

Colonel West example in CNF form

1. $\neg American(x) \vee \neg Weapon(y) \vee \neg Sells(x, y, z) \vee \neg Hostile(z) \vee Criminal(x)$
2. $\neg Missile(x) \vee \neg Owns(Nono, x) \vee Sells(West, x, Nono)$
3. $\neg Enemy(x, America) \vee Hostile(x)$
4. $\neg Missile(x) \vee Weapon(x)$

- Facts: $American(West), Owns(Nono, M1), Missile(M1), Enemy(Nono, America$
- Add $\neg Criminal(West)$

Colonel West example using Resolution Refutation - cont



Efficient Resolution Strategies

- Unit preference: prefer inferences where one of the sentences is a single literal - towards shorter clauses, finaly empty one
- Set of support: start with the negated query, and add resolvents into the set. In each inference use one of the sentences from this set - search space is reduced.
- Input resolution: In each inference use one of the sentences in the original KB or query, and one other sentence (i.e., a resolvent).

Forward chaining algorithm

- Employs (repeatedly) generalized Modus ponens - which is sound
- Continue matching and deriving consequences of implication sentences until deriving the goal.

- FC is complete for definite clause KBs without functions - i.e. finds an answer to each query of which answer is entailed by the KB

Inference through Forward Chaining in FOL

- Forward Chaining is an important inference in FOL - without propositionalisation
- FC uses definite clauses.
  - A definite clause: either atomic, or implication
  - Existential quantifiers are not allowed
  - Universal quantifications are implicit.

Inference through Forward Chaining in FOL

- Forward Chaining is an important inference in FOL - without propositionalisation
- FC uses definite clauses.
  - A definite clause: either atomic, or implication
  - Existential quantifiers are not allowed
  - Universal quantifications are implicit.
  - Example sentences in DC:

    King (John) - literal
    King(x) $\rightarrow$ Evil(x)
    Evil(x). (i.e., everyone is evil). - literal with variable

Forward Chaining - Colonel West example

- KB: The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

  We want to prove that **"Colonel West is a criminal".**
- We need
  - to translate these natural language sentences to FOL

Forward Chaining - Colonel West example

- KB: The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

  We want to prove that **"Colonel West is a criminal".**
- We need
  - to translate these natural language sentences to FOL
  - convert these sentences in the KB to Definite Clauses if they are not in DC form by eliminating quantifiers:

Forward Chaining - Colonel West example

- KB: The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

  We want to prove that **"Colonel West is a criminal".**
- We need
  - to translate these natural language sentences to FOL
  - convert these sentences in the KB to Definite Clauses if they are not in DC form by eliminating quantifiers:
    - remove $\forall$
    - eliminate $\exists$ by skolemization

Forward Chaining - Colonel West example

S1: It is a crime for an American to sell weapons to hostile nations:

Forward Chaining - Colonel West example

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z\ American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

Forward Chaining - Colonel West example

S1: It is a crime for an American to sell weapons to hostile nations:

$\forall x, y, z \; American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S1: $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

Forward chaining - Colonel West

... It is a crime for an American to sell weapons to hostile nations:

S1: $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles

Forward chaining - Colonel West

... It is a crime for an American to sell weapons to hostile nations:

S1: $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$

S2: Nono ... has some missiles

$\exists x \; Owns(Nono, x) \wedge Missile(x)$:

Forward chaining - Colonel West

. . . It is a crime for an American to sell weapons to hostile nations:

S1: $American(x) \land Weapon(y) \land Sells(x, y, z) \land Hostile(z) \implies Criminal(x)$

S2: Nono . . . has some missiles

$\exists\, x \; Owns(Nono, x) \land Missile(x)$:

S2: $Owns(Nono, M_1)$ and $Missile(M_1)$

Forward chaining - Colonel West

S3: . . . all of its missiles were sold to it by Colonel West

Forward chaining - Colonel West

S3: . . . all of its missiles were sold to it by Colonel West

$$\forall x\ Missile(x) \land Owns(Nono, x) \implies Sells(West, x, Nono)$$

Forward chaining - Colonel West

S3: . . . all of its missiles were sold to it by Colonel West

$\forall x \; Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

S3: $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$

Forward chaining - Colonel West

S4: Missiles are weapons.

Forward chaining - Colonel West

S4: Missiles are weapons.

$\forall x \; Missile(x) \Rightarrow Weapon(x)$

Forward chaining - Colonel West

S4: Missiles are weapons.

$\forall x \; Missile(x) \Rightarrow Weapon(x)$

S4: $Missile(x) \Rightarrow Weapon(x)$

S5: An enemy of America counts as "hostile":

Forward chaining - Colonel West

S4: Missiles are weapons.

$\forall x \; Missile(x) \Rightarrow Weapon(x)$

S4: $Missile(x) \Rightarrow Weapon(x)$

S5: An enemy of America counts as "hostile":

$\forall x \; Enemy(x, America) \implies Hostile(x)$

Forward chaining - Colonel West

S4: Missiles are weapons.

$\forall x \; Missile(x) \Rightarrow Weapon(x)$

S4: $Missile(x) \Rightarrow Weapon(x)$

S5: An enemy of America counts as "hostile":

$\forall x \; Enemy(x, America) \implies Hostile(x)$

S5: $Enemy(x, America) \implies Hostile(x)$

Forward chaining - Colonel West

S6: West, who is American ...

S6: $American(West)$

S7: The country Nono, an enemy of America ...

S7: $Enemy(Nono, America)$

Forward chaining - Colonel West

Our KB:

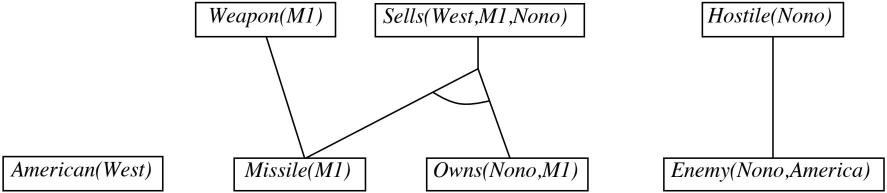
1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$
2. $Owns(Nono, M_1)$
3. $Missile(M_1)$
4. $Missile(x) \Rightarrow Weapon(x)$
5. $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
6. $Enemy(x, America) \implies Hostile(x)$
7. $American(West)$
8. $Enemy(Nono, America)$

Forward chaining proof of Colonel West

$\boxed{American(West)}$ $\boxed{Missile(M1)}$ $\boxed{Owns(Nono,M1)}$ $\boxed{Enemy(Nono,America)}$

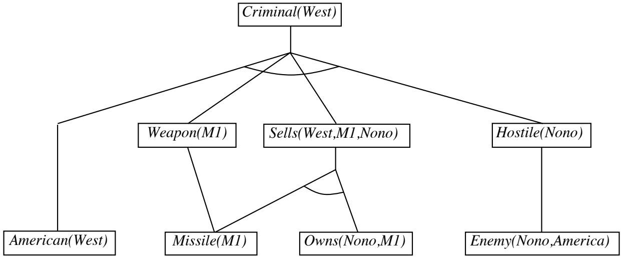
1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$
2. $Owns(Nono, M_1)$
3. $Missile(M_1)$
4. $Missile(x) \Rightarrow Weapon(x)$
5. $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
6. $Enemy(x, America) \implies Hostile(x)$
7. $American(West)$
8. $Enemy(Nono, America)$

Forward chaining proof



1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$
2. $Owns(Nono, M_1)$
3. $Missile(M_1)$
4. $Missile(x) \Rightarrow Weapon(x)$
5. $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
6. $Enemy(x, America) \implies Hostile(x)$
7. $American(West)$
8. $Enemy(Nono, America)$

Forward chaining proof



1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$
2. $Owns(Nono, M_1)$
3. $Missile(M_1)$
4. $Missile(x) \Rightarrow Weapon(x)$
5. $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
6. $Enemy(x, America) \implies Hostile(x)$
7. $American(West)$
8. $Enemy(Nono, America)$

Forward Chaining and efficiency

- checks every rule against every fact

  $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
  Heuristic: Order the conjuncts to be checked in the premise. Check the one with fewest fact sentence in the KB first - reminds the *Minimum remaining values* heuristic in CSP.
- rechecks every rule in every iteration

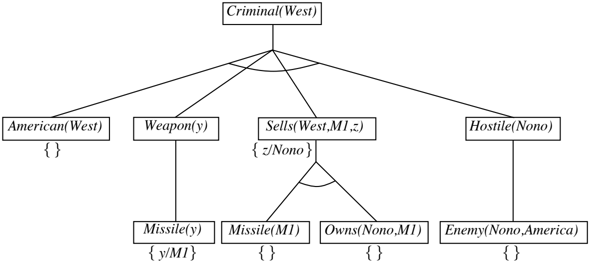
  e.g., eliminate redundant rule-matching through incremental FC: Check a rule at iteration t if only its premise includes a conjunct P1 that unifies with a fact $P1'$ inferred at iteration t-1
- can generate many facts irrelevant to the goal

  e.g., do Backward Chaining instead

Backward chaining example

- Employs dept-first search
- BC is incomplete as it may have the problem of repeated states.

Backward chaining example



1. $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \implies Criminal(x)$
2. $Owns(Nono, M_1)$
3. $Missile(M_1)$
4. $Missile(x) \Rightarrow Weapon(x)$
5. $Missile(x) \wedge Owns(Nono, x) \implies Sells(West, x, Nono)$
6. $Enemy(x, America) \implies Hostile(x)$
7. $American(West)$

More on Skolemization -Examples

- $\exists$ x Sibling (sofie, x)

  Skolemized: Sibling(Sofie,SkSister)

- $\forall$ x $\exists y$ Parent(x,y)

  Skolemized: $\forall$ x Parent(x, F(x))

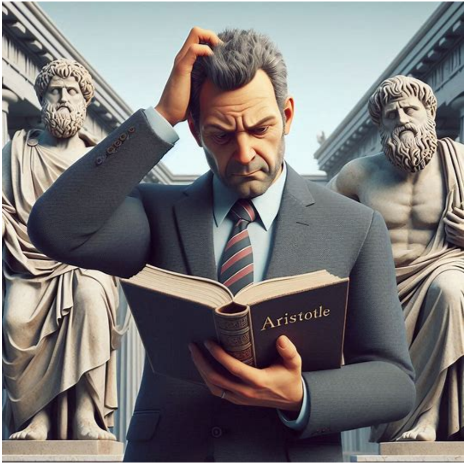
- $\forall$ x,y Grandpa(x,y) $\implies$ $\exists$ z Parent (x,z) $\wedge$ Parent(z,y)

  Skolemized: $\forall$ x,y Grandpa(x,y) $\implies$ Parent(x, F(x,y)) $\wedge$ Parent(F(x,y), y)

- $\forall$ x [$\exists y$ Animal(y) $\wedge \neg$ Loves(x,y)] $\vee$ [$\exists$ z Loves(z,x)]

  Skolemized: $\forall$ x [Animal(F(x)) $\wedge$ $\neg$ Loves(x, F(x))] $\vee$ Loves (G(x), x)

Logic for Politicians



Propositional Logic

The Knowledge Base

$(D \wedge G) \implies (I \vee W)$

$(W \vee C) \implies R$

$I \implies C$

$D \vee R$

The Knowledge Base in Conjunctive Normal Form

$\neg D \vee \neg G \vee I \vee W$

$\neg W \vee R$

$\neg C \vee R$

$\neg I \vee C$

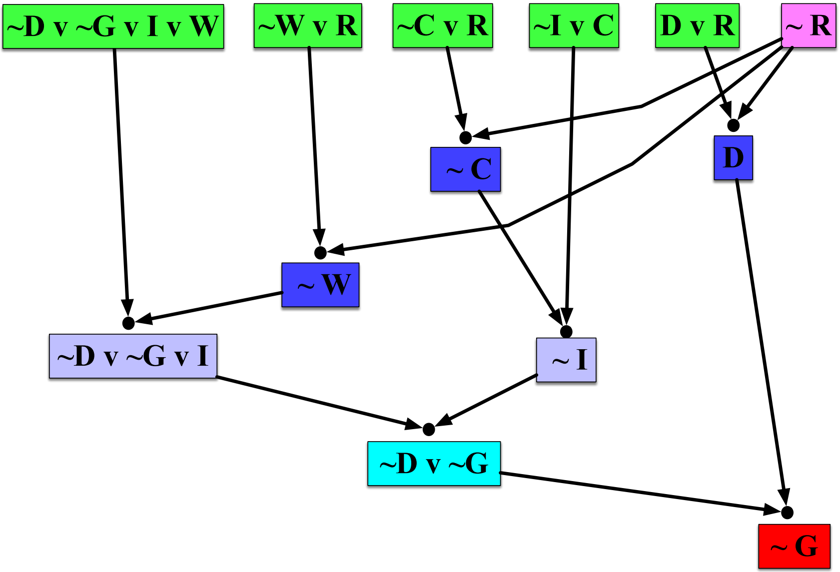
$D \vee R$

Problem

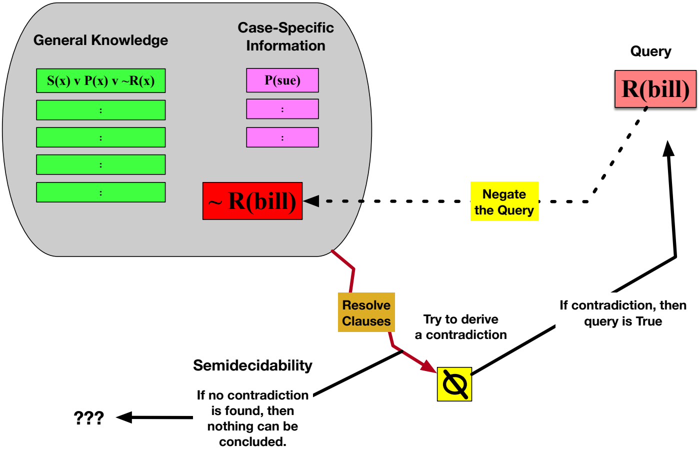
Given: $\neg R$

Prove: $\neg G$

Resolving Clauses in Propositional Logic



The Basic Resolution Algorithm

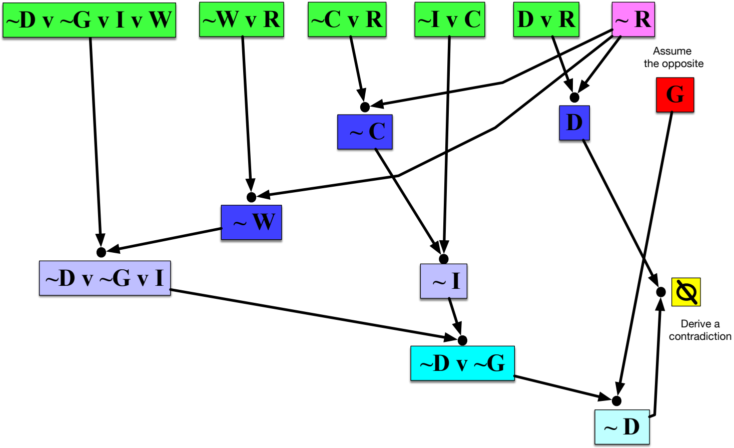


* Propositional logic is decidable (Just use model checking), but first-order logic is only semidecidable ⇒ In FOL, failure to find a contradiction does NOT prove that the query is False, but any sound and complete inference procedure (such as resolution) will eventually prove a true query.

The Resolution Algorithm for Propositional Logic

To Prove: $\neg G$

Assume: $\neg\neg G \equiv G$



Conjunctive Normal Form for FOL

A candidate wins a debate only if they appear wise on one issue and their opponent appears confused on another issue.

$\forall x, y : [Debate(x, y) \land Wins(x, y)] \implies [\exists w : Wise(x, w) \land \exists w : Confused(y, w)]$

1. Eliminate implications

   $\forall x, y : \neg[Debate(x, y) \land Wins(x, y)] \lor [\exists w : Wise(x, w) \land \exists w : Confused(y, w)]$
2. Move $\neg$ inwards.

   $\forall x, y : \neg Debate(x, y) \lor \neg Wins(x, y) \lor [\exists w : Wise(x, w) \land \exists w : Confused(y, w)]$
3. Standardize Variables

   $\forall x, y : \neg Debate(x, y) \lor \neg Wins(x, y) \lor [\exists w : Wise(x, w) \land \exists z : Confused(y, z)]$
4. Skolemize (Drop existential quantifiers and replace their variables with constants or functions of the scoping universally-quantified variables.)

   $\forall x, y : \neg Debate(x, y) \lor \neg Wins(x, y) \lor [Wise(x, F(x, y)) \land Confused(y, G(x, y))]$
5. Drop Universal Quantifiers

   $\neg Debate(x, y) \lor \neg Wins(x, y) \lor [Wise(x, F(x, y)) \land Confused(y, G(x, y))]$
6. Distribute $\lor$ over $\land$

   $[\neg Debate(x, y) \lor \neg Wins(x, y) \lor Wise(x, F(x, y))] \land [\neg Debate(x, y) \lor \neg Wins(x, y) \lor Confused(y, G(x, y)]$

FOL Conjunctive Normal Form: Exercise

There is at least one problem that any pair of candidates agree is important, but they never agree on any of its solutions.

$\exists w : Problem(w) \land \forall x, y : [Candidate(x) \land Candidate(y)] \implies [Agree(x, y, w) \land [\forall z : Solution(z, w) \implies \neg Agree(x, y, z)]]$

$\exists w : P(w) \land \forall x, y : [C(x) \land C(y)] \implies [A(x, y, w) \land [\forall z : S(z, w) \implies \neg A(x, y, z)]]$

1. Eliminate implications
2. Move $\neg$ inwards.
3. Standardize Variables
4. Skolemize (Drop existential quantifiers and replace their variables with constants or functions of the scoping universally-quantified variables.)
5. Drop Universal Quantifiers
6. Distribute $\lor$ over $\land$

First-Order Logic (FOL) for Politics

The Knowledge Base

$$\forall x : (Democrat(x) \wedge Generous(x)) \implies (Idealist(x) \vee Wealthy(x))$$

$$\forall x : (Wealthy(x) \vee Credulous(x)) \implies Republican(x)$$

$$\forall x : Idealist(x) \implies Credulous(x)$$

$$\forall x : Democrat(x) \vee Republican(x)$$

The Knowledge Base in Conjunctive Normal Form

Note: variables renamed to avoid conflicts

$$\neg Democrat(x) \vee \neg Generous(x) \vee Idealist(x) \vee Wealthy(x)$$

$$\neg Wealthy(y) \vee Republican(y)$$

$$\neg Credulous(z) \vee Republican(z)$$

$$\neg Idealist(w) \vee Credulous(w)$$

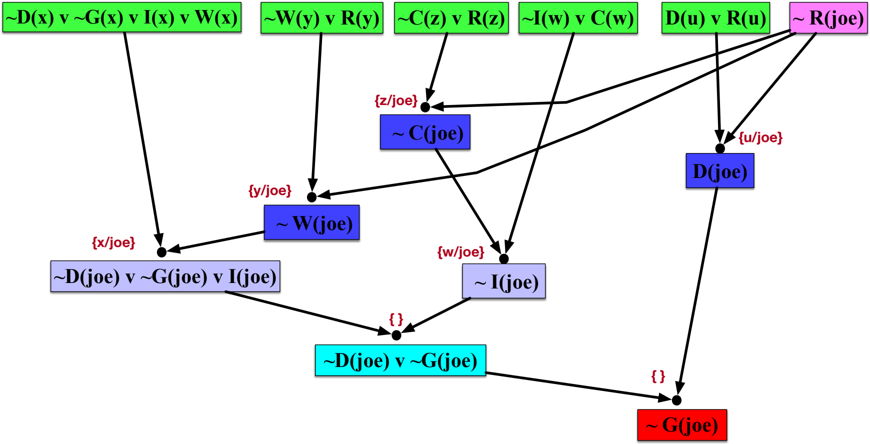
$$Democrat(u) \vee Republican(u)$$

Problem

Given: $\neg Republican(joe)$

Prove: $\neg Generous(joe)$

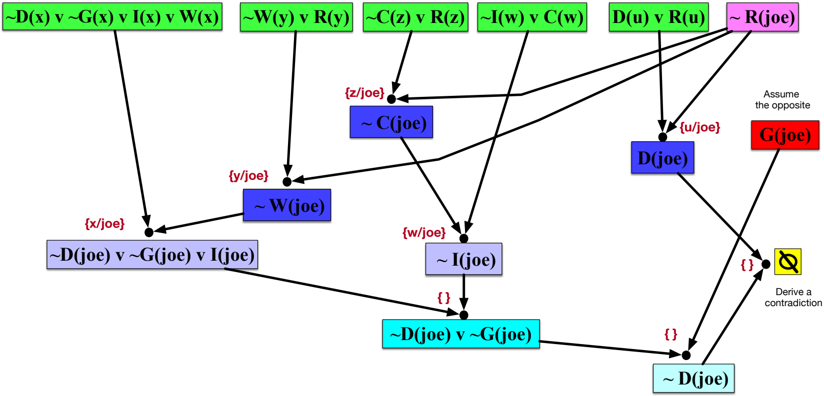
Resolving Clauses for Political FOL



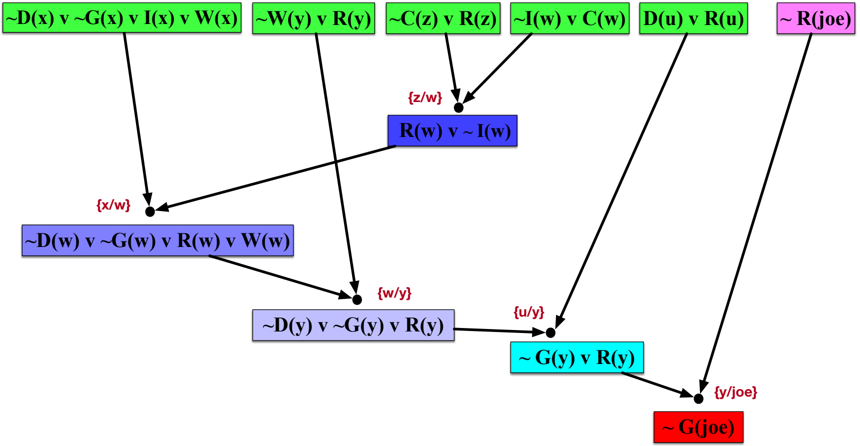
The Resolution Algorithm for Political FOL

To Prove: $\neg G(joe)$

Assume: $\neg\neg G(joe) \equiv G(joe)$



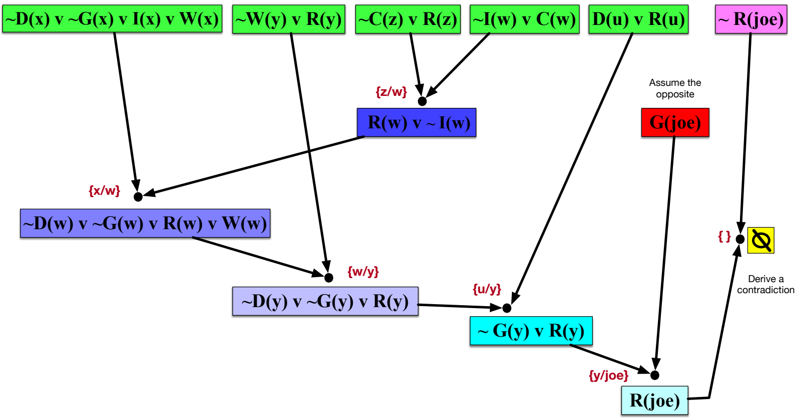
An Alternate Resolution Sequence



An Alternate Run of the Resolution Algorithm

To Prove: $\neg G(joe)$

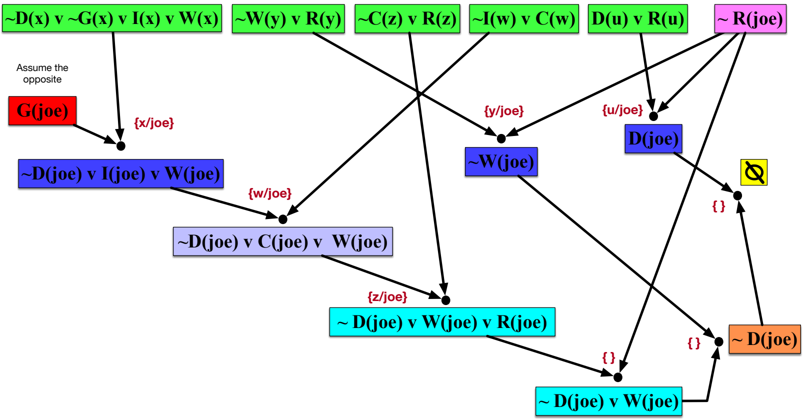
Assume: $\neg\neg G(joe) \equiv G(joe)$



...and Another Alternate

To Prove: $\neg G(joe)$

Assume: $\neg\neg G(joe) \equiv G(joe)$



FOL with Existential Quantification

The Knowledge Base

$\forall x : (Democrat(x) \wedge Generous(x)) \implies (Idealist(x) \vee Wealthy(x))$

$\forall x : (Wealthy(x) \vee Credulous(x)) \implies Republican(x)$

$\forall x : Idealist(x) \implies Credulous(x)$

$\forall x : Democrat(x) \vee Republican(x)$

$\forall x : Generous(x) \iff \exists y : Supports(x, y)$

The Knowledge Base in Conjunctive Normal Form

Note: variables renamed to avoid conflicts

$\neg Democrat(x) \vee \neg Generous(x) \vee Idealist(x) \vee Wealthy(x)$

$\neg Wealthy(y) \vee Republican(y)$

$\neg Credulous(z) \vee Republican(z)$

$\neg Idealist(w) \vee Credulous(w)$

$Democrat(u) \vee Republican(u)$

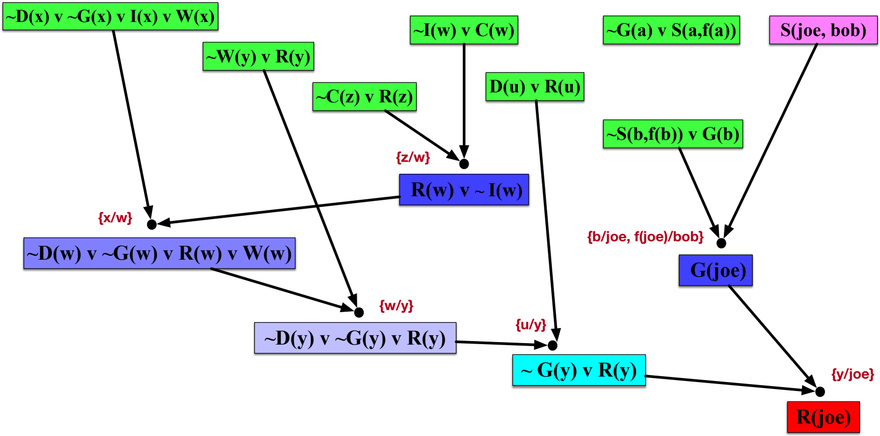
$\neg Generous(a) \vee Supports(a, f(a))$ f() = skolem function

$\neg Supports(b, f(b)) \vee Generous(b)$

Resolution with Existential Quantification

Given: *Supports*(*joe*, *bob*)

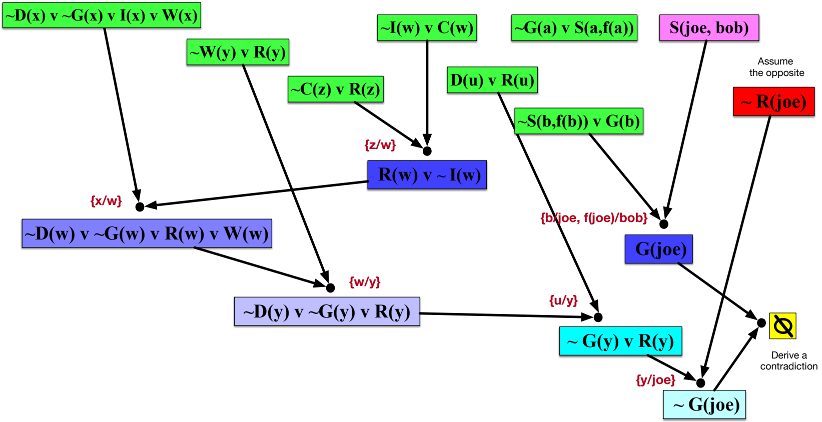
Prove: *Republican*(*joe*)



Resolution Algorithm with Existential Quantification

To Prove: *Republican(joe)*

Assume: $\neg$*Republican(joe)*



Forward Chaining Politics

The previous kwg base is not amenable to forward chaining, because:

- It involves an existential quantifier, but forward-chaining only deals with universal quantification.
- Several of the disjunctions are not *definite clauses*, since they have more than one positive literal.
- Let's try a different knowledge base.

Kwg Base

$\forall x, y : Democrat(x) \land Tax(y) \implies Likes(x, y)$

$\forall x, y, z : Likes(x, z) \land Likes(y, z) \implies Likes(x, y)$

$\forall x, y, z : Republican(x) \land Republican(y) \land Democrat(z) \land Likes(y, z)$

$\implies Dislikes(x, y)$

$Democrat(kamala)$

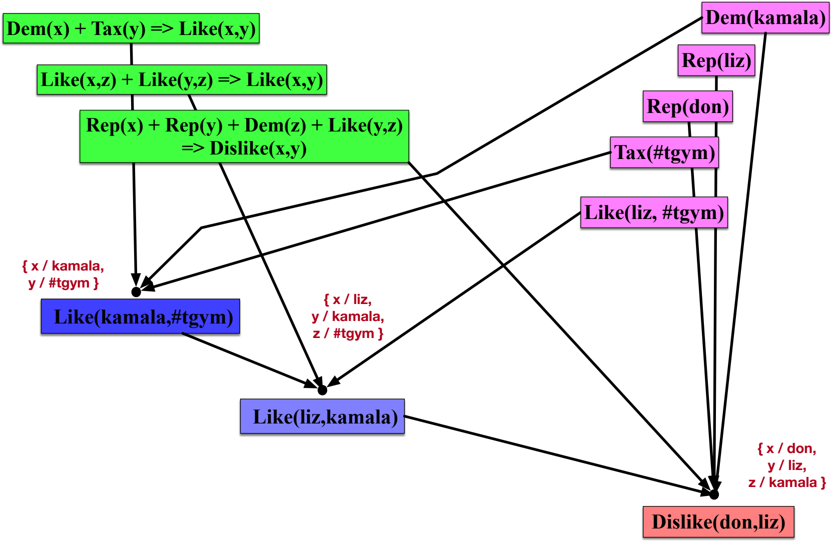
$Republican(liz)$

$Republican(don)$

$Tax(\#tgym)$ $\#$tgym = training studio tax

$Likes(Liz, \#tgym)$

The Logic of Polarization



Appendix: CNF Exercise Solution

There is at least one problem that any pair of candidates agree is important, but they never agree on any of its solutions.

$\exists w : Problem(w) \land \forall x, y : [Candidate(x) \land Candidate(y)] \implies [Agree(x, y, w) \land [\forall z : Solution(z, w) \implies \neg Agree(x, y, z)]]$

$\exists w : P(w) \land \forall x, y : [C(x) \land C(y)] \implies [A(x, y, w) \land [\forall z : S(z, w) \implies \neg A(x, y, z)]]$

1. Eliminate implications
   $\exists w : P(w) \land \forall x, y : \neg[C(x) \land C(y)] \lor [A(x, y, w) \land [\forall z : \neg S(z, w) \lor \neg A(x, y, z)]]$
2. Move $\neg$ inwards.
   $\exists w : P(w) \land \forall x, y : \neg C(x) \lor \neg C(y) \lor [A(x, y, w) \land [\forall z : \neg S(z, w) \lor \neg A(x, y, z)]]$
3. Standardize Variables (not necessary here)
   $\exists w : P(w) \land \forall x, y : \neg C(x) \lor \neg C(y) \lor [A(x, y, w) \land [\forall z : \neg S(z, w) \lor \neg A(x, y, z)]]$
4. Skolemize (K = skolem constant)
   $P(K) \land \forall x, y : \neg C(x) \lor \neg C(y) \lor [A(x, y, K) \land [\forall z : \neg S(z, K) \lor \neg A(x, y, z)]]$
5. Drop Universal Quantifiers
   $P(K) \land [\neg C(x) \lor \neg C(y) \lor [A(x, y, K) \land [\neg S(z, K) \lor \neg A(x, y, z)]]$
6. Distribute $\lor$ over $\land$ (Yielding 3 conjuncts)
   $P(K) \land$
   $[\neg C(x) \lor \neg C(y) \lor A(x, y, K)] \land$
   $[\neg C(x) \lor \neg C(y) \lor \neg S(z, K) \lor \neg A(x, y, z)]$

* Note: No skolem **function** was needed, since w was not **declared** within the scope of any universal variables, though it was used within their scope.

Summary

Let us summarize together!