First-Order Logic

First-Order Logic

- Propositional logic only deals with "facts", statements that may or may not be true of the world, e.g., "It is raining". But, one cannot have variables that stand for books or tables.

First-Order Logic

- Propositional logic only deals with "facts", statements that may or may not be true of the world, e.g., "It is raining". But, one cannot have variables that stand for books or tables.
- In first-order logic, variables refer to things in the world and, furthermore, you can quantify over them: talk about all of them or some of them without having to name them explicitly.

FOL motivation

- Statements that cannot be made in propositional logic but can be made in FOL

FOL motivation

Statements that cannot be made in propositional logic but can be made in FOL

- When you paint a block with green paint, it becomes green.
  - In propositional logic, one would need a statement about every single block, one cannot make the general statement about all blocks.

FOL motivation

Statements that cannot be made in propositional logic but can be made in FOL

- When you paint a block with green paint, it becomes green.
  - In propositional logic, one would need a statement about every single block, one cannot make the general statement about all blocks.
- When you sterilize a jar, all the bacteria are dead.
  - In FOL, we can talk about all the bacteria without naming them explicitly.

FOL motivation

Statements that cannot be made in propositional logic but can be made in FOL

- When you paint a block with green paint, it becomes green.
  - In propositional logic, one would need a statement about every single block, one cannot make the general statement about all blocks.
- When you sterilize a jar, all the bacteria are dead.
  - In FOL, we can talk about all the bacteria without naming them explicitly.
- A person is allowed access to this Web site if they have been formally authorized or they are known to someone who has access.

FOL syntax

FOL syntax

- Term

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39
  - Variables: x, y, a

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39
  - Variables: x, y, a
  - Function symbol applied to one or more terms: f(x), f(f(x)), mother-of(John)

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39
  - Variables: x, y, a
  - Function symbol applied to one or more terms: f(x), f(f(x)), mother-of(John)
- Sentence
  - A predicate symbol applied to zero or more terms: On(a,b), Sister(Jane, Joan), Sister(mother-of(John), Jane)

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39
  - Variables: x, y, a
  - Function symbol applied to one or more terms: f(x), f(f(x)), mother-of(John)
- Sentence
  - A predicate symbol applied to zero or more terms: On(a,b), Sister(Jane, Joan), Sister(mother-of(John), Jane)
  - $t_1=t_2$

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39
  - Variables: x, y, a
  - Function symbol applied to one or more terms: f(x), f(f(x)), mother-of(John)
- Sentence
  - A predicate symbol applied to zero or more terms: On(a,b), Sister(Jane, Joan), Sister(mother-of(John), Jane)
  - $t_1=t_2$
  - If v is a variable and Φ is a sentence, then $\forall v.\Phi$ and $\exists v.\Phi$ are sentences.

FOL syntax

- Term
  - Constant symbols: Fred, Japan, Bacterium39
  - Variables: x, y, a
  - Function symbol applied to one or more terms: f(x), f(f(x)), mother-of(John)
- Sentence
  - A predicate symbol applied to zero or more terms: On(a,b), Sister(Jane, Joan), Sister(mother-of(John), Jane)
  - $t_1 = t_2$
  - If v is a variable and Φ is a sentence, then $\forall v.\Phi$ and $\exists v.\Phi$ are sentences.
  - Closure under sentential operators: $\wedge \vee \rightarrow \neg$ ( )

FOL Interpretations

- Interpretation I

FOL Interpretations

- Interpretation I
  - U set of objects
    (called "domain of discourse" or "universe")

FOL Interpretations

- Interpretation I
  - U set of objects
    (called "domain of discourse" or "universe")
  - Maps constant symbols to elements of U

FOL Interpretations

- Interpretation I
  - U set of objects
    (called "domain of discourse" or "universe")
  - Maps constant symbols to elements of U
  - Maps predicate symbols to relations on U
    (binary relation is a set of pairs)

FOL Interpretations

- Interpretation I
  - U set of objects
    (called "domain of discourse" or "universe")
  - Maps constant symbols to elements of U
  - Maps predicate symbols to relations on U
    (binary relation is a set of pairs)
  - Maps function symbols to functions on U
    (function is a binary relation with a single pair for each element in U, whose first item is that element)

Denotation of Terms

Terms name objects in U

Denotation of Terms

Terms name objects in U

- I(Fred)   if Fred is constant, then given

Denotation of Terms

Terms name objects in U

- I(Fred) if Fred is constant, then given
- I(x) if x is a variable, then undefined

Denotation of Terms

Terms name objects in U

- I(Fred) if Fred is constant, then given
- I(x) if x is a variable, then undefined
- $I(f(t_1, \ldots, t_n))$ $I(f)(I(t_1), \ldots, I(t_n))$

Holds

When does a sentence hold in an interpretation?

Holds

When does a sentence hold in an interpretation?

- P is a relation symbol
- $t_1, \ldots, t_n$ are terms

holds($P(t_1, \ldots, t_n)$, I) iff $\langle I(t_1), \ldots, I(t_n)\rangle \in I(P)$

Holds

When does a sentence hold in an interpretation?

- P is a relation symbol
- $t_1, \ldots, t_n$ are terms

holds($P(t_1, \ldots, t_n)$, I) iff $\langle I(t_1), \ldots, I(t_n)\rangle \in I(P)$

Brother(Jon, Joe)??

Holds

When does a sentence hold in an interpretation?

- P is a relation symbol
- $t_1, \ldots, t_n$ are terms

holds($P(t_1, \ldots, t_n)$, I) iff $\langle I(t_1), \ldots, I(t_n) \rangle \in I(P)$

Brother(Jon, Joe)??

- I(Jon) =  [an element of U]

Holds

When does a sentence hold in an interpretation?

- P is a relation symbol
- $t_1, \ldots, t_n$ are terms

holds($P(t_1, \ldots, t_n)$, I) iff $\langle I(t_1), \ldots, I(t_n) \rangle \in I(P)$

Brother(Jon, Joe)??

- I(Jon) =  [an element of U]
- I(Joe) =  [an element of U]

Holds

When does a sentence hold in an interpretation?

- P is a relation symbol
- $t_1, \ldots, t_n$ are terms

holds($P(t_1, \ldots, t_n)$, I) iff $<I(t_1), \ldots, I(t_n)> \in I(P)$

Brother(Jon, Joe)??

- I(Jon) =  [an element of U]
- I(Joe) =  [an element of U]
- I(Brother) = {<>,<>, <...,...>, ... }

Holds

When does a sentence hold in an interpretation?

- P is a relation symbol
- $t_1, \ldots, t_n$ are terms

holds($P(t_1, \ldots, t_n)$, I) iff $<I(t_1), \ldots, I(t_n)> \in I(P)$

Brother(Jon, Joe)??

- I(Jon) =  [an element of U]
- I(Joe) =  [an element of U]
- I(Brother) = {< ,  >, <   >, <…,…>, … }

- holds(Brother(Jon, Joe), I)

Equality

holds($t_1 = t_2$, I) iff I(t_1) is the same object as I(t_2)

Equality

holds($t_1 = t_2$, I) iff I(t_1) is the same object as I(t_2)

Jon = Jack ?

- I(Jon) =  [an element of U]
- I(Jack) =  [an element of U]

- holds(Jon = Jack, I)

Semantics of Quantifiers

Semantics of Quantifiers

Extend an interpretation I to bind variable x to element $a \in U$: $I_{x/a}$

Semantics of Quantifiers

Extend an interpretation I to bind variable x to element $a \in U$: $I_{x/a}$

- holds($\forall x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for all $a \in U$

Semantics of Quantifiers

Extend an interpretation I to bind variable x to element $a \in U$: $I_{x/a}$

- holds($\forall x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for all $a \in U$
- holds($\exists x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for some $a \in U$

Semantics of Quantifiers

Extend an interpretation I to bind variable x to element $a \in U$: $I_{x/a}$

- holds($\forall x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for all $a \in U$
- holds($\exists x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for some $a \in U$

Quantifier applies to formula to right until an enclosing right parenthesis:

Semantics of Quantifiers

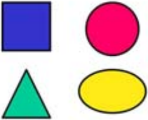
Extend an interpretation I to bind variable x to element $a \in U$: $I_{x/a}$

- holds($\forall x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for all $a \in U$
- holds($\exists x.\Phi$, I) iff holds(Φ, $I_{x/a}$) for some $a \in U$

Quantifier applies to formula to right until an enclosing right parenthesis:

$$(\forall x.P(x) \vee Q(x)) \wedge \exists x.R(x) \rightarrow Q(x)$$

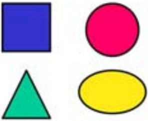
FOL Example Domain



The Real World

FOL Example Domain

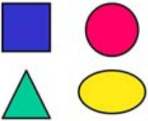
- U = {}



The Real World

FOL Example Domain

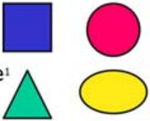
- U = {}
- Constants: Fred



The Real World

FOL Example Domain

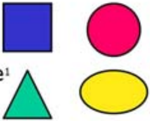
- U = {  }
- Constants: Fred
- Preds: Above^2, Circle^1, Oval^1, Square^1



The Real World

FOL Example Domain

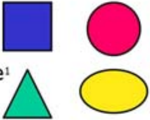
- U = {}
- Constants: Fred
- Preds: Above^2, Circle^1, Oval^1, Square^1
- Function: hat



The Real World

FOL Example Domain

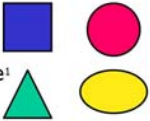
- U = {■, ▲, ●, ⬬}
- Constants: Fred
- Preds: Above^2, Circle^1, Oval^1, Square^1
- Function: hat
- I(Fred) = ▲



The Real World

FOL Example Domain

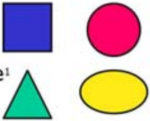
- U = {■, ▲, ●, ⬬}
- Constants: Fred
- Preds: Above^2, Circle^1, Oval^1, Square^1
- Function: hat
- I(Fred) = ▲
- I(Above) = {<■ ,▲ >,<● ,⬬>}



The Real World

FOL Example Domain

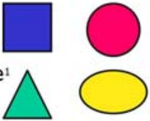
- U = {■, ▲, ●, ⬬}
- Constants: Fred
- Preds: Above^2, Circle^1, Oval^1, Square^1
- Function: hat
- I(Fred) = ▲
- I(Above) = {<■, ▲>, <●, ⬬>}
- I(Circle) = {<●>}



The Real World

FOL Example Domain

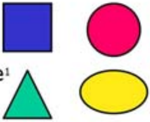
- U = {■, ▲, ●, ⬭}
- Constants: Fred
- Preds: Above[2], Circle[1], Oval[1], Square[1]
- Function: hat
- I(Fred) = ▲
- I(Above) = {<■, ▲>, <●, ⬭>}
- I(Circle) = {<●>}
- I(Oval) = {<●>, <⬭>}



The Real World

FOL Example Domain

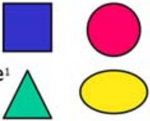
- U = {■, ▲, ●, ⬭}
- Constants: Fred
- Preds: Above², Circle¹, Oval¹, Square¹
- Function: hat
- I(Fred) = ▲
- I(Above) = {<■ ,▲ >,<● ,⬭>}
- I(Circle) = {<● >}
- I(Oval) = {<● >,<⬭>}
- I(hat) = {<▲ ,■ >,<⬭,● >,< ■,■ >,<● ,● >}



The Real
World

FOL Example Domain

- U = {■, ▲, ●, ⬭}
- Constants: Fred
- Preds: Above^2, Circle^1, Oval^1, Square^1
- Function: hat
- I(Fred) = ▲
- I(Above) = {<■, ▲>, <●, ⬭>}
- I(Circle) = {<●>}
- I(Oval) = {<●>, <⬭>}
- I(hat) = {<▲, ■>, <⬭, ●>, <■, ■>, <●, ●>}
- I(Square) = {<▲>}



The Real World

FOL Example

- holds(Square(Fred), I) ?

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬭>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬭>}
- I(hat) = {<▲,■>,<⬭,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

FOL Example

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬭>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬭>}
- I(hat) = {<▲,■>,<⬭,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(Square(Fred), I) ?
  yes
- holds(Above(Fred, hat(Fred)), I) ?

FOL Example

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●> <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(Square(Fred), I) ?

  yes
- holds(Above(Fred, hat(Fred)), I) ?
  - I(hat(Fred)) = ■

FOL Example

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(Square(Fred), I) ?
  yes
- holds(Above(Fred, hat(Fred)), I) ?
  - I(hat(Fred)) = ■
  - holds(Above(▲, ■), I) ?

FOL Example

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬭>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬭>}
- I(hat) = {<▲,■>,<⬭,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(Square(Fred), I) ?
  yes
- holds(Above(Fred, hat(Fred)), I) ? no
  - I(hat(Fred)) = ■
  - holds(Above(▲, ■), I) ? no

FOL Example

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(Square(Fred), I) ?
  yes
- holds(Above(Fred, hat(Fred)), I) ? no
  - I(hat(Fred)) = ■
  - holds(Above(▲, ■), I) ? no
- holds(∃x. Oval(x), I) ?

FOL Example

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(Square(Fred), I) ?
  yes
- holds(Above(Fred, hat(Fred)), I) ? no
  - I(hat(Fred)) = ■
  - holds(Above(▲, ■), I) ? no
- holds(∃x. Oval(x), I) ? yes
  - holds(Oval(x), $I_{x/●}$) ? yes

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ?

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ?
  - holds(∃y. Above(x,y) v Above(y,x), $I_{x/▲}$) ?

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬭>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬭>}
- I(hat) = {<▲,■>,<⬭,●>
<■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ?
  - holds(∃y. Above(x,y) v Above(y,x), $I_{x/▲}$) ? yes

    holds(Above(x,y) v Above(y,x), $I_{x/▲,y/■}$) ? yes

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ? yes
  - holds(∃y. Above(x,y) v Above(y,x), $I_{x/▲}$) ? yes
    holds(Above(x,y) v Above(y,x), $I_{x/▲,y/■}$) ? yes
  - verify for all other values of x

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ? yes
  - holds(∃y. Above(x,y) v Above(y,x), Ix/▲) ? yes
    holds(Above(x,y) v Above(y,x), Ix/▲,y/■) ? yes
  - verify for all other values of x
- holds(∀ x. ∀ y. Above(x,y) v Above(y,x), I) ?

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ? yes
  - holds(∃y. Above(x,y) v Above(y,x), $I_{x/\blacktriangle}$) ? yes
    holds(Above(x,y) v Above(y,x), $I_{x/\blacktriangle, y/\blacksquare}$) ? yes
  - verify for all other values of x

- holds(∀ x. ∀ y. Above(x,y) v Above(y,x), I) ?
  - holds(Above(x,y) v Above(y,x), $I_{x/\blacksquare, y/\bullet}$) ?

FOL Example: Continued

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●> <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(∀x. ∃y. Above(x,y) v Above(y, x), I) ? yes
  - holds(∃y. Above(x,y) v Above(y,x), $I_{x/▲}$) ? yes
    holds(Above(x,y) v Above(y,x), $I_{x/▲,y/■}$) ? yes
  - verify for all other values of x

- holds(∀ x. ∀ y. Above(x,y) v Above(y,x), I) ? no
  - holds(Above(x,y) v Above(y,x), $I_{x/■,y/●}$) ? no

Writing FOL

Writing FOL

- Cats are mammals [Cat^1, Mammal^1]

Writing FOL

- Cats are mammals   [Cat^1, Mammal^1]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)

Writing FOL

- Cats are mammals [Cat^1, Mammal^1]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [Tall^1, Surveyor^1, Jane]

Writing FOL

- Cats are mammals [Cat^1, $Mammal^1$]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [$Tall^1$, $Surveyor^1$, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)

Writing FOL

- Cats are mammals [Cat^1, $Mammal^1$]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [$Tall^1$, $Surveyor^1$, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [$Nephew^2$, $Sibling^2$, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$

Writing FOL

- Cats are mammals [Cat^1, Mammal^1]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [Tall^1, Surveyor^1, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [Nephew^2, Sibling^2, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$ $\exists$z . [Sibling(y,z) $\wedge$ Son(x,z)]]

Writing FOL

- Cats are mammals [Cat^1, $Mammal^1$]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [$Tall^1$, $Surveyor^1$, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [$Nephew^2$, $Sibling^2$, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$ $\exists$z . [Sibling(y,z) $\wedge$ Son(x,z)]]
- A maternal grandmother is a mother's mother [functions: mgm, mother-of]

Writing FOL

- Cats are mammals [Cat^1, Mammal^1]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [Tall^1, Surveyor^1, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [Nephew^2, Sibling^2, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$ $\exists$z . [Sibling(y,z) $\wedge$ Son(x,z)]]
- A maternal grandmother is a mother's mother [functions: mgm, mother-of]
  - $\forall$xy. x=mgm(y) $\leftrightarrow$
    $\exists$z. x=mother-of(z) $\wedge$ z=mother-of(y)

Writing FOL

- Cats are mammals [Cat^1, $Mammal^1$]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [$Tall^1$, $Surveyor^1$, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [$Nephew^2$, $Sibling^2$, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$ $\exists$z . [Sibling(y,z) $\wedge$ Son(x,z)]]
- A maternal grandmother is a mother's mother [functions: mgm, mother-of]
  - $\forall$xy. x=mgm(y) $\leftrightarrow$
    $\exists$z. x=mother-of(z) $\wedge$ z=mother-of(y)
- Everybody loves somebody [$loves^2$]

Writing FOL

- Cats are mammals [Cat^1, $Mammal^1$]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [$Tall^1$, $Surveyor^1$, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [$Nephew^2$, $Sibling^2$, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$ $\exists$z . [Sibling(y,z) $\wedge$ Son(x,z)]]
- A maternal grandmother is a mother's mother [functions: mgm, mother-of]
  - $\forall$xy. x=mgm(y) $\leftrightarrow$
    $\exists$z. x=mother-of(z) $\wedge$ z=mother-of(y)
- Everybody loves somebody [$Loves^2$]
  - $\forall$x. $\exists$y. Loves(x,y)

Writing FOL

- Cats are mammals [Cat^1, $Mammal^1$]
  - $\forall$ x. Cat(x) $\rightarrow$ Mammal(x)
- Jane is a tall surveyor [$Tall^1$, $Surveyor^1$, Jane]
  - Tall(Jane) $\wedge$ Surveyor(Jane)
- A nephew is a sibling's son [$Nephew^2$, $Sibling^2$, Son^2]
  - $\forall$xy. [Nephew(x,y) $\leftrightarrow$ $\exists$z . [Sibling(y,z) $\wedge$ Son(x,z)]]
- A maternal grandmother is a mother's mother [functions: mgm, mother-of]
  - $\forall$xy. x=mgm(y) $\leftrightarrow$
    $\exists$z. x=mother-of(z) $\wedge$ z=mother-of(y)
- Everybody loves somebody [$loves^2$]
  - $\forall$x. $\exists$y. Loves(x,y)
  - $\exists$y. $\forall$x. Loves(x,y)

Writing More FOL

- Nobody loves Jane

Writing More FOL

- Nobody loves Jane
  - $\forall x.\ \neg$ Loves(x,Jane)

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg \text{Loves}(x, \text{Jane})$
  - $\neg \exists x. \text{Loves}(x, \text{Jane})$

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg$ Loves(x,Jane)
  - $\neg \exists x.$ Loves(x,Jane)
- Everybody has a father

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg Loves(x,Jane)$
  - $\neg \exists x. Loves(x,Jane)$
- Everybody has a father
  - $\forall x. \exists y. Father(y,x)$

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg$ Loves(x,Jane)
  - $\neg\exists x.$ Loves(x,Jane)
- Everybody has a father
  - $\forall$ x. $\exists$ y. Father(y,x)
- Everybody has a father and a mother

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg Loves(x, Jane)$
  - $\neg \exists x. Loves(x, Jane)$
- Everybody has a father
  - $\forall x. \exists y. Father(y, x)$
- Everybody has a father and a mother
  - $\forall x. \exists y, z. Father(y, x) \wedge Mother(z, x)$

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg Loves(x,Jane)$
  - $\neg \exists x. Loves(x,Jane)$
- Everybody has a father
  - $\forall x. \exists y. Father(y,x)$
- Everybody has a father and a mother
  - $\forall x. \exists y, z. Father(y,x) \wedge Mother(z,x)$

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg Loves(x,Jane)$
  - $\neg\exists x. Loves(x,Jane)$
- Everybody has a father
  - $\forall x. \exists y. Father(y,x)$
- Everybody has a father and a mother
  - $\forall x. \exists y, z. Father(y,x) \wedge Mother(z,x)$
- Whoever has a father, has a mother

Writing More FOL

- Nobody loves Jane
  - $\forall x.\ \neg$ Loves(x,Jane)
  - $\neg\exists x.$ Loves(x,Jane)
- Everybody has a father
  - $\forall$ x. $\exists$ y. Father(y,x)
- Everybody has a father and a mother
  - $\forall$ x. $\exists$ y, z. Father(y,x) $\wedge$ Mother(z,x)
- Whoever has a father, has a mother
  - $\forall$ x.

Writing More FOL

- Nobody loves Jane
  - $\forall x.\ \neg$ Loves(x,Jane)
  - $\neg\exists x.$ Loves(x,Jane)
- Everybody has a father
  - $\forall$ x. $\exists$ y. Father(y,x)
- Everybody has a father and a mother
  - $\forall$ x. $\exists$ yz. Father(y,x) $\wedge$ Mother(z,x)
- Whoever has a father, has a mother
  - $\forall$ x. [$\exists$ y. Father(y,x)]

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg$ Loves(x,Jane)
  - $\neg \exists x.$ Loves(x,Jane)
- Everybody has a father
  - $\forall$ x. $\exists$ y. Father(y,x)
- Everybody has a father and a mother
  - $\forall$ x. $\exists$ yz. Father(y,x) $\wedge$ Mother(z,x)
- Whoever has a father, has a mother
  - $\forall$ x.[[$\exists$ y. Father(y,x)]   [$\exists$ y. Mother(y,x)]]

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg Loves(x,Jane)$
  - $\neg \exists x. Loves(x,Jane)$
- Everybody has a father
  - $\forall x. \exists y. Father(y,x)$
- Everybody has a father and a mother
  - $\forall x. \exists yz. Father(y,x) \wedge Mother(z,x)$
- Whoever has a father, has a mother
  - $\forall x.[[\exists y. Father(y,x)] \rightarrow [\exists y. Mother(y,x)]]$

Writing More FOL

- Nobody loves Jane
  - $\forall x. \neg$ Loves(x,Jane)
  - $\neg\exists x.$ Loves(x,Jane)
- Everybody has a father
  - $\forall$ x. $\exists$ y. Father(y,x)
- Everybody has a father and a mother
  - $\forall$ x. $\exists$ yz. Father(y,x) $\wedge$ Mother(z,x)
- Whoever has a father, has a mother
  - $\forall$ x.[[$\exists$ y. Father(y,x)] $\rightarrow$ [$\exists$ y. Mother(y,x)]]

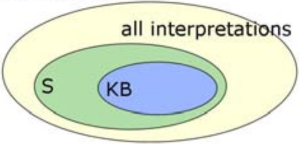
Entailment in First-Order Logic

Entailment in First-Order Logic

- KB entails S: for every interpretation I, if KB holds in I, then S holds in I

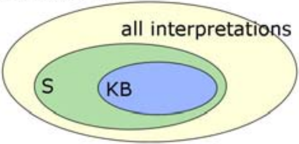
Entailment in First-Order Logic

- KB entails S: for every interpretation I, if KB holds in I, then S holds in I



Entailment in First-Order Logic

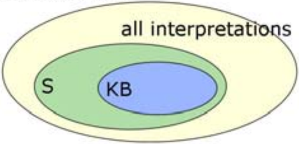
- KB entails S: for every interpretation I, if KB holds in I, then S holds in I



- Computing entailment is impossible in general, because there are infinitely many possible interpretations

Entailment in First-Order Logic

- KB entails S: for every interpretation I, if KB holds in I, then S holds in I



- Computing entailment is impossible in general, because there are infinitely many possible interpretations
- Even computing holds is impossible for interpretations with infinite universes

Intended Interpretations

$KB : (\forall x.\, \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\, \text{Square}(x) \rightarrow \neg \text{Oval}(x))$

Intended Interpretations

$KB : (\forall x.\, \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\, \text{Square}(x) \rightarrow \neg \text{Oval}(x))$

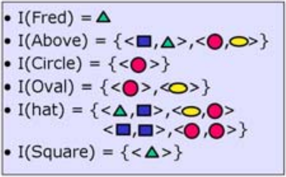
$S : \forall x.\, \text{Square}(x) \rightarrow \neg \text{Oval}(x)$

Intended Interpretations

$KB : (\forall x.\,\text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\,\text{Square}(x) \rightarrow \neg\text{Oval}(x))$

$S : \forall x.\,\text{Square}(x) \rightarrow \neg\text{Oval}(x)$

- We know holds(KB, I)
- We wonder whether holds(S, I)

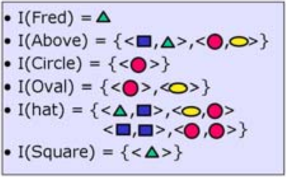


Intended Interpretations

$$KB: (\forall x.\, \mathrm{Circle}(x) \rightarrow \mathrm{Oval}(x)) \wedge (\forall x.\, \mathrm{Square}(x) \rightarrow \neg \mathrm{Oval}(x))$$

$$S: \forall x.\, \mathrm{Square}(x) \rightarrow \neg \mathrm{Oval}(x)$$

- We know holds(KB, I)
- We wonder whether holds(S, I)
- We could ask: Does KB entail S?



Intended Interpretations

$KB : (\forall x.\,\text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\,\text{Square}(x) \rightarrow \neg\text{Oval}(x))$

$S : \forall x.\,\text{Square}(x) \rightarrow \neg\text{Oval}(x)$

- We know holds(KB, I)
- We wonder whether holds(S, I)
- We could ask: Does KB entail S?
- Or we could just try to check whether holds(S, I)

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●> <■,■>,<●,●>}
- I(Square) = {<▲>}

An Infinite Interpretation

$KB: (\forall x. \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x))$

$S: \forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x)$

An Infinite Interpretation

$KB: (\forall x. \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x))$

$S: \forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x)$

$U_1 = \{1, 2, 3, \ldots\}$

An Infinite Interpretation

$KB : (\forall x. \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x))$

$S : \forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x)$

$U_1 = \{1, 2, 3, \ldots\}$
$I_1(\text{circle}) = \{4, 8, 12, 16, \ldots\}$

An Infinite Interpretation

$KB : (\forall x.\, \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\, \text{Square}(x) \rightarrow \neg\text{Oval}(x))$

$S : \forall x.\, \text{Square}(x) \rightarrow \neg\text{Oval}(x)$

$U_1 = \{1, 2, 3, \ldots\}$
$I_1(\text{circle}) = \{4, 8, 12, 16, \ldots\}$
$I_1(\text{oval}) = \{2, 4, 6, 8, \ldots\}$

An Infinite Interpretation

$KB: (\forall x.\,\text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\,\text{Square}(x) \rightarrow \neg\text{Oval}(x))$

$S: \forall x.\,\text{Square}(x) \rightarrow \neg\text{Oval}(x)$

$U_1 = \{1, 2, 3, \ldots\}$
$I_1(\text{circle}) = \{4, 8, 12, 16, \ldots\}$
$I_1(\text{oval}) = \{2, 4, 6, 8, \ldots\}$
$I_1(\text{square}) = \{1, 3, 5, 7, \ldots\}$

An Infinite Interpretation

$KB: (\forall x.\, \mathrm{Circle}(x) \rightarrow \mathrm{Oval}(x)) \wedge (\forall x.\, \mathrm{Square}(x) \rightarrow \neg \mathrm{Oval}(x))$

$S: \forall x.\, \mathrm{Square}(x) \rightarrow \neg \mathrm{Oval}(x)$

- Does KB hold in I_1?

$U_1 = \{1, 2, 3, \ldots\}$
$I_1(\text{circle}) = \{4, 8, 12, 16, \ldots\}$
$I_1(\text{oval}) = \{2, 4, 6, 8, \ldots\}$
$I_1(\text{square}) = \{1, 3, 5, 7, \ldots\}$

An Infinite Interpretation

$KB: (\forall x. \text{Circle}(x) \to \text{Oval}(x)) \wedge (\forall x. \text{Square}(x) \to \neg\text{Oval}(x))$

$S: \forall x. \text{Square}(x) \to \neg\text{Oval}(x)$

- Does KB hold in I_1?
- Yes, but can't answer via enumerating U

$U_1 = \{1, 2, 3, \ldots\}$
$I_1(\text{circle}) = \{4, 8, 12, 16, \ldots\}$
$I_1(\text{oval}) = \{2, 4, 6, 8, \ldots\}$
$I_1(\text{square}) = \{1, 3, 5, 7, \ldots\}$

An Infinite Interpretation

$KB : (\forall x.\,\text{Circle}(x) \to \text{Oval}(x)) \wedge (\forall x.\,\text{Square}(x) \to \neg\text{Oval}(x))$

$S : \forall x.\,\text{Square}(x) \to \neg\text{Oval}(x)$

- Does KB hold in I_1?
- Yes, but can't answer via enumerating U
- S also holds in I_1
- No way to verify mechanically

U_1 = {1, 2, 3, …}
I_1(circle)={4, 8, 12, 16, …}
I_1(oval) = {2, 4, 6, 8, …}
I_1(square)={1, 3, 5, 7, …}

An Argument for Entailment

$KB: (\forall x. \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x. \text{Square}(x) \rightarrow \neg \text{Oval}(x))$

$S_1: \forall x, y.\ \text{Circle}(x) \wedge \text{Oval}(y) \wedge \neg \text{Circle}(y) \rightarrow \text{Above}(x, y)$

An Argument for Entailment

$KB : (\forall x.\, \mathrm{Circle}(x) \rightarrow \mathrm{Oval}(x)) \wedge (\forall x.\, \mathrm{Square}(x) \rightarrow \neg \mathrm{Oval}(x))$

$S_1 : \forall x, y.\, \mathrm{Circle}(x) \wedge \mathrm{Oval}(y) \wedge \neg \mathrm{Circle}(y) \rightarrow \mathrm{Above}(x, y)$

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

- holds(KB, I)
- holds(S_1, I)

An Argument for Entailment

$$KB: (\forall x.\, \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\, \text{Square}(x) \rightarrow \neg \text{Oval}(x))$$

$$S_1: \forall x, y.\, \text{Circle}(x) \wedge \text{Oval}(y) \wedge \neg \text{Circle}(y) \rightarrow \text{Above}(x, y)$$

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●> <■,■>,<●,●>}
- I(Square) = {<▲>}

U_1 = {1, 2, 3, …}
I_1(Circle)={4, 8, 12, 16, …}
I_1(Oval) = {2, 4, 6, 8, …}
I_1(Square)={1, 3, 5, 7, …}
I_1(Above) = >

- holds(KB, I)
- holds(S_1, I)

An Argument for Entailment

$$KB : (\forall x.\, \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\, \text{Square}(x) \rightarrow \neg \text{Oval}(x))$$

$$S_1 : \forall x, y.\ \text{Circle}(x) \wedge \text{Oval}(y) \wedge \neg \text{Circle}(y) \rightarrow \text{Above}(x, y)$$

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●>
  <■,■>,<●,●>}
- I(Square) = {<▲>}

U_1 = {1, 2, 3, …}
I_1(Circle)={4, 8, 12, 16, …}
I_1(Oval) = {2, 4, 6, 8, …}
I_1(Square)={1, 3, 5, 7, …}
I_1(Above) = >

- holds(KB, I)
- holds(S_1, I)

- holds(KB, I_1)
- fails(S_1, I_1)

An Argument for Entailment

$KB: (\forall x.\, \text{Circle}(x) \rightarrow \text{Oval}(x)) \wedge (\forall x.\, \text{Square}(x) \rightarrow \neg\text{Oval}(x))$

$S_1: \forall x, y.\, \text{Circle}(x) \wedge \text{Oval}(y) \wedge \neg\text{Circle}(y) \rightarrow \text{Above}(x, y)$

- I(Fred) = ▲
- I(Above) = {<■,▲>,<●,⬬>}
- I(Circle) = {<●>}
- I(Oval) = {<●>,<⬬>}
- I(hat) = {<▲,■>,<⬬,●> <■,■>,<●,●>}
- I(Square) = {<▲>}

U_1 = {1, 2, 3, …}
I_1(Circle)={4, 8, 12, 16, …}
I_1(Oval) = {2, 4, 6, 8, …}
I_1(Square)={1, 3, 5, 7, …}
I_1(Above) = >

- holds(KB, I)
- holds(S_1, I)

- holds(KB, I_1)
- fails(S_1, I_1)

KB doesn't entail S_1!

Proof and Entailment

- Entailment captures general notion of "follows from"

Proof and Entailment

- Entailment captures general notion of "follows from"
- Can't evaluate it directly by enumerating interpretations

Proof and Entailment

- Entailment captures general notion of "follows from"
- Can't evaluate it directly by enumerating interpretations
- So, we'll do proofs

Proof and Entailment

- Entailment captures general notion of "follows from"
- Can't evaluate it directly by enumerating interpretations
- So, we'll do proofs
- In FOL, if S is entailed by KB, then there is a finite proof of S from KB

First-Order Resolution

First-Order Resolution

$$\frac{\begin{array}{c}\forall x.\ P(x) \rightarrow Q(x) \\ P(A)\end{array}}{Q(A)}$$

uppercase letters: constants

lowercase letters: variables

First-Order Resolution

$$\frac{\forall x.\ P(x) \rightarrow Q(x) \qquad P(A)}{Q(A)}$$

Syllogism:

All men are mortal

Socrates is a man

Socrates is mortal

uppercase letters: constants

lowercase letters: variables

First-Order Resolution

$$\frac{\forall x.\ P(x) \rightarrow Q(x) \qquad P(A)}{Q(A)}$$

Syllogism:

All men are mortal
Socrates is a man
———
Socrates is mortal

uppercase letters: constants

lowercase letters: variables

$$\frac{\forall x.\ \neg P(x) \vee Q(x) \qquad P(A)}{Q(A)}$$

Equivalent by definition of implication

First-Order Resolution

$$\frac{\begin{array}{c}\forall x.\ P(x) \rightarrow Q(x) \\ P(A)\end{array}}{Q(A)}$$

Syllogism:

$$\frac{\begin{array}{l}\text{All men are mortal} \\ \text{Socrates is a man}\end{array}}{\text{Socrates is mortal}}$$

uppercase letters: constants

lowercase letters: variables

$$\frac{\begin{array}{c}\forall x.\ \neg P(x) \vee Q(x) \\ P(A)\end{array}}{Q(A)}$$

Equivalent by definition of implication

$$\frac{\begin{array}{c}\neg P(A) \vee Q(A) \\ P(A)\end{array}}{Q(A)}$$

Substitute A for x, still true

then

Propositional resolution

First-Order Resolution

$$\frac{\forall x.\ P(x) \rightarrow Q(x) \qquad P(A)}{Q(A)}$$

Syllogism:
All men are mortal
Socrates is a man
Socrates is mortal

uppercase letters: constants

lowercase letters: variables

$$\frac{\forall x.\ \neg P(x) \vee Q(x) \qquad P(A)}{Q(A)}$$

Equivalent by definition of implication

Two new things:

- converting FOL to clausal form
- resolution with variable substitution

$$\frac{\neg P(A) \vee Q(A) \qquad P(A)}{Q(A)}$$

Substitute A for x, still true

then

Propositional resolution

Clausal Form

- like CNF in outer structure
- no quantifiers

$$\forall x.\ \exists y.\ P(x) \rightarrow R(x, y)$$

$$\Downarrow$$

$$\neg P(x) \vee R(x, F(x))$$

Converting to Clausal Form

Converting to Clausal Form

1. Eliminate arrows

$$\alpha \leftrightarrow \beta \Rightarrow (\alpha \to \beta) \wedge (\beta \to \alpha)$$

$$\alpha \to \beta \Rightarrow \neg \alpha \vee \beta$$

Converting to Clausal Form

1. Eliminate arrows

$$\alpha \leftrightarrow \beta \Rightarrow (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$$

$$\alpha \rightarrow \beta \Rightarrow \neg\alpha \vee \beta$$

2. Drive in negation

$$\neg(\alpha \vee \beta) \Rightarrow \neg\alpha \wedge \neg\beta$$

$$\neg(\alpha \wedge \beta) \Rightarrow \neg\alpha \vee \neg\beta$$

$$\neg\neg\alpha \Rightarrow \alpha$$

$$\neg\forall x.\ \alpha \Rightarrow \exists x.\ \neg\alpha$$

$$\neg\exists x.\ \alpha \Rightarrow \forall x.\ \neg\alpha$$

Converting to Clausal Form

1. Eliminate arrows

$$\alpha \leftrightarrow \beta \Rightarrow (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$$

$$\alpha \rightarrow \beta \Rightarrow \neg\alpha \vee \beta$$

2. Drive in negation

$$\neg(\alpha \vee \beta) \Rightarrow \neg\alpha \wedge \neg\beta$$

$$\neg(\alpha \wedge \beta) \Rightarrow \neg\alpha \vee \neg\beta$$

$$\neg\neg\alpha \Rightarrow \alpha$$

$$\neg\forall x.\ \alpha \Rightarrow \exists x.\ \neg\alpha$$

$$\neg\exists x.\ \alpha \Rightarrow \forall x.\ \neg\alpha$$

3. Rename variables apart

$$\forall x.\ \exists y.\ (\neg P(x) \vee \exists x.\ Q(x, y)) \Rightarrow$$

$$\forall x_1.\ \exists y_2.\ (\neg P(x_1) \vee \exists x_3.\ Q(x_3, y_2))$$

Skolemization

4. Skolemize

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\, P(x) \Rightarrow P(\text{Fred})$$

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\, P(x) \Rightarrow P(\text{Fred})$$

$$\exists x, y.\, R(x, y) \Rightarrow R(\text{Thing1}, \text{Thing2})$$

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\, \mathrm{P}(x) \Rightarrow \mathrm{P}(\mathrm{Fred})$$

$$\exists x, y.\, \mathrm{R}(x, y) \Rightarrow \mathrm{R}(\mathrm{Thing1}, \mathrm{Thing2})$$

$$\exists x.\, \mathrm{P}(x) \wedge \mathrm{Q}(x) \Rightarrow \mathrm{P}(\mathrm{Fleep}) \wedge \mathrm{Q}(\mathrm{Fleep})$$

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\ P(x) \Rightarrow P(\text{Fred})$$

$$\exists x, y.\ R(x, y) \Rightarrow R(\text{Thing1}, \text{Thing2})$$

$$\exists x.\ P(x) \wedge Q(x) \Rightarrow P(\text{Fleep}) \wedge Q(\text{Fleep})$$

$$\exists x.\ P(x) \wedge \exists x.\ Q(x) \Rightarrow P(\text{Frog}) \wedge Q(\text{Grog})$$

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\ P(x) \Rightarrow P(\text{Fred})$$

$$\exists x, y.\ R(x, y) \Rightarrow R(\text{Thing1}, \text{Thing2})$$

$$\exists x.\ P(x) \wedge Q(x) \Rightarrow P(\text{Fleep}) \wedge Q(\text{Fleep})$$

$$\exists x.\ P(x) \wedge \exists x.\ Q(x) \Rightarrow P(\text{Frog}) \wedge Q(\text{Grog})$$

$$\exists y.\ \forall x.\ \text{Loves}(x, y) \Rightarrow \forall y.\ \text{Loves}(x, \text{Englebert})$$

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\ P(x) \Rightarrow P(\text{Fred})$$
$$\exists x, y.\ R(x, y) \Rightarrow R(\text{Thing1}, \text{Thing2})$$
$$\exists x.\ P(x) \wedge Q(x) \Rightarrow P(\text{Fleep}) \wedge Q(\text{Fleep})$$
$$\exists x.\ P(x) \wedge \exists x.\ Q(x) \Rightarrow P(\text{Frog}) \wedge Q(\text{Grog})$$
$$\exists y.\ \forall x.\ \text{Loves}(x, y) \Rightarrow \forall y.\ \text{Loves}(x, \text{Englebert})$$

   - substitute new function of all universal vars in outer scopes

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\ P(x) \Rightarrow P(\text{Fred})$$
$$\exists x, y.\ R(x, y) \Rightarrow R(\text{Thing1}, \text{Thing2})$$
$$\exists x.\ P(x) \wedge Q(x) \Rightarrow P(\text{Fleep}) \wedge Q(\text{Fleep})$$
$$\exists x.\ P(x) \wedge \exists x.\ Q(x) \Rightarrow P(\text{Frog}) \wedge Q(\text{Grog})$$
$$\exists y.\ \forall x.\ \text{Loves}(x, y) \Rightarrow \forall y.\ \text{Loves}(x, \text{Englebert})$$

   - substitute new function of all universal vars in outer scopes

$$\forall x.\ \exists y.\ \text{Loves}(x, y) \Rightarrow \forall x.\ \text{Loves}(x, \text{Beloved}(x))$$

Skolemization

4. Skolemize
   - substitute new name for each existential var

$$\exists x.\ P(x) \Rightarrow P(\text{Fred})$$
$$\exists x,y.\ R(x,y) \Rightarrow R(\text{Thing1}, \text{Thing2})$$
$$\exists x.\ P(x) \wedge Q(x) \Rightarrow P(\text{Fleep}) \wedge Q(\text{Fleep})$$
$$\exists x.\ P(x) \wedge \exists x.\ Q(x) \Rightarrow P(\text{Frog}) \wedge Q(\text{Grog})$$
$$\exists y.\ \forall x.\ \text{Loves}(x,y) \Rightarrow \forall y.\ \text{Loves}(x, \text{Englebert})$$

   - substitute new function of all universal vars in outer scopes

$$\forall x.\ \exists y.\ \text{Loves}(x,y) \Rightarrow \forall x.\ \text{Loves}(x, \text{Beloved}(x))$$
$$\forall x.\ \exists y.\ \forall z.\ \exists w.\ P(x,y,z) \wedge R(y,z,w) \Rightarrow$$
$$P(x,F(x),z) \wedge R(F(x),z,G(x,z))$$

Convert to Clausal Form: Last Steps

5. Drop universal quantifiers

   $\forall x.\ \text{Loves}(x, \text{Beloved}(x)) \Rightarrow \text{Loves}(x, \text{Beloved}(x))$

Convert to Clausal Form: Last Steps

5. Drop universal quantifiers

   $\forall x.\ \mathrm{Loves}(x, \mathrm{Beloved}(x)) \Rightarrow \mathrm{Loves}(x, \mathrm{Beloved}(x))$

6. Distribute and over or; return clauses

   $$\mathrm{P}(z) \vee (\mathrm{Q}(z,w) \wedge \mathrm{R}(w,z)) \Rightarrow \{\{\mathrm{P}(z), \mathrm{Q}(z,w)\}, \{\mathrm{P}(z), \mathrm{R}(w,z)\}\}$$

Convert to Clausal Form: Last Steps

5. Drop universal quantifiers

$$\forall x.\ \text{Loves}(x, \text{Beloved}(x)) \Rightarrow \text{Loves}(x, \text{Beloved}(x))$$

6. Distribute and over or; return clauses

$$P(z) \vee (Q(z,w) \wedge R(w,z)) \Rightarrow \{\{P(z), Q(z,w)\}, \{P(z), R(w,z)\}\}$$

7. Rename the variables in each clause

$$\{\{P(z), Q(z,w)\}, \{P(z), R(w,z)\}\} \Rightarrow \{\{P(z_1), Q(z_1,w_1)\}, \{P(z_2), R(w_2,z_2)\}\}$$

Example: Converting to clausal form

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$

Example: Converting to clausal form

a. John owns a dog
$\exists$ x. D(x) $\wedge$ O(J,x)
D(Fido) $\wedge$ O(J, Fido)

Example: Converting to clausal form

a. John owns a dog
$\exists$ x. D(x) $\wedge$ O(J,x)
D(Fido) $\wedge$ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
$\forall$ x. ($\exists$ y. D(y) $\wedge$ O(x,y)) $\rightarrow$ L(x)

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$
$D(Fido) \wedge O(J, Fido)$

b. Anyone who owns a dog is a lover-of-animals
$\forall x.\ (\exists y.\ D(y) \wedge O(x,y)) \rightarrow L(x)$
$\forall x.\ (\neg\exists y.\ (D(y) \wedge O(x,y)) \vee L(x)$

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$
$D(Fido) \wedge O(J, Fido)$

b. Anyone who owns a dog is a lover-of-animals
$\forall x.\ (\exists y.\ D(y) \wedge O(x,y)) \rightarrow L(x)$
$\forall x.\ (\neg\exists y.\ (D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg(D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg D(y) \vee \neg O(x,y) \vee L(x)$

Example: Converting to clausal form

a. John owns a dog
$\exists$ x. D(x) $\wedge$ O(J,x)
D(Fido) $\wedge$ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
$\forall$ x. ($\exists$ y. D(y) $\wedge$ O(x,y)) $\rightarrow$ L(x)
$\forall$ x. ($\neg\exists$ y. (D(y) $\wedge$ O(x,y)) v L(x)
$\forall$ x. $\forall$ y. $\neg$(D(y) $\wedge$ O(x,y)) v L(x)
$\forall$ x. $\forall$ y. $\neg$ D(y) v $\neg$ O(x,y) v L(x)
$\neg$ D(y) v $\neg$ O(x,y) v L(x)

Example: Converting to clausal form

a. John owns a dog
$\exists$ x. D(x) $\wedge$ O(J,x)
D(Fido) $\wedge$ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
$\forall$ x. ($\exists$ y. D(y) $\wedge$ O(x,y)) $\rightarrow$ L(x)
$\forall$ x. ($\neg\exists$ y. (D(y) $\wedge$ O(x,y)) v L(x)
$\forall$ x. $\forall$ y. $\neg$(D(y) $\wedge$ O(x,y)) v L(x)
$\forall$ x. $\forall$ y. $\neg$ D(y) v $\neg$ O(x,y) v L(x)
$\neg$ D(y) v $\neg$ O(x,y) v L(x)

c. Lovers-of-animals do not kill animals
$\forall$ x. L(x) $\rightarrow$ ($\forall$ y. A(y) $\rightarrow$ $\neg$ K(x,y))

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$
$D(Fido) \wedge O(J, Fido)$

b. Anyone who owns a dog is a lover-of-animals
$\forall x.\ (\exists y.\ D(y) \wedge O(x,y)) \rightarrow L(x)$
$\forall x.\ (\neg\exists y.\ (D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg(D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg D(y) \vee \neg O(x,y) \vee L(x)$
$\neg D(y) \vee \neg O(x,y) \vee L(x)$

c. Lovers-of-animals do not kill animals
$\forall x.\ L(x) \rightarrow (\forall y.\ A(y) \rightarrow \neg K(x,y))$
$\forall x.\ \neg L(x) \vee (\forall y.\ A(y) \rightarrow \neg K(x,y))$
$\forall x.\ \neg L(x) \vee (\forall y.\ \neg A(y) \vee \neg K(x,y))$

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$
$D(Fido) \wedge O(J, Fido)$

b. Anyone who owns a dog is a lover-of-animals
$\forall x.\ (\exists y.\ D(y) \wedge O(x,y)) \rightarrow L(x)$
$\forall x.\ (\neg\exists y.\ (D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg(D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg D(y) \vee \neg O(x,y) \vee L(x)$
$\neg D(y) \vee \neg O(x,y) \vee L(x)$

c. Lovers-of-animals do not kill animals
$\forall x.\ L(x) \rightarrow (\forall y.\ A(y) \rightarrow \neg K(x,y))$
$\forall x.\ \neg L(x) \vee (\forall y.\ A(y) \rightarrow \neg K(x,y))$
$\forall x.\ \neg L(x) \vee (\forall y.\ \neg A(y) \vee \neg K(x,y))$
$\neg L(x) \vee \neg A(y) \vee \neg K(x,y)$

More converting to clausal form

d. Either Jack killed Tuna or curiosity killed Tuna
K(J,T) v K(C,T)

More converting to clausal form

d. Either Jack killed Tuna or curiosity killed Tuna
K(J,T) v K(C,T)

e. Tuna is a cat
C(T)

More converting to clausal form

d. Either Jack killed Tuna or curiosity killed Tuna
K(J,T) v K(C,T)

e. Tuna is a cat
C(T)

f. All cats are animals
¬ C(x) v A(x)

A Proof Using Resolution

We are given:

$$\forall x\ [P(x) \Rightarrow [\forall y\ [P(y) \Rightarrow P(f(x,y))] \wedge \\ \neg\forall y\ [Q(x,y) \Rightarrow P(y)]]]$$

where: P, Q are predicates
x, y are variables
f is a function

Step 1:

Eliminate Implication

use $x \Rightarrow y \equiv \neg x \vee y$

We obtain:

$$\forall x\ [\neg P(x) \vee [\forall y\ [\neg P(y) \vee P(f(x,y))] \wedge \\ \neg\forall y\ [\neg Q(x,y) \vee P(y)]]]$$

Step 2

Reduce Scope of Negation Symbols

Use:

- DeMorgan's Laws
  $\neg(X \wedge Y) \equiv \neg X \vee \neg Y$
  $\neg(X \vee Y) \equiv \neg X \wedge \neg Y$
- $\neg(\neg X) \equiv X$
- And from Quantifier Meaning Rules:
  $\neg\forall x \;\; P(x) \equiv \exists x[\neg P(x)]$
  $\neg\exists x \;\; P(x) \equiv \forall x[\neg P(x)]$

Step 2 Continued:

Reduce Scope of Negation Symbols

Actual Reduction:

$$\forall x\ [\neg P(x) \vee [\forall y\ [\neg P(y) \vee P(f(x,y))] \wedge$$
$$\underbrace{\neg \forall y\ [\neg Q(x,y) \vee P(y)]}_{\textit{only underlined part}}]]$$

$$\Downarrow \neg \forall x\ \ P(x) \equiv \exists x\ \ [\neg P(x)]$$

$$\exists y\ \ [\neg[\neg Q(x,y) \vee P(y)]]$$

$$\Downarrow \text{DeMorgan's Law}$$

$$\exists y\ \ [\neg[\neg Q(x,y)] \wedge \neg P(y)]$$

$$\Downarrow \text{Double Negation}$$

$$\exists y\ \ [Q(x,y) \wedge \neg P(y)]$$

Step 3

Standardize Variables

Rename variable associated with each quantifier so it is unique. Can do this without altering truth values (in quantifiers only).
For example:

$$\forall x \ [P(x) \Rightarrow \exists x \ Q(x)]$$

is rewritten

$$\forall x \ [P(x) \Rightarrow \exists y \ Q(y)]$$

Result in Example:

$$\forall x \ [\neg P(x) \vee [\forall y \ [\neg P(y) \vee P(f(x,y))] \wedge \underline{\exists y \ [Q(x,y) \wedge \neg P(y)]}]]$$

$$\Downarrow \forall x \ [\neg P(x) \vee [\forall y \ [\neg P(y) \vee P(f(x,y))] \wedge \underline{\exists w \ [Q(x,w) \wedge \neg P(w)]}]]$$

Step 4

Get Rid of Existential Quantifiers:
"Skolemize"

How to use Skolem Functions:

$$\forall y \ [\exists x \ \ P(x, y)]$$

For all y, there exists an x, <u>perhaps depending on y</u>, such that $P(x, y)$.
Skolem function $g(y)$ maps y onto the necessary x.

$$\forall y \ \ P[g(y), y)]$$

<u>Rule:</u>
Replace each occurrence of existential quantifier <u>variable</u> by a Skolem function whose arguments are universally quantified <u>variables</u> whose scopes <u>include</u> the existential quantifier being eliminated.

Step 4 Continued

Example of Skolemization

$$\underline{\forall x\ \ P(x)} \wedge \forall y \forall z\ \ [\exists w\ \ [P(y,z,w) \wedge R(y)]]$$

not in scope

$$\Downarrow$$

$$\forall x\ \ P(x) \wedge \forall y \forall z\ \ [P(y,z,\underline{g(y,z)}) \wedge R(y)]]$$

Result in Example Proof:

$$\forall x\ \ [\neg P(x) \vee [\forall y\ \ [\neg P(y) \vee P(f(x,y))] \wedge$$
$$\underline{\exists w}\ \ [Q(x,\underline{w}) \wedge \neg P(\underline{w})]]]$$

$\Downarrow$ only x in scope

$$\forall x\ \ [\neg P(x) \vee [\forall y\ \ [\neg P(y) \vee P(f(x,y))] \wedge$$
$$[Q(x,g(x)) \wedge \neg P(g(x))]]]$$

Step 5

Move all Universal Quantifiers to Left
<u>Prenex</u> Form

This can be done because all variables are unique to each universal quantifier.

Result in Example Proof:

$$\forall x\ [\neg P(x) \vee [\underline{\forall y}\ [\neg P(y) \vee P(f(x,y))] \wedge$$
$$[Q(x,g(x)) \wedge \neg P(g(x))]]]$$
$$\Downarrow$$
$$\forall x \underline{\forall y}\ [\neg P(x) \vee [\neg P(y) \vee P(f(x,y))] \wedge$$
$$[Q(x,g(x)) \wedge \neg P(g(x))]]$$

Step 6

Rewrite in Conjunctive Normal Form

Resulting in a conjunction of a set of disjunctions of literals.
(i.e., move disjunctions down to the literals.)

Use the Distributive laws repeatedly.

$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$

Result in Example Proof:

$$\begin{array}{l} \forall x \forall y\ [\neg P(x) \vee [\neg P(y) \vee P(f(x,y))] \wedge \\ \quad [Q(x,g(x)) \wedge \neg P(g(x))]] \\ \qquad\qquad \Downarrow \text{apply distributive law twice} \\ \forall x \forall y\ [[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge \\ \quad [\neg P(x) \vee Q(x,g(x))] \wedge \\ \quad [\neg P(x) \vee \neg P(g(x))]] \end{array}$$

Step 7

Eliminate Universal Quantifiers

All variables are assumed to be universally quantified.

Result in Example Proof:

$$[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge$$
$$[\neg P(x) \vee Q(x, g(x))] \wedge$$
$$[\neg P(x) \vee \neg P(g(x))]$$

Step 8

Eliminate $\wedge$ Symbols

Separate into clauses

Result in Example Proof:

$$\neg P(x) \vee \neg P(y) \vee P(f(x,y))$$
$$\neg P(x) \vee Q(x, g(x))$$
$$\neg P(x) \vee \neg P(g(x))$$

Step 9

Rename Variables

No variable symbol should appear in more than one clause.

(Standardizing the variables apart).

Result in Example Proof:

$$\neg P(x1) \vee \neg P(y) \vee P(f(x1, y))$$
$$\neg P(x2) \vee Q(x2, g(x2))$$
$$\neg P(x3) \vee \neg P(g(x3))$$

Done! (Whew!!)

Now we can **start** actual proving by resolution from these ground clauses.

Resolution Theorem Proving

- Proof by contradiction

- Unification is the major process:
  matching while propagating variable bindings

Procedure

1. Negate theorem trying to prove
2. Add to axiom set
3. Convert all axioms to clause form
4. Resolve clauses until:
   a) Empty clause produced -> T
   b) No resolvable clauses -> F

Empty clause denotes contradiction

Completeness and Decidability

- Complete: If KB entails S, then we can prove S from KB

Completeness and Decidability

- Complete: If KB entails S, then we can prove S from KB
- Gödel's Completeness Theorem: *There exists a complete proof system for FOL*

Completeness and Decidability

- Complete: If KB entails S, then we can prove S from KB
- Gödel's Completeness Theorem: *There exists a complete proof system for FOL*
- Robinson's Completeness Theorem: *Resolution refutation is a complete proof system for FOL*

Completeness and Decidability

- Complete: If KB entails S, then we can prove S from KB

- Gödel's Completeness Theorem: *There exists a complete proof system for FOL*
- Robinson's Completeness Theorem: *Resolution refutation is a complete proof system for FOL*

- *FOL is semi-decidable:* if the desired conclusion follows from the premises then eventually resolution refutation will find a contradiction.
  - If there's a proof, we'll halt with it
  - If not, maybe we'll halt, maybe not

Adding Arithmetic

Adding Arithmetic

- Gödel's Incompleteness Theorem: *There is no consistent, complete proof system for FOL + Arithmetic.*
- Either there are sentences that are true, but not provable or there are sentences that are provable, but not true.

Adding Arithmetic

- Gödel's Incompleteness Theorem: *There is no consistent, complete proof system for FOL + Arithmetic.*
- Either there are sentences that are true, but not provable or there are sentences that are provable, but not true.

- Arithmetic gives you the ability to construct code-names for sentences within the logic.

  P = "P is not provable."

Adding Arithmetic

- Gödel's Incompleteness Theorem: *There is no consistent, complete proof system for FOL + Arithmetic.*
- Either there are sentences that are true, but not provable or there are sentences that are provable, but not true.

- Arithmetic gives you the ability to construct code-names for sentences within the logic.

  P = "P is not provable."
  - If P is true: it's not provable (incomplete)
  - If P is false: it's provable (inconsistent)