Warm-Up Problem

Let α and β be two well formed formulas. Prove or disprove the following:

$$\text{If } \alpha \vDash (\beta \to \alpha) \text{ then } \emptyset \vdash (\alpha \to (\beta \to \alpha))$$

Predicate Logic: Informal Introduction, Syntax and Translation

Carmen Bruni

Lecture 10

These slides are based on materials from Alice Gao's slides which in turn were based on CS245 at UWaterloo and CPSC121 at UBC.

Based on slides by Jonathan Buss, Lila Kari, Anna Lubiw and Steve Wolfman with thanks to B. Bonakdarpour, A. Gao, D. Maftuleac, C. Roberts, R. Trefler, and P. Van Beek

Learning Goals

- Describe the structure of Predicate Logic; this includes constants, variables, function symbols, terms and predicates.
- Translate sentences from English into Predicate Logic and vice versa
- Draw a Parse Tree for a Predicate Logic Formula
- Define free and bound variables and determine when variables in a formula are of which type
- Define a substitution and be able to perform substitutions with or without variable capture in formulas.

What can't we express using propositional logic?

Can we express the following ideas using propositional logic?

- Translate this sentence: Alice is married to Jay and Alice is not married to Leon.
- Translate this sentence: Every bear likes honey.
- Define what it means for a natural number to be prime.

What can't we express using propositional logic?

A few things that are difficult to express using propositional logic:

- Relationships among individuals: Alice is married to Jay and Alice is not married to Leon.
- Generalizing patterns: Every bear likes honey.
- Infinite domains: Define what it means for a natural number to be prime.

We can use predicate logic (first-order logic) to express all of these.

Elements of predicate logic

Predicate logic generalizes propositional logic.

New things in predicate logic:

- Domains
- Constants, Variables and Function symbols
- Terms
- Predicates
- Quantifiers

Everything will be done with respect to a language $\mathcal{L}$, that is, the set of allowable symbols will be symbols in our language.

Domains

A *domain* is a non-empty set of objects. It is a world that our statement is situated within.

Examples of domains: natural numbers, people, animals, etc.

Why is it important to specify a domain? *The same statement can have different truth values in different domains.*

Consider this statement: There exists a number whose square is 2.

- If our domain is the set of natural numbers, is this statement true or false?
- If our domain is the set of real numbers, is this statement true or false?

Objects in a domain

Constants: concrete objects in the language (interpreted as domain elements eventually)

- Natural numbers: 0, 6, 100, ...
- Alice, Bob, Eve, ...
- Animals: Winnie the Pooh, Mickey Mouse, Simba, ...

Variables: language elements; placeholders for concrete objects, e.g. x, y, z.

A variable lets us refer to an object without specifying which particular object it is.

Functions

Function Symbol: For now, just a symbol f. We say such a function has *arity* n and sometimes denote this by $f^{(n)}$. Later we will attach meaning so that a function symbol behaves like a function in the mathematical sense mapping from n copies of the domain into the domain:

$$f : \mathcal{D}^n \rightarrow \mathcal{D}$$

Terms

Terms: Defined inductively as:

1. Each constant symbol is a term and each variable is a term (atomic terms).
2. If t_1, ..., t_n are terms and f is a function symbol of arity n, then $f(t_1, ..., t_n)$ is a term.
3. Nothing else is a term.

Note: Binary functions, terms and later binary predicates are sometimes denoted like $(t_1 \ f \ t_2)$ instead of $f(t_1, t_2)$. For example, we usually write $(t_1 + t_2)$ instead of $+(t_1, t_2)$.

Examples

If 0 is a constant symbol, x and y are variables and $s^{(1)}$ and $+^{(2)}$ are function symbols, then 0, x, y, $s(0)$, $s(x)$, $s(y)$, $+(x, s(y))$ and $(x + y)$ are all examples of terms.

$s(x, y)$ is not a term (s is a unary function and $(s + x)$ is not a term either as s is a function and not a term on its own).

Predicates

A *predicate* represents

- a property of an individual, or
- a relationship among multiple individuals.

An *atomic formula* (or atom) is an expression of the form

$$P(t_1, \ldots, t_n)$$

where P is an n-ary predicate symbol and each t_i is a term $(1 \leq i \leq n)$.

Note: Binary predicates are sometimes denoted like $(t_1 \ P \ t_2)$ instead of $P(t_1, t_2)$.

It helps to think of a predicate as a function mapping from n copies of the domain $\mathcal{D}^n$ into $\{\mathtt{T}, \mathtt{F}\}$, though we will actually attach this meaning to predicates later.

Examples:

- Define $L(x)$ to mean “x is a lecturer”.
  (unary predicate)
  - Alice is a lecturer: $L(\text{Alice})$
  - Mickey Mouse is not a lecturer: $(\neg L(\text{Mickey Mouse}))$
  - y is a lecturer: $L(y)$
- Define $O(x, y)$ to mean “x is older than y”.
  (binary predicate/relation)
  - Alex is older than Sam: $O(\text{Alex}, \text{Sam})$
  - a is older than b: $O(a, b)$

Representing Predicates

Mathematically, we represent a predicate by the set of all things that have the property. If S is the set of all students, then $x \in S$ means x is a student. The only restriction on a predicate is that it must be a subset of the domain.

A k-ary predicate (relation) is a set of k-tuples of domain elements. For example, the binary predicate less-than, over a domain $\mathcal{D}$, is represented by the set

$$\{ \langle x, y \rangle \in \mathcal{D}^2 \mid x < y \} .$$

We will later be interpreting predicate symbols as these relations.

Quantifiers

For how many objects in the domain is the statement true?

- The universal quantifier $\forall$: the statement is true for every object in the domain.
- The existential quantifier $\exists$: the statement is true for one or more objects in the domain.

General Formulas

We define the set of well-formed formulas of first-order logic inductively as follows.

1. A predicate (atomic formula) is a well-formed formula.
2. If α is a well-formed formula, then $(\neg\alpha)$ is a well-formed formula.
3. If α and β are well-formed formulas, and $\star$ is a binary connective symbol,
   then $(\alpha \star \beta)$ is a well-formed formula.
4. If α is a well-formed formula and x is a variable,
   then each of $(\forall x\ \alpha)$ and $(\exists x\ \alpha)$ is a well-formed formula.
5. Nothing else is a well-formed formula.

In case 4, the formula α is called the *scope* of the quantifier. The quantifier keeps the same scope if it is included in a larger formula.

The Language of Predicate Logic

The seven kinds of symbols (our language and punctuation):

1. Constant symbols. Usually $c, d, c_1, c_2, \ldots, d_1, d_2 \ldots$
2. Variables. Usually $x, y, z, \ldots x_1, x_2, \ldots, y_1, y_2 \ldots$
3. Function symbols. Usually $f, g, h, \ldots f_1, f_2, \ldots, g_1, g_2, \ldots$
4. Predicate symbols. P, Q, …P_1, P_2, …, Q_1, Q_2, …
5. Connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$
6. Quantifiers: $\forall$ and $\exists$
7. Punctuation: '(', ')', and ','

Function symbols and predicate symbols have an assigned *arity*—the number of arguments required.

The last three kinds of symbols—connectives, quantifiers, and punctuation—will have their meaning fixed by the syntax and semantics.

Constants, variables, functions and predicate symbols are not restricted. They may be assigned any meaning, consistent with their kind and arity.

Multiple Quantifiers (Exercise for Home)

Let the domain be the set of people. Let $L(x, y)$ mean that person x likes person y.

Translate the following formulas into English.

1. $(\forall x\,(\forall y\ L(x, y)))$
2. $(\exists x\,(\exists y\ L(x, y)))$
3. $(\forall x\,(\exists y\ L(x, y)))$
4. $(\exists y\,(\forall x\ L(x, y)))$

Parse Trees of Predicate Logic Formulas

New elements in the parse tree:

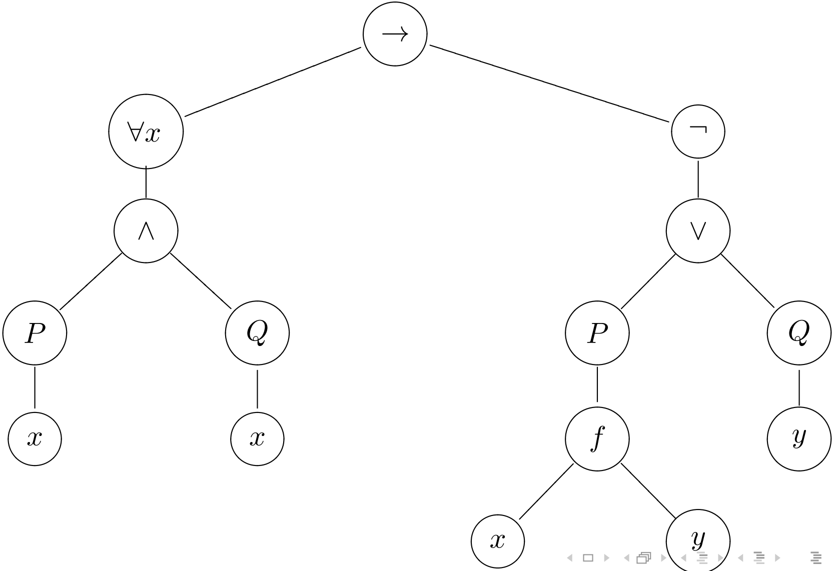
- Quantifiers $\forall x$ and $\exists y$ have one subtree, similar to the unary connective negation.
- A predicate symbol $P(t_1, t_2, \ldots, t_n)$ has a node labelled P with a sub-tree for each of the terms $t_1, t_2, \ldots, t_n$.
- A function symbol $f(t_1, t_2, \ldots, t_n)$ has a node labelled f with a sub-tree for each of the terms $t_1, t_2, \ldots, t_n$.

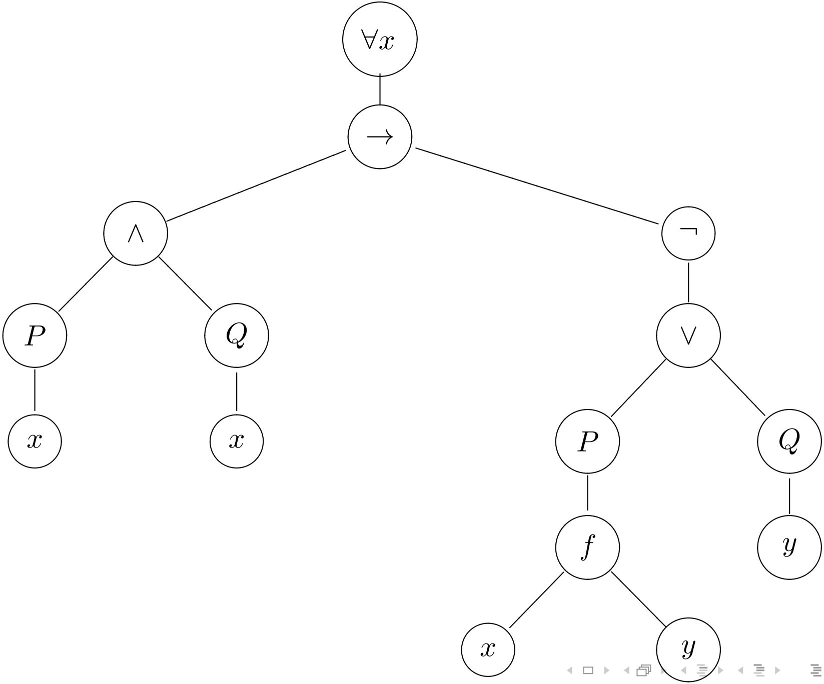
Below P and Q are unary predicate symbols and f is a binary function symbol.

Example 1: $((\forall x\,(P(x) \wedge Q(x))) \rightarrow (\neg P(f(x,y)) \vee Q(y)))$

Example 2: $(\forall x\,((P(x) \wedge Q(x)) \rightarrow (\neg(P(f(x,y)) \vee Q(y)))))$

Example 1 Solution (Example 2 on next slide)





$\forall x$
$\rightarrow$
$\wedge$
$\neg$
P
Q
$\vee$
x
x
P
Q
f
y
x
y

Evaluating a Formula

To evaluate the truth value of a formula, we need to replace the variables by concrete objects in the domain. However, we don't necessarily have to perform this substitution for every variable.

There are two types of variables in a formula:

- A variable may be *free*. To evaluate the formula, we need to replace a free variable by an object in the domain.
- A variable may be *bound by a quantifier*. The quantifier tells us how to evaluate the formula.

We need to understand *how to determine whether a variable is free/bound* and *how to replace a free variable with an object in the domain*.

Free and Bound Variables

In a formula $(\forall x\ \alpha)$ or $(\exists x\ \alpha)$, the *scope* of a quantifier is the formula α. A quantifier *binds* its variable within its scope.

An occurrence of a variable in a formula is *bound* if it lies in the scope of some quantifier of the same variable. Otherwise the occurrence of this variable is *free*.

- If a variable occurs multiple times, we need to consider each occurrence of the variable separately.
- The variable symbol immediately after $\exists$ or $\forall$ is neither free nor bound.

A formula with no free variables is called a *closed formula* or *sentence*.

Free and Bound Variables

Formally, a variable occurs free in a formula α if and only if it is a member of the set $\mathsf{FV}(\alpha)$ defined as follows.

1. If α is $P(t_1, \dots, t_k)$, then $\mathsf{FV}(\alpha) = \{\, x \mid x$ appears in some $t_i \,\}$.
2. If α is $(\neg\beta)$, then $\mathsf{FV}(\alpha) = \mathsf{FV}(\beta)$.
3. If α is $(\beta \star \gamma)$, then $\mathsf{FV}(\alpha) = \mathsf{FV}(\beta) \cup \mathsf{FV}(\gamma)$.
4. If α is $(\mathsf{Q}x\ \beta)$ (for $\mathsf{Q} \in \{\forall, \exists\}$), then $\mathsf{FV}(\alpha) = \mathsf{FV}(\beta) - \{x\}$.

A formula has the same free variables as its parts, except that a quantified variable becomes bound.

A formula with no free variables is called a *closed formula*, or a *sentence*.

Substitution

When writing natural deduction proofs in predicate logic, it is often useful to replace a variable in a formula with a term.

Suppose that the following sentences are true:

$$(\forall x\, (\mathit{Fish}(x) \to \mathit{Swim}(x)))\ (1)$$

$$\mathit{Fish}(\mathit{Nemo})\ (2)$$

To conclude that Nemo can swim, we need to replace every occurrence of the variable x in the implication $(\mathit{Fish}(x) \to \mathit{Swim}(x))$ by the term *Nemo*. This gives us

$$(\mathit{Fish}(\mathit{Nemo}) \to \mathit{Swim}(\mathit{Nemo}))\ (3)$$

By *modus ponens* on (2) and (3), we conclude that $\mathit{Swim}(\mathit{Nemo})$.

Formally, we use *substitution* to refer to the process of replacing x by *Nemo* in the formula $(\forall x\, (\mathit{Fish}(x) \to \mathit{Swim}(x)))$.

Substitution

Intuitively, $\alpha[t/x]$ answers the question,

"What happens to α if x has the value specified by term t?"

For a variable x, a term t, and a formula α, $\alpha[t/x]$ denotes the resulting formula by replacing each *free* occurrence of x in α with t. In other words, substitution *does NOT* affect *bound* occurrences of the variable.

Examples.

- If α is the formula $E(f(x))$, then $\alpha[(y+y)/x]$ is $E(f((y+y)))$.
- $\alpha[f(x)/x]$ is $E\big(f(f(x))\big)$.
- $E(f((x+y)))[y/x]$ is $E(f((y+y)))$.

- If β is $\big(\forall x\,(E(f(x)) \wedge S(x,y))\big)$, then $\beta[g(x,y)/x]$ is β, because β has no free occurrence of x.

Examples: Substitution

Example. Let β be $\big(P(x) \wedge (\exists x\ Q(x))\big)$. What is $\beta[y/x]$?

Examples: Substitution

Example. Let β be $\big(P(x) \wedge (\exists x\ Q(x))\big)$. What is $\beta[y/x]$?

$\beta[y/x]$ is $\big(P(y) \wedge (\exists x\ Q(x))\big)$. Only the free x gets substituted.

Examples: Substitution

Example. Let β be $\big(P(x) \wedge (\exists x\ Q(x))\big)$. What is $\beta[y/x]$?

$\beta[y/x]$ is $\big(P(y) \wedge (\exists x\ Q(x))\big)$. Only the free x gets substituted.

Example. What about $\beta[(y-1)/z]$, where β is $\Big(\forall x \left(\exists y \left(\left(x+y\right)=z\right)\right)\Big)$?

Examples: Substitution

Example. Let β be $\big(P(x) \wedge (\exists x\ Q(x))\big)$. What is $\beta[y/x]$?

$\beta[y/x]$ is $\big(P(y) \wedge (\exists x\ Q(x))\big)$. Only the free x gets substituted.

Example. What about $\beta[(y-1)/z]$, where β is $\Big(\forall x\,\big(\exists y\,((x+y)=z)\big)\Big)$?

At first thought, we might say $\Big(\forall x\,\big(\exists y\,((x+y)=(y-1))\big)\Big)$. But there's a problem—the free variable y in the term $(y-1)$ got "captured" by the quantifier $\exists y$.

We want to avoid this capture.

Example. Formula $\alpha \stackrel{\text{def}}{=} (S(x) \wedge (\forall y\,(P(x) \to Q(y))))$; term $t = f(y, y)$.

The leftmost x can be substituted by t since it is not in the scope of any quantifier, but substituting in $P(x)$ puts the variable y into the scope of $\forall y$.

We can prevent capture of variables by a different choice of variable. (Above, we might be able to substitute $f(z, z)$ instead of $f(y, y)$. Or alter α to quantify some other variable.)

Substitution—Formal Definition

Let x be a variable and t be a term. For a formula α,

1. If α is $P(t_1, \dots, t_k)$, then $\alpha[t/x]$ is $P\Big(t_1[t/x], \dots, t_k[t/x]\Big)$.
2. If α is $(\neg\beta)$, then $\alpha[t/x]$ is $(\neg\beta[t/x])$.
3. If α is $(\beta \star \gamma)$, then $\alpha[t/x]$ is $(\beta[t/x] \star \gamma[t/x])$.
4. If α is $(\forall x\ \beta)$ or $(\exists x\ \beta)$, then $\alpha[t/x]$ is α.
5. If α is $(\mathcal{Q}y\ \beta)$ for some other variable y and some quantifier $\mathcal{Q} \in \{\exists, \forall\}$, then
   (a) If y does not occur in t, then $\alpha[t/x]$ is $(\mathcal{Q}y\ \beta[t/x])$.
   (b) Otherwise, select a variable z that occurs in neither α nor t; then $\alpha[t/x]$ is $(\mathcal{Q}z\ (\beta[z/y])[t/x])$.

The last case prevents capture by renaming the quantified variable to something harmless.

Example, Revisited

Example. If α is $\bigl(\forall x\,(\exists y\,((x+y)=z))\bigr)$, what is $\alpha[(y-1)/z]$?

This falls under case 5(b): the term to be substituted, namely $y-1$, contains a variable y quantified in formula α.

Let β be $((x+y)=z)$; thus α is $(\forall x\,(\exists y\ \beta))$.

Example, Revisited

Example. If α is $\left(\forall x\left(\exists y\left((x+y)=z\right)\right)\right)$, what is $\alpha[(y-1)/z]$?

This falls under case 5(b): the term to be substituted, namely $y-1$, contains a variable y quantified in formula α.

Let β be $((x+y)=z)$; thus α is $(\forall x\,(\exists y\ \beta))$.

Select a new variable, say w. Then

$$\beta[w/y] \quad \text{is} \quad ((x+w)=z),$$

and

$$\beta[w/y][(y-1)/z] \quad \text{is} \quad ((x+w)=(y-1)) \ .$$

Thus the required formula $\alpha[(y-1)/z]$ is

$$\left(\forall x\left(\exists w\left(((x+w)=(y-1))\right)\right)\right) \ .$$