Home Page

Title Page

Contents

Page 1 of 16

Go Back

Full Screen

Close

Quit

CS206 Lecture 07

Predciate Logic: Introduction and Syntax

G. Sivakumar
Computer Science Department
IIT Bombay
siva@iitb.ac.in
http://www.cse.iitb.ac.in/∼siva

Tue, Jan 14, 2003

Plan for Lecture 07

- Why Predicate Logic?
- Terms
- Atomic Formulae, WFF

Why Predicate Logic

- Propositional Logic lacks expressive power.
- Good for structure of arguments.
- Simple and elegant context to introduce syntax, semantics, proof methods, soundness, completeness, ...
- But, not suitable for richer domains we wish to work with. Examples:
  - Numbers

    Between any n and $2n$ there is a prime number.
  - Lists

    The length of a list is not changed by reversing the order of elements.

Home Page

Title Page

Contents

Page 3 of 16

Go Back

Full Screen

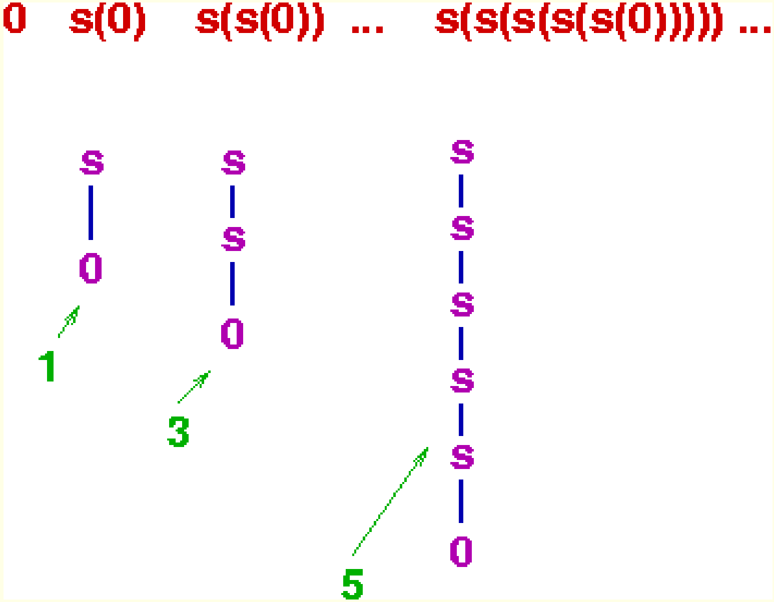
Close

Quit

How to Represent Numbers

There are infinitely many numbers. How many symbols do we need?

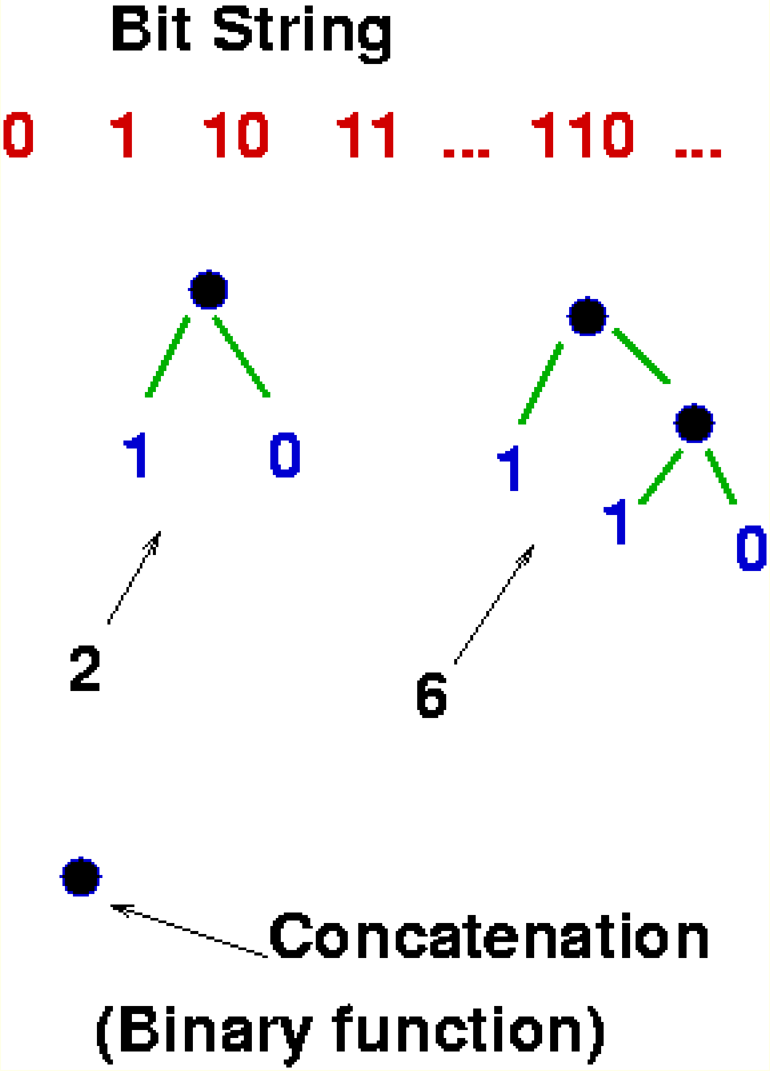
Unary Notation



Two symbols

- Constant $\mathbf{0}$
- Unary Function Symbol s

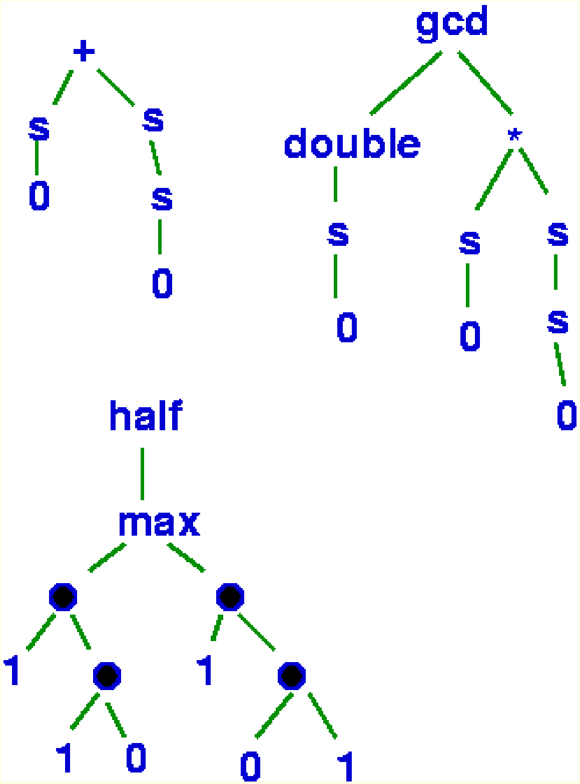
Binary Representation



Terms

Introducing more **function** symbols

- double, half, square, ..
- $+, *, gcd, \ldots$
- ...



Functions: Arities

The arity of a function symbol is the number of arguments (parameters) it takes.

- Constants (arity 0) 0, 1, a, b, c, ...
- Unary (arity 1) s, half, double, ...
- Binary (arity 2) $+$, *, gcd, $\cdot$, ...
- Ternary, Quarternary, ...

In general F is a set of function symbols $f, g, h, \ldots$ with arity $k_1, k_2, k_3, \ldots$.

Functions: Types

Intuitive notion of typing.

Base Types and Derived Types

- Numbers (N)
- Characters
- Strings
- Lists ($List$)
- Records, Arrays, ...

Function Types

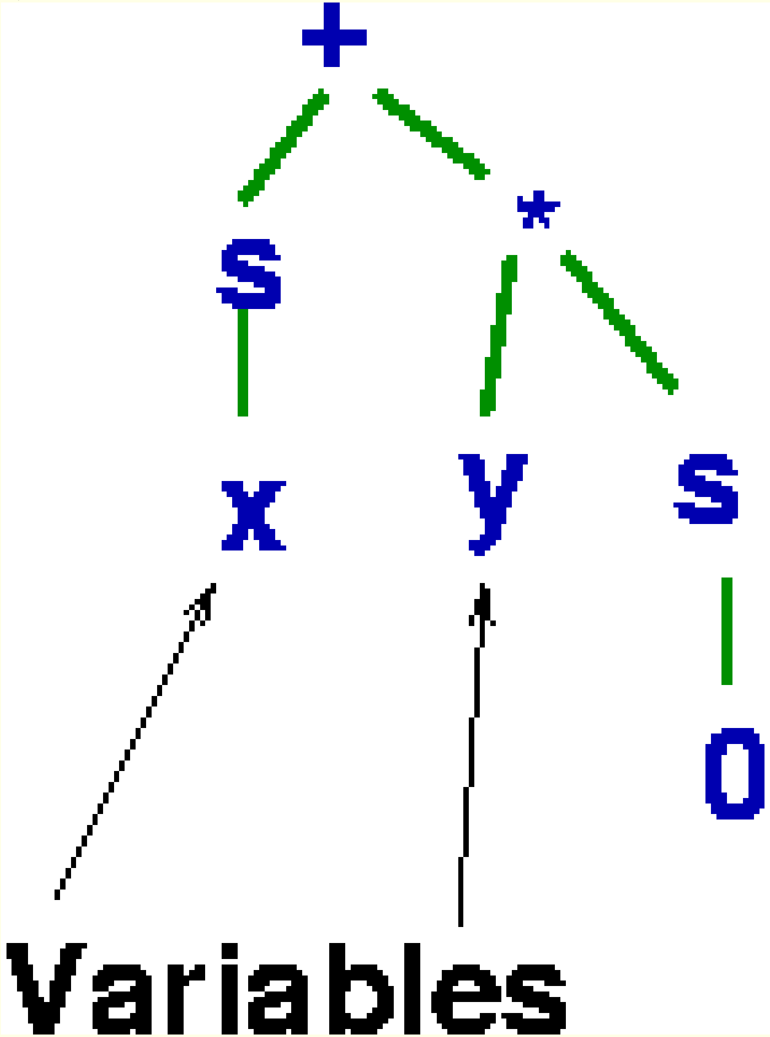
- Addition $+$ has type $N \times N \rightarrow N$
- Append @ has type $List \times List \rightarrow List$
- Length len has type $List \rightarrow N$

In general, Uninterpreted, Untyped Terms $f(g(a, b), h(c))$

Terms with Variables

Variables stand for terms.

Typically, $x, y, z, \ldots$



Typed and Untyped.

Definition of Terms

Intuitive: *A tree with leaves labelled by constants or variables and interior nodes with function symbol of correct arity*

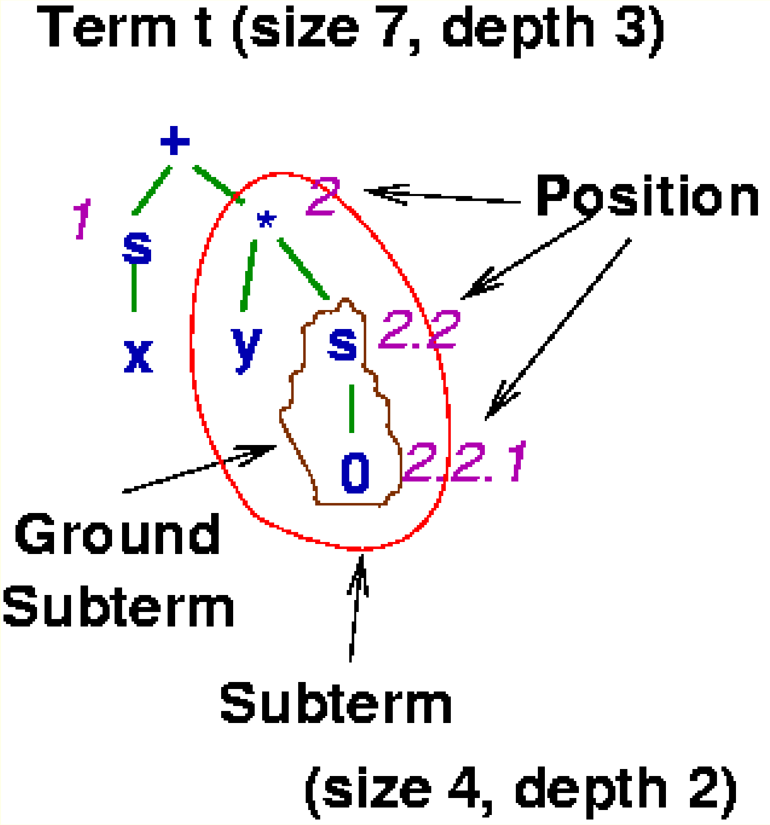
Using Structural Induction

The terms $s, t, u, ...$ of a first-order language are defined recursively as follows.

- A variable is a term.
- A constant symbol is a term.
- If $t_1, ..., t_n$ are terms and f is an function symbol with arity n, then $f(t_1, ..., t_n)$ is a term.
- Nothing else is a term.

More about Terms

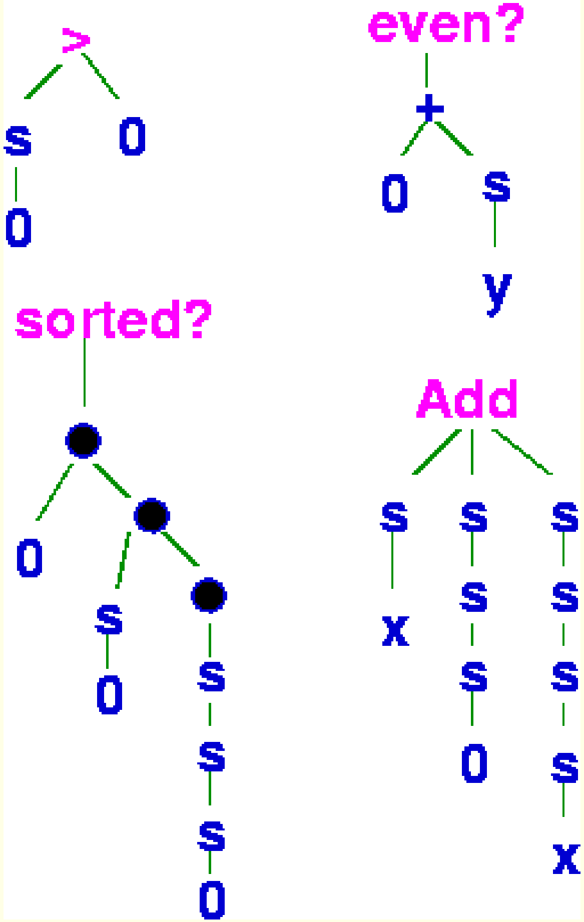
- A ground term is one with no variables.
- Define formally notions of size, depth, position, subterm



- Later we will see substitutions, unifiers, ...

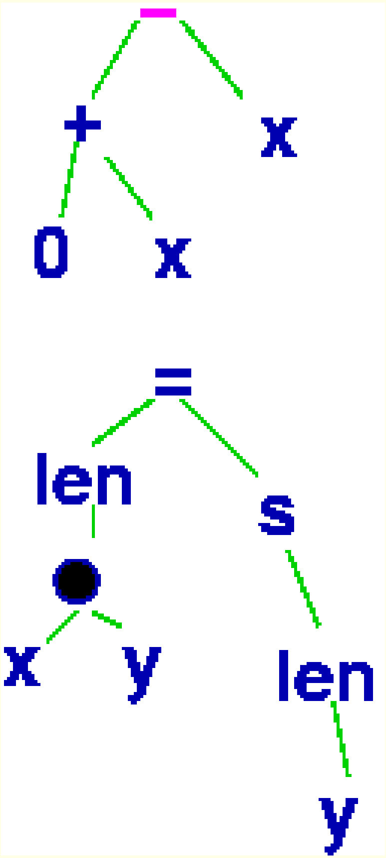
Predicate Symbols

- Predicate symbols (like propositions) express truth values
- Like function symbols they have arities.



- They take only **terms** as arguments (cannot be nested)

Equality: A Special Predicate



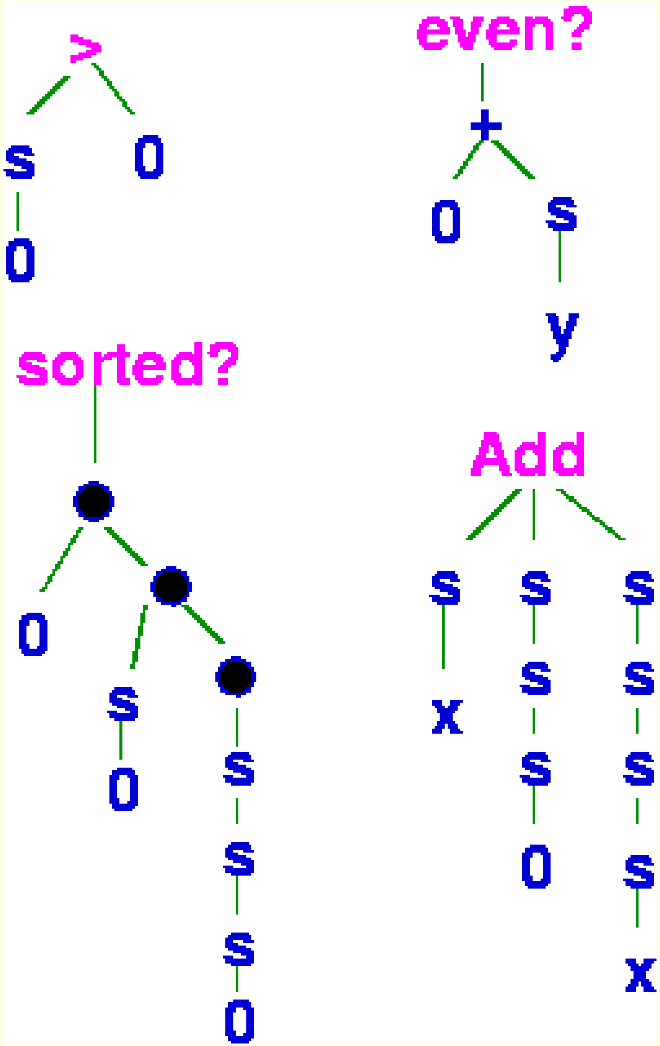
Built-in properties

- Reflexive $x = x$
- Symmetric $x = y \rightarrow y = x$
- Transitivity

More on this later in Equational Logic

Atomic Formulae

If $t_1, ..., t_n$ are terms and P is an n-place predicate symbol then $P(t_1, ..., t_n)$ is an atomic formula.



Quantifiers

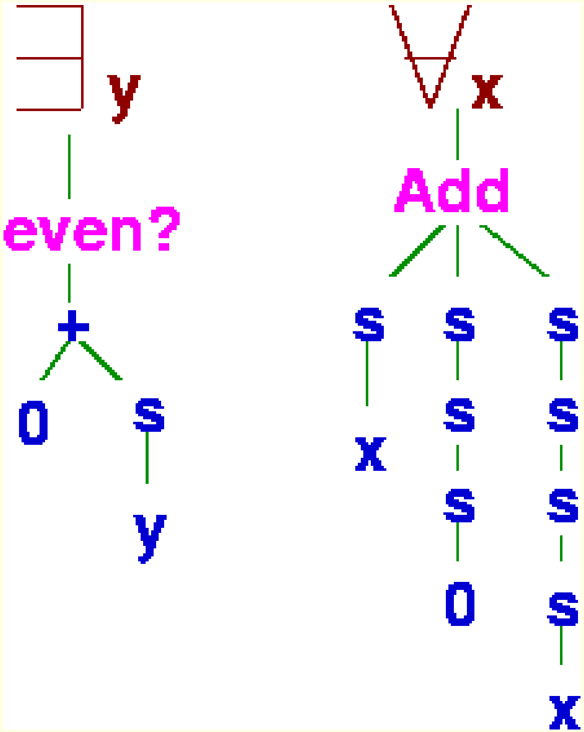
Variables in formulae can be quantified in two ways.

- Universal (for all values)

$$\forall$$

- Existential (for some values)

$$\exists$$



Putting it All Together

The alphabet of predicate logic uses the following sets.

1. $C = \{a, b, \ldots\}$ is the finite or countably infinite set of constant symbols.
2. $F = \{f, g, \ldots\}$ is the finite or countably infinite set of function symbols. Each symbol has a finite arity.
3. $P = \{p, q, \ldots, =, \ldots\}$ is the finite or countably infinite set of predicate symbols. Each symbol has a finite arity.
4. $Vars = \{x, y, \ldots\}$ is the (countably infinite) set of variables.
5. $Conn = \{\neg, \wedge, \ldots\}$ is the set of connectives.
6. $Val = \{\mathbf{t}, \mathbf{f}\}$ is the set of truth constants.
7. $Q = \{\forall, \exists\}$ is the set of quantifiers.

Well Formed Formulae

The set of atomic formulae (or atoms) is defined by structural induction.

1. The truth constants $\mathbf{t}$ and $\mathbf{f}$ are atomic formulae.
2. If p is a predicate symbol with arity n and $t_1, t_2, \ldots, t_n$ are terms, then $p(t_1, t_2, \ldots, t_n)$ is an atomic formula.
3. Nothing else is an atomic formula.

The set of well-formed formulae (or just formulas or wffs) is also defined by structural induction.

1. Every atomic formula is a wff.
2. If ϕ_1, ϕ_2 are wff, then so are
   - $\neg(\phi)$
   - $\phi_1 \wedge \phi_2$
   - $\phi_1 \vee \phi_2$
   - $\ldots$
3. If ϕ is a wff and x is a variable, then $\forall x\phi$ and $\exists x\phi$ are wffs.
4. Nothing else is a wff.