Artificial Intelligence

10. Predicate Logic Reasoning, Part I: Basics

Do *You* Think About the World in Terms of "Propositions"?

Jörg Hoffmann, Daniel Fiser, Daniel Höller, Sophia Saller

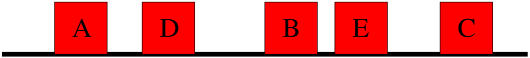
SAARLAND UNIVERSITY
COMPUTER SCIENCE

Summer Term 2022

Agenda

Let's Talk About Blocks, Baby . . .

Dear students: What do you see here?



You say: "All blocks are red"; "All blocks are on the table"; "A is a block".

And now: Say it in propositional logic!

$\rightarrow$ "isRedA","isRedB", . . . , "onTableA", "onTableB", . . . , "isBlockA", . . .

Wait a sec! Why don't we just say, e.g., "AllBlocksAreRed" and "isBlockA"?

$\rightarrow$ Could we conclude that A is red? No. These statements are atomic (just strings); their inner structure ("all blocks", "is a block") is not captured.

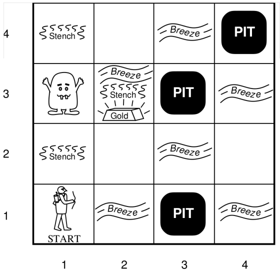
$\rightarrow$ Predicate Logic extends propositional logic with the ability to explicitly speak about objects and their properties.

$\rightarrow$ Variables ranging over objects, predicates describing object properties, . . .

$\rightarrow$ "$\forall x[Block(x) \rightarrow Red(x)]$"; "$Block(A)$"

$\rightarrow$ We consider first-order logic, and will abbreviate PL1.

Let's Talk About the Wumpus Instead?



Percepts: $[\textit{Stench}, \textit{Breeze}, \textit{Glitter}, \textit{Bump}, \textit{Scream}]$

- Cell adjacent to Wumpus: *Stench* (else: *None*).
- Cell adjacent to Pit: *Breeze* (else: *None*).
- Cell that contains gold: *Glitter* (else: *None*).
- You walk into a wall: *Bump* (else: *None*).
- Wumpus shot by arrow: *Scream* (else: *None*).

Say, in propositional logic: "Cell adjacent to Wumpus: *Stench*."

- $W_{1,1} \rightarrow S_{1,2} \wedge S_{2,1}$
- $W_{1,2} \rightarrow S_{2,2} \wedge S_{1,1} \wedge S_{1,3}$
- $W_{1,3} \rightarrow S_{2,3} \wedge S_{1,2} \wedge S_{1,4}$
- ...

$\rightarrow$ The propositional formulation typically is way too large to write (by hand).

$\rightarrow$ PL1 solution: "$\forall x[Wumpus(x) \rightarrow \forall y[Adjacent(x,y) \rightarrow Stench(y)]]$"

Blocks/Wumpus, Who Cares? Let's Talk About Numbers!

→ Even worse!

Example "Integers": (A limited vocabulary to talk about these)

- The objects: $\{1, 2, 3, \dots\}$.
- Predicate 1: "$Even(x)$" should be true iff x is even.
- Predicate 2: "$Equals(x, y)$" should be true iff $x = y$.
- Function: $Succ(x)$ maps x to $x + 1$.

Old problem: Say, in propositional logic, that "$1 + 1 = 2$".

→ Inner structure of vocabulary is ignored (cf. "AllBlocksAreRed").

→ PL1 solution: "$Equals(Succ(1), 2)$".

New problem: Say, in propositional logic, "if x is even, so is $x + 2$".

→ It is impossible to speak about infinite sets of objects!

→ PL1 solution: "$\forall x[Even(x) \to Even(Succ(Succ(x)))]$".

Now We're Talking . . .

$$\forall y, x_1, x_2, x_3 \; [Equals(Plus(PowerOf(x_1, y), PowerOf(x_2, y)), PowerOf(x_3, y)) \rightarrow (Equals(y, 1) \vee Equals(y, 2))]$$

Theorem proving in PL1! Arbitrary theorems, in principle.

Fermat's last theorem is of course infeasible, but interesting theorems can and have been proved automatically.
See `http://en.wikipedia.org/wiki/Automated_theorem_proving`.

Note: Need to axiomatize "Plus", "PowerOf", "Equals".
See `http://en.wikipedia.org/wiki/Peano_axioms`

What Are the Practical Relevance/Applications?

. . . even asking this question is a sacrilege: (Quotes from Wikipedia)

"In Europe, logic was first developed by Aristotle. Aristotelian logic became widely accepted in science and mathematics."

"The development of logic since Frege, Russell, and Wittgenstein had a profound influence on the practice of philosophy and the perceived nature of philosophical problems, and Philosophy of mathematics."

"During the later medieval period, major efforts were made to show that Aristotle's ideas were compatible with Christian faith." In other words: the Catholic church decreed for a long time that Aristotle's ideas were *in*compatible with Christian faith.

What Are the Practical Relevance/Applications?

You're asking it anyhow?

- Logic programming. Prolog et al.
- Databases. Deductive databases where elements of logic allow to conclude additional facts. Logic is tied deeply with database theory.
- Semantic technology. Large trend since 2 decades. Use PL1 fragments to annotate data sets, facilitating their use and analysis.

  $\rightarrow$ Prominent PL1 fragment: Web Ontology Language OWL.

  $\rightarrow$ Prominent data set: The WWW. ($\rightarrow$ Semantic Web)

Assorted quotes on Semantic Web and OWL:

"The brain of humanity."

"The Semantic Web will never work."

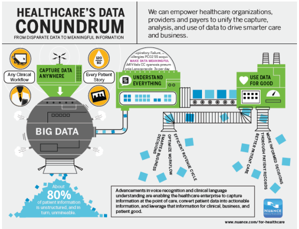
(A Few) Semantic Technology Applications

Semantic Queries

Jeopardy (IBM Watson)

Context-Aware Apps

Healthcare Data



Our Agenda for This Topic

→ Our treatment of the topic "Predicate Logic Reasoning" consists of Chapters 10 and 11.

- **This Chapter:** Basic definitions and concepts; normal forms.

  → Sets up the framework and basic operations.

- **Chapter 11:** Compilation to propositional reasoning; unification; lifted resolution.

  → Algorithmic principles for reasoning about predicate logic.

Our Agenda for This Chapter

- **Syntax:** How to write PL1 formulas?
  $\rightarrow$ Obviously required.
- **Semantics:** What is the meaning of PL1 formulas?
  $\rightarrow$ Obviously required.
- **Normal Forms:** What are the basic normal forms, and how to obtain them?
  $\rightarrow$ Needed for algorithms, which are defined on these normal forms.

The Alphabet of PL1

General symbols:

- Variables: $x, x_1, x_2, \dots, x', x'', \dots, y, \dots, z, \dots$
- Truth/Falseness: $\top$, $\bot$. (As in propositional logic)
- Operators: $\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$. (As in propositional logic)
- Quantifiers: $\forall$, $\exists$.
  $\rightarrow$ Precedence: $\neg \;>\; \forall, \exists \;>\; \dots$ (we'll be using brackets).

Application-specific symbols:

- Constant symbols ("object", e.g., $BlockA$, $BlockB, a, b, c, \dots$)
- Predicate symbols, arity ≥ 1 (e.g., $Block(.)$, $Above(., .)$)
- Function symbols, arity ≥ 1 (e.g., $WeightOf(.)$, $Succ(.)$)

Definition (Signature). *A signature Σ in predicate logic is a finite set of constant symbols, predicate symbols, and function symbols.*

$\rightarrow$ In mathematics, Σ can be infinite; not considered here.

Our "Silly Running Example": Lassie & Garfield

Constant symbols: $Lassie$, $Garfield$, $Bello$, $Lasagna$, $\dots$

Predicate symbols: $Dog(.)$, $Cat(.)$, $Eats(.,.)$, $Chases(.,.)$, $\dots$

Function symbols: $FoodOf(.)$, $\dots$

Example: $\forall x[Dog(x) \rightarrow \exists y\, Chases(x,y)]$, which in words means "Every dog chases something".

[We'll be showing the Lassie & Garfield example in this color and square brackets all over the place.]

Syntax of PL1

→ Terms represent objects:

Definition (Term). *Let Σ be a signature. Then:*

1. *Every variable and every constant symbol is a Σ-term.* $[x,\ \mathit{Garfield}]$
2. *If $t_1, t_2, \dots, t_n$ are Σ-terms and $f \in \Sigma$ is an n-ary function symbol, then $f(t_1, t_2, \dots, t_n)$ also is a Σ-term.* $[\mathit{FoodOf}(x)]$

Terms without variables are ground terms. $[\mathit{FoodOf}(\mathit{Garfield})]$

→ For simplicity, we usually don't write the "Σ-".

→ Atoms represent atomic statements about objects:

Definition (Atom). *Let Σ be a signature. Then:*

1. *$\top$ and $\bot$ are Σ-atoms.*
2. *If $t_1, t_2, \dots, t_n$ are terms and $P \in \Sigma$ is an n-ary predicate symbol, then $P(t_1, t_2, \dots, t_n)$ is a Σ-atom.* $[\mathit{Chases}(\mathit{Lassie}, y)]$

Atoms without variables are ground atoms. $[\mathit{Chases}(\mathit{Lassie}, \mathit{Garfield})]$

Syntax of PL1, ctd.

$\rightarrow$ Formulas represent complex statements about objects:

Definition (Formula). *Let Σ be a signature. Then:*

1. *Each Σ-atom is a Σ-formula.*
2. *If φ is a Σ-formula, then so is $\neg\varphi$.*

The formulas that can be constructed by rules 1. and 2. are literals.

If φ and ψ are Σ-formulas, then so are:

3. $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$, *and* $\varphi \leftrightarrow \psi$.

If φ is a Σ-formula and x is a variable, then

4. $\forall x \varphi$ *is a Σ-formula* ("Universal Quantification").
5. $\exists x \varphi$ *is a Σ-formula* ("Existential Quantification").

$\rightarrow$ [E.g., $Cat(Garfield) \vee \neg Cat(Garfield)$; and $\exists x[Eats(Garfield, x)]$]

Alternative Notation

Here	**Elsewhere**
$\neg\varphi$	$\sim\varphi$ $\quad \overline{\varphi}$
$\varphi \wedge \psi$	$\varphi \& \psi$ $\quad \varphi \bullet \psi$ $\quad \varphi, \psi$
$\varphi \vee \psi$	$\varphi \vert \psi$ $\quad \varphi; \psi$ $\quad \varphi + \psi$
$\varphi \rightarrow \psi$	$\varphi \Rightarrow \psi$ $\quad \varphi \supset \psi$
$\varphi \leftrightarrow \psi$	$\varphi \Leftrightarrow \psi$ $\quad \varphi \equiv \psi$
$\forall x \varphi$	$(\forall x)\varphi \wedge x\varphi$
$\exists x \varphi$	$(\exists x)\varphi \vee x\varphi$

Questionnaire

Example "Animals" Σ: Constant symbols $\{Lassie, Garfield, Bello, Lasagna\}$; predicate symbols $\{Dog(.), Cat(.), Eats(.,.), Chases(.,.)\}$; funtion symbols $\{FoodOf(.)\}$.

Question!

Which of these are Σ-formulas?

(A): $\forall x[Chases(x, Garfield) \rightarrow Chases(Lassie, x)]$

(B): $Eats(Bello, Cat(Garfield))$

(C): $\forall x[(Dog(x) \wedge Eats(x, Lasagna)) \rightarrow \exists y(Cat(y) \wedge Chases(y, x))]$

(D): $\exists x[Dog(x) \wedge Eats(x, Lasagna) \forall y(Cat(y) \rightarrow Chases(y, x))]$

$\rightarrow$ (A), (C): Yes.

$\rightarrow$ (B): No, we can't nest predicates.

$\rightarrow$ (D): No, missing a connective between "$Eats(x, Lasagna)$" and "$\forall y(Cat(y) \rightarrow Chases(y, x))$".

Questionnaire, ctd.

Example "Integers" Σ: Constant symbols $\{1, 2, 3, \dots\}$; predicate symbols $\{Even(.), Equals(., .)\}$; funtion symbols $\{Succ(.)\}$.

Question!

Which of these are Σ-formulas?

(A): $\exists x[Even(x) \rightarrow Even(Succ(Succ(x)))]$.

(B): $\exists x[Even(x) \rightarrow Succ(Even(Succ(x)))]$.

(C): $Even(1) \rightarrow \forall x Equals(x, Succ(x))$.

(D): $Even(1) \rightarrow \forall 2 Equals(2, 2)$.

→ (A): Yes.

→ (B): No, we can't apply a function to a predicate.

→ (C): Yes.

→ (D): No, we can't quantify over constants.

The Meaning of PL1 Formulas

Example: $\forall x[Block(x) \rightarrow Red(x)]$, $Block(A)$

$\rightarrow$ For all objects x, if x is a block, then x is red. A is a block.

More generally: (Intuition)

- Terms represent objects. $[FoodOf(Garfield) = Lasagna]$
- Predicates represent relations on the universe. $[Dog = \{Lassie, Bello\}]$
- Universally-quantified variables: "for all objects in the universe".
- Existentially-quantified variables: "at least one object in the universe".

$\rightarrow$ Similar to propositional logic, we define interpretations, models, satisfiability, validity, . . .

Semantics of PL1: Interpretations

Definition (Interpretation). *Let Σ be a signature. A Σ-interpretation is a pair (U, I) where U, the universe, is an arbitrary non-empty set $[U = \{o_1, o_2, \dots\}]$, and I is a function, notated as superscript, so that*

1. *I maps constant symbols to elements of U: $c^I \in U$ $\quad [Lassie^I = o_1]$*
2. *I maps n-ary predicate symbols to n-ary relations over U:*
   $P^I \subseteq U^n \quad [Dog^I = \{o_1, o_3\}]$
3. *I maps n-ary function symbols to n-ary functions over U:*
   $f^I \in [U^n \mapsto U] \quad [FoodOf^I = \{(o_1 \mapsto o_4), (o_2 \mapsto o_5), \dots\}]$

$\rightarrow$ We will often refer to I as the interpretation, omitting U. Note that U may be infinite.

Definition (Ground Term Interpretation). *The interpretation of a ground term under I is $(f(t_1, \dots, t_n))^I = f^I(t_1^I, \dots, t_n^I)$.* $[(FoodOf(Lassie))^I = FoodOf^I(Lassie^I) = FoodOf^I(o_1) = o_4]$

Definition (Ground Atom Satisfaction). *Let Σ be a signature and I a Σ-interpretation. We say that I satisfies a ground atom $P(t_1, \dots, t_n)$, written $I \models P(t_1, \dots, t_n)$, iff $(t_1^I, \dots, t_n^I) \in P^I$. We also call I a model of $P(t_1, \dots, t_n)$.* $\quad [I \models Dog(Lassie)$ because $Lassie^I = o_1 \in Dog^I]$

Interpretations: Example

Example "Integers": $U = \{1, 2, 3, \dots\}$; $1^I = 1$, $2^I = 2$, $3^I = 3$, $\dots$;
$Even^I = \{2, 3, 4, 6, \dots\}$, $Equals^I = \{\langle 1, 1\rangle, \langle 2, 2\rangle, \dots\}$;
$Succ^I = \{(1 \mapsto 2), (2 \mapsto 3), \dots\}$.

Question 1: $I \models Even(2)$? Yes.

Question 2: $I \models Even(Succ(2))$? Yes! $Succ(2)^I = 3 \in Even^I$.

Note: Nobody forces us to design I in accordance with the standard meaning of the predicates. Need to axiomatize them. [Remember:

$$\forall y, x_1, x_2, x_3\ [Equals(Plus(PowerOf(x_1, y), PowerOf(x_2, y)), PowerOf(x_3, y)) \rightarrow (Equals(y, 1) \vee Equals(y, 2))]$$

$\rightarrow$ Details: `http://en.wikipedia.org/wiki/Peano_axioms`]

Question 3: $I \models Equals(x, Succ(2))$? Interpretations do not handle variables. We must fix a variable assignment first.

Semantics of PL1: Variable Assignments

Definition (Variable Assignment). *Let Σ be a signature and (U, I) a Σ-interpretation. Let X be the set of all variables. A variable assignment α is a function $\alpha : X \mapsto U$.* $[\alpha(x) = o_1]$

Definition (Term Interpretation). *The interpretation of a term under I and α is:*

1. $x^{I,\alpha} = \alpha(x)$ $[x^{I,\alpha} = o_1]$
2. $c^{I,\alpha} = c^I$ $[Lassie^{I,\alpha} = Lassie^I]$
3. $(f(t_1, \dots, t_n))^{I,\alpha} = f^I(t_1^{I,\alpha}, \dots, t_n^{I,\alpha})$
   $[(FoodOf(x))^{I,\alpha} = FoodOf^I(x^{I,\alpha}) = FoodOf^I(o_1) = o_4]$

Definition (Atom Satisfaction). *Let Σ be a signature, I a Σ-interpretation, and α a variable assignment. We say that I and α satisfy an atom $P(t_1, \dots, t_n)$, written $I, \alpha \models P(t_1, \dots, t_n)$, iff $(t_1^{I,\alpha}, \dots, t_n^{I,\alpha}) \in P^I$. We also call I and α a model of $P(t_1, \dots, t_n)$.*

$[I, \alpha \not\models Dog(FoodOf(x))\text{: } (FoodOf(x))^{I,\alpha} = o_4 \notin Dog^I]$

Semantics of PL1: Formula Satisfaction

Notation: In α^x_o we overwrite x with o in α: for $\alpha = \{(x \mapsto o_1), (y \mapsto o_2), \dots\}$, $\alpha^x_o = \{(x \mapsto o), (y \mapsto o_2), \dots\}$.

Definition (Formula Satisfaction). *Let Σ be a signature, I a Σ-interpretation, and α a variable assignment. We set:*

$I, \alpha \models \top$ *and*	$I, \alpha \not\models \bot$	
$I, \alpha \models \neg\varphi$	*iff*	$I, \alpha \not\models \varphi$
$I, \alpha \models \varphi \wedge \psi$	*iff*	$I, \alpha \models \varphi$ *and* $I, \alpha \models \psi$
$I, \alpha \models \varphi \vee \psi$	*iff*	$I, \alpha \models \varphi$ *or* $I, \alpha \models \psi$
$I, \alpha \models \varphi \rightarrow \psi$	*iff*	*if* $I, \alpha \models \varphi$, *then* $I, \alpha \models \psi$
$I, \alpha \models \varphi \leftrightarrow \psi$	*iff*	*if* $I, \alpha \models \varphi$ *if and only if* $I, \alpha \models \psi$
$I, \alpha \models \forall x \varphi$	*iff*	*for all* $o \in U$ *we have* $I, \alpha^x_o \models \varphi$
$I, \alpha \models \exists x \varphi$	*iff*	*there exists* $o \in U$ *so that* $I, \alpha^x_o \models \varphi$

If $I, \alpha \models \varphi$, we say that I and α satisfy φ (are a model of φ).

[$\varphi = \forall x[Dog(x) \rightarrow \exists y Chases(x, y)]$, $Dog^{I,\alpha} = \{Lassie^{I,\alpha}, Bello^{I,\alpha}\}$, $Chases^{I,\alpha} = \{(Lassie^{I,\alpha}, Garfield^{I,\alpha})\}$. Then $I, \alpha \not\models \varphi$ because Bello does not chase anything.]

PL1 Satisfiability etc.

Satisfiability

A PL1 formula φ is:

- satisfiable if there exist I, α that satisfy φ.
- unsatisfiable if φ is not satisfiable.
- falsifiable if there exist I, α that do not satisfy φ.
- valid if $I, \alpha \models \varphi$ holds for all I and α. We also call φ a tautology.

Entailment and Equivalence

φ entails ψ, $\varphi \models \psi$, if every model of φ is a model of ψ.
φ and ψ are equivalent, $\varphi \equiv \psi$, if $\varphi \models \psi$ and $\psi \models \varphi$.

Attention: In presence of free variables!

$\rightarrow$ Do we have $Dog(x) \models Dog(y)$? No. Example: $Dog^I = \{o_1\}$, $\alpha = \{(x \mapsto o_1), (y \mapsto o_2)\}$. Then $I, \alpha \models Dog(x)$ but $I, \alpha \not\models Dog(y)$.

Free and Bound Variables

$$\varphi := \forall x[R(\boxed{y}, \boxed{z}) \wedge \exists y(\neg P(y,x) \vee R(y, \boxed{z}))]$$

Definition (Free Variables). *By $vars(e)$, where e is either a term or a formula, we denote the set of variables occuring in e. We set:*

$$\begin{array}{lcl} free(P(t_1,\dots,t_n)) & := & vars(t_1) \cup \dots \cup vars(t_n) \\ free(\neg\varphi) & := & free(\varphi) \\ free(\varphi * \psi) & := & free(\varphi) \cup free(\psi) \text{ for } * \in \{\wedge, \vee, \rightarrow, \leftrightarrow\} \\ free(\forall x\varphi) & := & free(\varphi) \setminus \{x\} \\ free(\exists x\varphi) & := & free(\varphi) \setminus \{x\} \end{array}$$

$free(\varphi)$ are the free variables of φ. φ is closed if $free(\varphi) = \emptyset$.

→ In the above φ, which variable appearances are free? The boxed ones.

→ Knowledge Base (aka logical theory) = set of closed formulas. From now on, we asume that φ is closed.

→ We can ignore α, and will write $I \models \varphi$ instead of $I, \alpha \models \varphi$.

Questionnaire

Example "Animals": $U = \{o_1, o_2, o_3, o_4, o_5\}$; $Lassie^I = o_1$, $Garfield^I = o_2$, $Bello^I = o_3$, $Lasagna^I = o_4$, $Chappi^I = o_5$; $Dog^I = \{o_1, o_3\}$, $Cat^I = \{o_2\}$, $Eats^I = \{(o_1, o_4), (o_2, o_4), (o_3, o_5)\}$, $Chases^I = \{(o_1, o_3), (o_3, o_2), (o_2, o_1)\}$.

Question!

For which of these φ do we have $I \models \varphi$?

(A): $\forall x[Chases(x, Garfield) \rightarrow Chases(Lassie, x)]$

(B): $Eats(Bello, Cat(Garfield))$

(C): $\forall x[(Dog(x) \wedge Eats(x, Lasagna)) \rightarrow \exists y(Cat(y) \wedge Chases(y, x))]$

(D): $\exists x[Dog(x) \wedge Eats(x, Lasagna) \wedge \forall y(Cat(y) \rightarrow Chases(y, x))]$

→ (A): Yes. (Only Bello chases Garfield; Lassie chases Bello.)

→ (B): Not a formula (cf. slide 17).

→ (C): Yes. (The only dog eating Lasagna is Lassie; Garfield chases Lassie.)

→ (D): Yes. (Lassie is a dog eating Lasagna; it is chased by the only cat, Garfield.)

Questionnaire, ctd.

Example "Integers": $U = \{1, 2, 3, \dots\}$; $1^I = 1$, $2^I = 2$, $\dots$; $Even^I = \{2, 4, 6, \dots\}$, $Equals^I = \{\langle 1, 1\rangle, \langle 2, 2\rangle, \dots\}$; $Succ^I = \{(1 \mapsto 2), (2 \mapsto 3), \dots\}$.

Question!

For which of these φ do we have $I \models \varphi$?

(A): $\exists x[Even(x) \rightarrow Even(Succ(Succ(x)))]$.

(B): $\exists x[Even(x) \rightarrow Succ(Even(Succ(x)))]$.

(C): $Even(1) \rightarrow \forall x Equals(x, Succ(x))$.

(D): $Even(1) \rightarrow \forall 2 Equals(2, Succ(2))$.

→ (A): Yes: $x = 2$ does the job. Actually we can strengthen the formula to $\forall x[Even(x) \rightarrow Even(Succ(Succ(x)))]$.

→ (B): Not a formula (cf. slide 18).

→ (C): Yes: While $\forall x Equals(x, Succ(x))$ is false, $Even(1)$ is false as well and thus the overall implication is true.

→ (D): Not a formula (cf. slide 18).

Before We Begin

Why normal forms?

- Convenient: full syntax when describing the problem at hand.
- Not convenient: full syntax when solving the problem.

"Solving the problem"? Decide satisfiability!

$\rightarrow$ Tackles deduction as well as other applications. (Same as in propositional logic, cf. **Chapter 9**.)

Which normal forms?

- Prenex normal form: Move all quantifiers up front.
- Skolem normal form: Prenex + remove all existential quantifiers while preserving satisfiability.
- Clausal normal form: Skolem + CNF transformation while preserving satisfiability.

Prenex Normal Form: Step 1

quantifier prefix + (quantifier-free) matrix

$$Qx_1Qx_2Qx_3\dots Qx_n\ \varphi$$

Step 1: Eliminate $\rightarrow$ and $\leftrightarrow$, move $\neg$ inwards

1. $(\varphi \leftrightarrow \psi) \equiv [(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)]$ (Eliminate "$\leftrightarrow$")
2. $(\varphi \rightarrow \psi) \equiv (\neg\varphi \vee \psi)$ (Eliminate "$\rightarrow$")
3. $\neg(\varphi \wedge \psi) \equiv (\neg\varphi \vee \neg\psi)$ and $\neg(\varphi \vee \psi) \equiv (\neg\varphi \wedge \neg\psi)$
   $\neg\forall x\varphi \equiv \exists x\neg\varphi$ and $\neg\exists x\varphi \equiv \forall x\neg\varphi$ (Move "$\neg$" inwards)

Example: $\neg\forall x[(\forall x P(x)) \rightarrow R(x)]$

Eliminate $\rightarrow$ and $\leftrightarrow$: $\neg\forall x[\neg(\forall x P(x)) \vee R(x)]$.

Move $\neg$ across first quantifier: $\exists x\neg[\neg(\forall x P(x)) \vee R(x)]$.

Move $\neg$ inwards: $\exists x[(\forall x P(x)) \wedge \neg R(x)]$.

Prenex Normal Form: Step 2

quantifier prefix + (quantifier-free) matrix

$$Qx_1Qx_2Qx_3 \dots Qx_n \; \varphi$$

Step 2: Move quantifiers outwards

- $(\forall x \varphi) \wedge \psi \equiv \forall x(\varphi \wedge \psi)$, if x not free in ψ.
- $(\forall x \varphi) \vee \psi \equiv \forall x(\varphi \vee \psi)$, if x not free in ψ.
- $(\exists x \varphi) \wedge \psi \equiv \exists x(\varphi \wedge \psi)$, if x not free in ψ.
- $(\exists x \varphi) \vee \psi \equiv \exists x(\varphi \vee \psi)$, if x not free in ψ.

Example "Animals": $\forall x[\neg Dog(x) \vee \exists y \, Chases(x, y)]$
$\rightarrow$ Move "$\exists y$" outwards: $\forall x \exists y[\neg Dog(x) \vee Chases(x, y)]$.

Example: $\exists x[(\forall x P(x)) \wedge \neg R(x)]$
$\rightarrow$ We can't move "$\forall x$" outwards because x is free in "$\psi = \neg R(x)$".

Prenex Normal Form: Variable Renaming

Notation: If x is a variable, t a term, and φ a formula, then the instantiation of x with t in φ, written $\varphi\frac{x}{t}$, replaces all free appearances of x in φ by t. If $t = y$ is a variable, then $\varphi\frac{x}{y}$ renames x to y in φ.

Lemma. *If $y \notin vars(\varphi)$, then $\forall x \varphi \equiv \forall y \varphi\frac{x}{y}$ and $\exists x \varphi \equiv \exists y \varphi\frac{x}{y}$.*

Step 2 Addition: Rename variables if needed

For each Step 2 rule: If x is free in ψ, then rename x in $(\forall x \varphi)$ respectively $(\exists x \varphi)$ to some new variable y. Then, the rule can be applied.

Example: $\exists x[(\forall x P(x)) \land \neg R(x)]$

$\rightarrow$ Rename $\frac{x}{y}$ in $(\forall x P(x))$: $\exists x[(\forall y P(y)) \land \neg R(x)]$.

$\rightarrow$ Move $\forall y$ outwards: $\exists x \forall y[P(y) \land \neg R(x)]$.

Theorem. *There exists an algorithm that, for any PL1 formula φ, efficiently (i.e., in polynomial time) calculates an equivalent formula in prenex normal form.* (Proof: We just outlined that algorithm.)

Skolem Normal Form

universal prefix + (quantifier-free) matrix

$$\forall x_1 \forall x_2 \forall x_3 \dots \forall x_n \; \varphi$$

Theorem (Skolem). *Let $\varphi = \forall x_1 \dots \forall x_k \exists y \psi$ be a closed PL1 formula in prenex normal form, such that all quantified variables are pairwise distinct, and the k-ary function symbol f does not appear in φ. Then φ is satisfiable if and only if $\forall x_1 \cdots \forall x_k \psi \frac{y}{f(x_1,\dots,x_k)}$ is satisfiable.* (Proof omitted.)

Note: Here, "0-ary function symbol" = constant symbol.

Transformation to Skolem normal form

Rename quantified variables until distinct. Then iteratively remove the outmost existential quantifier, using Skolem's theorem.

Example. $\exists x \forall y \exists z R(x, y, z)$ is transformed to:

$\rightarrow$ Remove "$\exists x$": $\forall y \exists z R(f, y, z)$. Remove "$\exists z$": $\forall y R(f, y, g(y))$.

$\rightarrow$ Note the arity/arguments of f vs. g: "$x_1 \dots x_k$" in the above!

Skolem Normal Form, ctd.

Notation: A formula is in Skolem normal form (SNF) if it is in prenex normal form and has no existential quantifiers.

Theorem. *There exists an algorithm that, for any closed PL1 formula φ, efficiently calculates an SNF formula that is satisfiable iff φ is. We denote that formula φ^*.* (Proof: We just outlined that algorithm.)

Example 1: (a) $\varphi_1 = \exists y \forall x[\neg Dog(x) \vee Chases(x, y)]$: "There exists a y chased by every dog x". (b) $\varphi_1^* = \forall x[\neg Dog(x) \vee Chases(x, f)]$: "The object named f is chased by every dog x".

Example 2: (a) $\varphi_2 = \forall x \exists y[\neg Dog(x) \vee Chases(x, y)]$: "For every dog x, there exists y chased by x". (b) $\varphi_2^* = \forall x[\neg Dog(x) \vee Chases(x, f(x))]$: "For every dog x, we can interprete $f(x)$ with a y chased by x".

→ Satisfying $\forall x_1 \dots \forall x_k \exists y \psi$ by choosing suitable objects for y = satisfying $\forall x_1 \cdots \forall x_k \psi \frac{y}{f(x_1, \dots, x_k)}$ by choosing suitable values for $f(x_1, \dots, x_k)$.

Note: φ^* is *not* equivalent to φ. It is more specific: φ^* implies φ, but not vice versa. Example: $\varphi = \exists x\ Dog(x)$, $\varphi^* = Dog(f)$; $Dog^I = \{Lassie, Bello\}$, $f^I = Garfield$.

Questionnaire

Question!

Which are Skolem normal forms of
$\forall x \exists y[\neg Dog(x) \vee \neg Eats(x, Lasagna) \vee (Cat(y) \wedge Chases(y, x))]$**?**

(A): $\forall x \exists y[\neg Dog(x) \vee \neg Eats(x, Lasagna) \vee (Cat(f(x)) \wedge Chases(f(x), x))]$

(B): $\forall x[\neg Dog(x) \vee \neg Eats(x, Lasagna) \vee (Cat(f) \wedge Chases(f, x))]$

(C): $\forall x[\neg Dog(x) \vee \neg Eats(x, Lasagna) \vee (Cat(f(x)) \wedge Chases(f(x), x))]$

(D): $\forall x[\neg Dog(x) \vee \neg Eats(x, Lasagna) \vee (Cat(g(x)) \wedge Chases(g(x), x))]$

→ (A): No, we need to remove the existential quantifier over y. (B): No, f needs x as an argument (else we are saying that there is one unique cat chasing all dogs that eat Lasagna). (C): Yes: "$\exists y$" is removed, and "y" is replaced by a new function symbol with argument x. (D): Same as (C).

→ Note the different function symbols in (C) and (D): The Skolem normal form is unique up to renaming of function symbols.

Clausal Normal Form

universal prefix + disjunction of literals

$$\forall x_1 \forall x_2 \forall x_3 \dots \forall x_n (l_1 \vee \dots \vee l_n)$$

$\rightarrow$ Written $\{l_1, \dots, l_n\}$.

Transformation to clausal normal form

1. Transform to SNF: $\forall x_1 \forall x_2 \forall x_3 \cdots \forall x_n \; \varphi$.
2. Transform φ to satisfiability-equivalent CNF ψ. (Same as in propositional logic.)
3. Write as set of clauses: One for each disjunction in ψ.
4. Standardize variables apart: Rename variables so that each occurs in at most one clause. (Needed for PL1 resolution, **Chapter 11**.)

Theorem. *There exists an algorithm that, for any closed PL1 formula φ, efficiently calculates a formula in clausal normal form that is satisfiable iff φ is.* (Proof: We just outlined that algorithm.)

All 3 Transformations: Example

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

→ Means what? "Everyone who loves all animals is loved by someone."

1. **Eliminate equivalences and implications:**

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

2. **Move negation inwards:**

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

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

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

3. **Move quantifiers outwards:** → Prenex normal form.

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

   → Note: y is **not** free in "$\exists y Loves(y,x)$".

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

   → Note: y **is** free in "$(Animal(y) \wedge \neg Loves(x,y))$".

All 3 Transformations: Example, ctd.

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

4. **Make quantified variables distinct:** (Nothing to do)
5. **Remove existential quantifiers:** → Skolem normal form.
   $\forall x \exists z[(Animal(f(x)) \wedge \neg Loves(x, f(x))) \vee Loves(z, x)]$
   $\forall x[(Animal(f(x)) \wedge \neg Loves(x, f(x))) \vee Loves(g(x), x)]$
6. **Transform to CNF:**
   $\forall x[(Animal(f(x)) \vee Loves(g(x), x)) \wedge (\neg Loves(x, f(x)) \vee Loves(g(x), x))]$
7. **Write as set of clauses:**
   $\{\{Animal(f(x)), Loves(g(x), x)\}, \{\neg Loves(x, f(x)), Loves(g(x), x)\}\}$
8. **Standardize variables apart:** → Clausal normal form.
   $\{\{Animal(f(x)), Loves(g(x), x)\}, \{\neg Loves(y, f(y)), Loves(g(y), y)\}\}$

Questionnaire

Example "Animals" (simplified): $U = \{o_1, o_2, o_3\}$; $Lassie^I = o_1$, $Garfield^I = o_2$, $Bello^I = o_3$; $Dog^I = \{o_1, o_3\}$, $Chases^I = \{(o_1, o_3), (o_3, o_2)\}$.

Question!

Which of these φ (1) have $I \models \varphi$, or (2) are satisfiable by extending I with a suitable interpretation of f?

(A): $\forall x \exists y [Dog(x) \rightarrow Chases(x, y)]$

(B): $\exists y \forall x [Dog(x) \rightarrow Chases(x, y)]$

(C): $\forall x [Dog(x) \rightarrow Chases(x, f(x))]$

(D): $\forall x [Dog(x) \rightarrow Chases(x, f)]$

$\rightarrow$ (A): Yes, (1) because Bello chases Garfield and Lassie chases Bello. (B): No, because Bello respectively Lassie chase *different* y. (C): Yes, (2) by choosing $f(o_3) := o_2$ and $f(o_1) := o_3$ (cf. (A)). (D): No, because f has no argument (cf. (B)).

$\rightarrow$ Note that (C) is a SNF for (A), and (D) is a SNF for (B). Note also that (D) is a "flawed SNF" for (A) where we forgot to give f the argument x. (Compare slide 35)

Summary

- Predicate logic allows to explicitly speak about objects and their properties. It is thus a more natural and compact representation language than propositional logic; it also enables us to speak about infinite sets of objects.
- Logic has thousands of years of history. A major current application in AI is Semantic Technology.
- First-order predicate logic (PL1) allows universal and existential quantification over objects.
- A PL1 interpretation consists of a universe U and a function I mapping constant symbols/predicate symbols/function symbols to elements/relations/functions on U.
- In prenex normal form, all quantifiers are up front. In Skolem normal form, additionally there are no existential quantifiers. In clausal normal form, additionally the formula is in CNF.
- Any PL1 formula can efficiently be brought into a satisfiability-equivalent clausal normal form.

Reading

- *Chapter 8: First-Order Logic*, Sections 8.1 and 8.2 [Russell and Norvig (2010)].

  Content: A less formal account of what I cover in “Syntax” and “Semantics”. Contains different examples, and complementary explanations. Nice as additional background reading.

  Sections 8.3 and 8.4 provide additional material on using PL1, and on modeling in PL1, that I don't cover in this lecture. Nice reading, not required for exam.

- *Chapter 9: Inference in First-Order Logic*, Section 9.5.1 [Russell and Norvig (2010)].

  Content: A very brief (2 pages) description of what I cover in “Normal Forms”. Much less formal; I couldn't find where (if at all) RN cover transformation into prenex normal form. Can serve as additional reading, can't replace the lecture.

References I

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (Third Edition)*. Prentice-Hall, Englewood Cliffs, NJ, 2010.