COMP44[illegible]nowledge Representation and Reas[illegible]g

First-Order Logic 1

Maurice Pagnucco
School of Computer Science and Engineering

COMP4418, Week 2

First-Order Logic

- First-order logic furnishes a much more expressive knowledge representation language than propositional logic
- We can directly talk about objects and their properties, relations between them, etc. . . .
- Here we discuss first-order logic and resolution
- However, there is a price to pay for this expressiveness in terms of decidability
- References:
  - Ivan Bratko, *Prolog Programming for Artificial Intelligence*, Addison-Wesley, 2001. (Chapter 15)
  - Stuart J. Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach*, Prentice-Hall International, 1995. (Chapter 6)

Overview

- Syntax of First-Order Logic
- Semantics of First-Order Logic
- Conjunctive Normal Form
- Unification
- First-Order Resolution
- Soundness and Completeness
- Decidability
- Conclusion

Syntax of First-Order Logic

- **Constant Symbols:** *a*, *b*, *Mary* (objects)
- **Variables:** *x*, *y*, . . .
- **Function Symbols:** *f*, *mother_of*, *sine*, . . .
- **Predicate Symbols:** *Mother*, *likes*, . . .
- **Quantifiers:** ∀ (universal), ∃ (existential)

Terms: constant, variable, functions applied to terms (refer to objects)

- **Atomic Sentences:** predicate applied to terms (state facts)
- **Ground (closed) term:** a term with no variable symbols

Syntax of First-Order Logic

Sentence ::= AtomicSentence || Sentence Connective Sentence
|| Quantifier Variable Sentence || ¬ Sentence || **(** Sentence **)**
AtomicSentence ::= Predicate **(** Term **,** ... **)**
Term ::= Function **(** Term* **)** || Constant || Variable
Connective ::= → || ∧ || ∨ || ↔
Quantifier ::= ∀ || ∃
Constant ::= **a** || **John** || ...
Variable ::= *x* || *men* || ...
Predicate ::= *P* || **Red** || **Between** || ...
Function ::= *f* || **Father** || ...

Converting English into First-Order Logic

- Everyone likes lying on the beach — $\forall x\ Beach(x)$
- Someone likes Fido — $\exists x\ Likes(x,\ Fido)$
- No one likes Fido — $\neg\exists x\ Likes(x,\ Fido)$
- Fido doesn't like everyone — $\neg\forall x\ Likes(Fido,\ x)$
- All cats are mammals — $\forall x\ (Cat(x) \rightarrow Mammal(x))$
- Some mammals are carnivorous — $\exists x\ (Mammal(x) \wedge Carnivorous(x))$

Nested Quantifiers

Note that the order of quantifiers is very important

- Everything likes everything — $\forall x\ \forall y\ Likes(x,\ y)$
- Something likes something — $\exists x\ \exists y\ Likes(x,\ y)$
- Everything likes something — $\forall x\ \exists y\ Likes(x,\ y)$
- There is something liked by everything — $\exists y\ \forall x\ Likes(x,\ y)$

Scope of Quantifiers

- The *scope* of a quantifier in formula ϕ is that subformula ψ of ϕ of which that quantifier is the main logical operator
- Variables belong to the *innermost* quantifier that mentions them
- Examples:
  - $Q(x) \to \forall y\ P(x,\ y)$ — scope of $\forall y$ is $P(x,\ y)$
  - $\forall z\ P(z) \to \neg Q(z)$ — scope of $\forall z$ is $P(z)$ but not $Q(z)$
  - $\exists x(P(x) \to \forall x\ P(x))$
  - $\forall x(P(x) \to Q(x)) \to (\forall x\ P(x) \to \forall x\ Q(x))$

Terminology

- *Free-variable occurrence* in a formula —
  - All variables in an atomic formula
  - The free-variable occurrences in $\neg\phi$ are those in ϕ
  - The free-variable occurrences in $\phi \oplus \psi$ are those in ϕ and ψ for any connective $\oplus$
  - The free-variable occurrences in $\forall x\ \Phi$ and $\exists x\ \Phi$ are those in Φ except for occurrences of x
- *Open formula* — A formula in which free variables occur
- *Closed formula* — A formula with no free variables
- Closed formulae are also known as *sentences*

Semantics of First-Order Logic

- A world in which a sentence is true under a particular interpretation is known as a *model* of that sentence under the interpretation
- Constant symbols an interpretation specifies which object in the world a constant refers to

  Predicate symbols an interpretation specifies which relation in the model a predicate refers to

  Function symbols an interpretation specifies which function in the model a function symbol refers to

  Universal quantifier is true iff all instances are true

  Existential quantifier is true iff one instance is true

Conversion into Conjunctive Normal Form

1. Eliminate implication

$$\phi \rightarrow \psi \equiv \neg\phi \vee \psi$$

2. Move negation inwards (ne[illegible] normal form)

$$\neg(\phi \wedge \psi) \equiv \neg\phi \vee \neg\psi$$

$$\neg(\phi \vee \psi) \equiv \neg\phi \wedge \neg\psi$$

$$\neg\,\forall x\ \phi \equiv \exists x\ \neg\phi$$

$$\neg\,\exists x\ \phi \equiv \forall x\ \neg\phi$$

$$\neg\neg\phi \equiv \phi$$

3. Standardise variables

   $(\forall x\ P(x)) \vee (\exists x\ Q(x))$

   becomes $(\forall x\ P(x)) \vee (\exists y\ Q(y))$

Conversion into Conjunctive Normal Form

4. Skolemise
   - $\exists x\ P(x) \Rightarrow P(a)$
   - $\forall x \exists y\ P(x,\ y) \Rightarrow \forall x\ P(x,$
   - $\forall x \forall y \exists z\ P(x,\ y,\ z) \Rightarrow \forall x \forall y\ P(x,\ y,\ f(x,\ y))$
5. Drop universal quantifiers
6. Distribute $\wedge$ over $\vee$

$$(\phi \wedge \psi) \vee \chi \equiv (\phi \vee \chi) \wedge (\psi \vee \chi)$$

7. Flatten nested conjunctions and disjunctions

$$(\phi \wedge \psi) \wedge \chi \equiv \phi \wedge \psi \wedge \chi; (\phi \vee \psi) \vee \chi \equiv \phi \vee \psi \vee \chi$$

(8. In proofs, rename variables in separate clauses — *standardise apart*)

CNF — Example 1

$\forall x[(\forall y\ P(x,\ y)) \rightarrow \neg\forall y(Q(x,$  $x,\ y))]$

1. $\forall x[\neg(\forall y\ P(x,\ y)) \vee \neg\forall y(\neg$ $\vee\ R(x,\ y))]$
2. $\forall x[(\exists y\ \neg P(x,\ y)) \vee \exists y(Q(x,\ y) \wedge \neg R(x,\ y))]$
3. $\forall x[(\exists y\ \neg P(x,\ y)) \vee \exists z(Q(x,\ z) \wedge \neg R(x,\ z))]$
4. $\forall x[\neg P(x,\ f(x)) \vee (Q(x,\ g(x)) \wedge \neg R(x,\ g(x)))]$
5. $\neg P(x,\ f(x)) \vee (Q(x,\ g(x)) \wedge \neg R(x,\ g(x)))$
6. $(\neg P(x,\ f(x)) \vee Q(x,\ g(x))) \wedge (\neg P(x,\ f(x)) \vee \neg R(x,\ g(x)))$
8. $\neg P(x,\ f(x)) \vee Q(x,\ g(x))$
   $\neg P(y,\ f(y)) \vee \neg R(y,\ g(y))$

CNF — Example 2

$\neg\exists x\forall y\forall z((P(y) \vee Q(z)) \rightarrow (P(x) \vee Q(x)))$

$\neg\exists x\forall y\forall z(\neg(P(y) \vee Q(z)) \vee (P(x) \vee Q(x)))$ [Eliminate $\rightarrow$]

$\forall x\neg\forall y\forall z(\neg(P(y) \vee Q(z)) \vee (P(x) \vee Q(x)))$ [Move $\neg$ inwards]

$\forall x\exists y\neg\forall z(\neg(P(y) \vee Q(z)) \vee (P(x) \vee Q(x)))$ [Move $\neg$ inwards]

$\forall x\exists y\exists z\neg(\neg(P(y) \vee Q(z)) \vee (P(x) \vee Q(x)))$ [Move $\neg$ inwards]

$\forall x\exists y\exists z(\neg\neg(P(y) \vee Q(z)) \wedge \neg(P(x) \vee Q(x)))$ [Move $\neg$ inwards]

$\forall x\exists y\exists z((P(y) \vee Q(z)) \wedge (\neg P(x) \wedge \neg Q(x)))$ [Move $\neg$ inwards]

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

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

Unification

- Unification takes two atomic formulae and returns a *substitution* that makes them look the same
- Example:

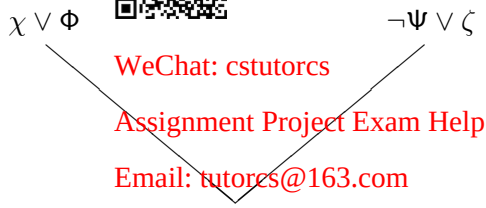
$z,\ w/f(b,\ c)\}$

- Note:
  1. Each variable has at most one associated expression
  2. No variable with an associated expression occurs within any associated expression
- $\{x/g(y),\ y/f(x)\}$ is not a substitution
- Substitution σ that makes a set of expressions identical known as a *unifier*
- Substitution σ_1 is a *more general unifier* than a substitution σ_2 if for some substitution τ, $\sigma_2 = \sigma_1\tau$.

First-Order Resolution

- *Generalised Resolution Rule*

  For clauses $\chi \vee \Phi$ and $\neg\Psi \vee \zeta$



- Where θ is a unifier for atomic formulae Φ and Ψ
- $\chi \vee \zeta$ is known as the *resolvent*

Resolution — Example 1

$\vdash \exists x(P(x) \to \forall x P(x))$

CNF($\neg\exists x(P(x) \to \forall x P(x))$

$\forall x \neg(\neg P(x) \vee \forall x\ P(x))$ [Drive $\neg$ inwards]

$\forall x(\neg\neg P(x) \wedge \neg\forall x\ P(x))$ [Drive $\neg$ inwards]

$\forall x(P(x) \wedge \exists x\ \neg P(x))$ [Drive $\neg$ inwards]

$\forall x(P(x) \wedge \exists z\ \neg P(z))$ [Standardise variables]

$\forall x(P(x) \wedge \neg P(f(x)))$ [Skolemise]

$P(x) \wedge \neg P(f(x))$ [Drop $\forall$]

---

1. $P(x)$ [$\neg$ Conclusion]
2. $\neg P(f(y))$ [$\neg$ Conclusion]
3. $P(f(y))$ [1. $\{x/f(y)\}$]
4. $\square$ [2, 3. Resolution]

Resolution — Example 2

1. $P(f(x)) \vee Q(g(x))$ [¬ Co
2. $\neg P(y)$ [¬ Conclusion]
3. $\neg Q(z)$ [¬ Conclusion]
4. $P(f(a)) \vee Q(g(a))$ [1. $\{x/a\}$]
5. $\neg P(f(a))$ [2. $\{y/f(a)\}$]
6. $\neg Q(g(a))$ [3. $\{z/g(a)\}$]
7. $Q(g(a))$ [4, 5. Resolution]
8. $\square$ [6, 7. Resolution]

Resolution — Example 3

1. $man(Marcus)$ [Premise]
2. $Pompeian(Marcus)$ [Pre
3. $\neg Pompeian(x) \vee Roman(x$ nise]
4. $ruler(Caesar)$ [Premise]
5. $\neg Roman(y) \vee loyaltyto(y, Caesar) \vee hate(y, Caesar)$ [Premise]
6. $loyaltyto(z, f(z))$ [Premise]
7. $\neg man(w) \vee \neg ruler(u) \vee \neg tryassassinate(w, u) \vee loyaltyto(w, u)$ [Premise]
8. $tryassassinate(Marcus, Caesar)$ [Premise]
9. $\neg hate(Marcus, Caesar)$ [¬ Conclusion]
10. $\neg Roman(Marcus) \vee loyaltyto(Marcus, Caesar) \vee hate(Marcus, Caesar)$ [5. $\{y/Marcus\}$]
11. $\neg Roman(Marcus) \vee loyaltyto(Marcus, Caesar)$ [9, 10. Resolution]

Resolution — Example 3

12. $\neg Pompeian(Marcus) \vee R$ [illegible] $arcus)$ [3. $\{x/Marcus\}$]
13. $loyaltyto(Marcus, Caesar) \vee$ [illegible] $mpeian(Marcus)$ [11, 12. Resolution]
14. $loyaltyto(Marcus, Caesar)$ [illegible] 3. Resolution]
15. $\neg man(Marcus) \vee \neg ruler(Caesar) \vee \neg tryassassinate(Marcus, Caesar) \vee \neg loyaltyto(Marcus, Caesar)$ [7. $\{w/Marcus, u/Caesar\}$]
16. $\neg man(Marcus) \vee \neg ruler(Caesar) \vee \neg tryassassinate(Marcus, Caesar)$ [14, 15. Resolution]
17. $\neg ruler(Caesar) \vee \neg tryassassinate(Marcus, Caesar)$ [1, 16. Resolution]
18. $\neg tryassassinate(Marcus, Caesar)$ [4, 17. Resolution]
19. $\square$ [8, 18. Resolution]

Soundness and Completeness

- Resolution is
  - *sound* (if $\lambda \vdash \rho$, then $\lambda \models \rho$)
  - *complete* (if $\lambda \models \rho$, then $\lambda \vdash \rho$)

Decidability

- First-order logic is not decidable
- How would you prove this?

Conclusion

- First-order logic allows us to speak about objects, properties of objects and relationships between ob
- It also allows quantificati variables
- First-order logic is quite an expressive knowledge representation language; much more so than propositional logic
- However, we do need to add things like equality if we wish to be able to do things like counting
- We have also traded expressiveness for decidability
- How much of a problems is this?
- If we add (Peano) axioms for mathematics, then we encounter Gödel's famous *incompleteness theorem* (which is beyond the scope of this course)