# CS 5090: Software Fault Tolerance – Predicate Logic

Ali Ebnenasir
Department of Computer Science
Michigan Technological University



## Acknowledgement

- The contents of this lecture are adapted from

Anil Nerode and Richard A. Shore, "**Logic for Applications**", Springer-Verlag, 1997.

## Basic Concepts - Predicates

- **Predicate**: relation between objects
  - Special case: unary relations specify property of objects
- **Predicate logic**: Logic of predicates
- Predicates could be n-ary
- What is 0-ary predicate?
  - Proposition
  - Statement of facts independent of variables

## Basic Concepts – Domain of Discourse

- Example:
  - Domain of natural numbers
  - Let φ(x, y) be "x is less than y". Then
  - φ(3, y) denotes a unary predicate
  - What does φ(3, 4) denote?
- x and y are variables
- 3 and 4 are constants
- Variables are placeholders for objects in a domain of discourse



## Basic Concepts – Domain of Discourse

- To reason about the relations between objects in a domain of discourse
  - Identify
    - the non-empty domain of objects
    - variables that range over the domain of discourse
    - constants that are the names of objects
- An **n-ary predicate** is an n-tuple representing a relation over n objects



## Basic Concepts – Functions

- A **function** takes one or more objects in the domain and generates another object of the domain
  - Example: addition on natural numbers in mathematical discourse
  - Is subtraction a function?
- Example: a ternary function f(x,y,z) = x.y + z
- What is f(1,y,1)?
- How about f(1,0,1)?

## Basic Concepts – Functions vs. Predicates

- Predicates are relations
- Functions are specific relations
- 0-ary predicates → propositions
- 0-ary functions → constants
- **Term**: all the symbols generated by functions, constants and variables
- Predicate constructors:
  - – Truth-functional connectives
  - – Universal and existential quantifications



## Syntax - Symbols

- Distinct **primitive symbols** of a language
  - – Variables: x, y, z, …, $x_0$, $x_1$, …, $y_0$, $y_1$, …
    - may be infinite
  - – Constants: c, d, $c_0$, $d_0$, …
    - may be empty, finite or infinite
  - – Connectives: $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$
  - – Quantifiers: $\forall$, $\exists$
  - – Predicate symbols: P, Q, R, $P_0$, $P_1$, …
  - – Function symbols: $f$, $g$, $h$, $f_0$, $f_1$, …
  - – Punctuation: the comma, ), (



## Syntax - Terms

- **Term**: Symbol whose interpretation is an element of the domain of discourse
- Formal inductive definition:
  1. Every variable is a term
  2. Every constant symbols is a term
  3. If $f$ is an n-ary function symbol, for n =1,2, …, and $t_1$, …, $t_n$ are terms, then $f(t_1, …, t_n)$ is also a term.
- ***Ground terms***: terms with no variables; also called *variable-free* terms
  - – Constants and terms built up from constants by Rule 3

## Syntax - Formulas

- **Atomic formula**: an expression of the form $R(t_1, \ldots, t_n)$ for a predicate $R$ and terms $t_1, \ldots, t_n$
- Formula:
  - **Atomic** formula
  - $(\alpha\wedge\beta)$, $(\alpha\vee\beta)$, $(\alpha\rightarrow\beta)$, $(\alpha\leftrightarrow\beta)$, $(\neg\alpha)$, where $\alpha$ and $\beta$ are formulas
  - $((\forall v)\ \alpha)$ , $((\exists\ v)\ \alpha)$, where $v$ is a variable and $\alpha$ is a formula



## Syntax – Some Definitions

- **Subformula**: a consecutive sequence of symbols that is a formula itself
- **Bound occurrence of a variable *v***: if there is a subformula that begins with $(\forall v)$ or $(\exists\ v)$
- **Free occurrence of *v***: if that occurrence is not bound
- **Free occurrence of *v* in a formula**: if *v appears free* at least once
- **Sentence**: a formula with no free occurrences
- **Open formula**: a formula without quantifiers



## Syntax – Instantiation (substitution)

- $\varphi$ (v) means v **occurs free** in formula $\varphi$
- $\varphi$ (v/t) is the resulting formula by **substituting** every free occurrence of v in $\varphi$ with t; also denoted $\varphi$(t)
  - $\varphi$ (t) is called an **instance** of $\varphi$
- **Ground instance**: $\varphi$(t) contains no free vars.
- **Substitutable**: if all (free) variables in t remain free in every occurrence of v in $\varphi$ (v/t), then we say t is substitutable for the variable v in $\varphi$ (v)

## Example

- A language for the integers with constants 0 and 1
- Unary function succ(x)
- Predicate P(x,y,x) represents the relation x+y = z
- Is $\varphi = \forall x \, \exists y$ P(x,y,0) a sentence?
- Is $\varphi$ true? Why?
- How about $\forall x \, \exists y$ P(succ(y),y,0)? Is it true? Why?



## More Examples

- Is (($\forall x$) $R$(x,y)) a sentence? Why?
- How about (($\forall x$) (($\exists y$) $R$(x,y)))? Why?
- Is x free in ((($\forall x$) $R$(x,y)) $\vee$ (($\exists y$) $R$(x,y)))? How about y?



## Syntax – Formation Tree

- Term formation trees: finitely branching trees labeled with terms satisfying
  1. Leaves are labeled with variables and constants
  2. Non-leaf nodes are labeled with terms $f(t_1, \ldots, t_n)$
  3. A node labeled with $f(t_1, \ldots, t_n)$ has exactly n immediate successors labeled with $t_1, \ldots, t_n$
- The root of the tree is associated with the term

## Semantics

- Context (domain of discourse) → an interpretation of a language of predicate logic
- Meaning of predicates and functions
- Example:
  - Context → natural number
  - Predicate → "less than" <
  - Function → addition +



## Semantics - Structure

- A **structure** ***S*** for a language ***L*** includes
  - A non-empty domain ***D***
    - Rational numbers
  - An assignment of an actual n-ary relation to each n-ary predicate symbol
    - A binary realtion "less than or equal" (n = 2)
  - An assignment of each element of ***D*** to each constant symbol of ***L***
    - Constants c = 0, d = 1
  - An assignment of an n-ary actual function $D_n \rightarrow D$ to each function symbol *f* of ***L***
    - Multiplication, addition



## Semantics – Ground Terms

1. Each constant term $t_c$ names an element $e_S$ in the domain ***D*** of a structure ***S***

2. If terms $t_1, \dots, t_n$ of the language ***L*** name the elements $e_1, \dots, e_n$ of ***D***, and *f* is an n-ary function symbol of ***L***, then

   the term $f(t_1, \dots, t_n)$ names the element $f^S(e_1, \dots, e_n)$, where $f^S$ is the interpretation of *f* in ***S***

## Semantics - Sentences

- The **truth of a sentence** φ of a language ***L*** is defined w.r.t. a structure ***S,***
  - Assume that every constant in ***S*** is named by a ground term in ***L***
- ***S*** *satisfies* φ, denoted ***S*** **|=** φ

1) Let $R(t_1, \dots, t_n)$ be an atomic sentence
   - ***S*** |= $R(t_1, \dots, t_n)$ iff all the elements $(e_1, \dots, e_n)$ in ***S*** that are named by the terms $t_1, \dots, t_n$ are related by the interpretation of predicate R in ***S***



## Semantics - Sentences

2) ***S*** |= ¬φ **iff** it is not the case that ***S*** |= φ
3) ***S*** |= (α∨β) **iff** ***S*** |= α or ***S*** |= β
4) ***S*** |= (α∧β) **iff** ***S*** |= α and ***S*** |= β
5) ***S*** |= (α→β) **iff** ***S*** |≠ α or ***S*** |= β
6) ***S*** |= (α↔β) **iff** (***S*** |= α and ***S*** |= β) or (***S*** |≠ α and ***S*** |≠ β)
7) ***S*** |= ∃*v* φ(*v*) **iff** for some ground term t, ***S*** |= φ(*t*)
8) ***S*** |= ∀*v* φ(*v*) **iff** for all ground terms t, ***S*** |= φ(*t*)



## Semantics – Some Definitions

- A sentence φ of a language ***L*** is **valid** (denoted **|=** φ) if all structures satisfy φ
  - i.e., φ is true in all structures
- Let Σ be a set of sentences.
  - A sentence α is a **logical consequence** of Σ (denoted Σ |=α) if α is true in every structure that all members of Σ are true
  - Σ is **satisfiable** if there exists a structure ***S*** in which all members of are true; i.e., ***S*** is a **model** of Σ

## Semantics – Some Definitions

- A formula ψ of a language ***L*** with free variables $v_1, \ldots, v_n$ is **valid in a structure** ***S*** (denoted ***S*** |= ψ) if ***S*** satisfies the universal closure of ψ
  - Universal closure of ψ is obtained by putting a $\forall v_i$ in front of ψ for every $v_i$
- A formula ψ of a language ***L*** is **valid** if it is valid in every structure for ***L***
- What does "a structure ***S*** satisfies a formula ψ" mean?



## Semantics – Some Definitions

- Let Σ be a set of formulas of a language ***L***.
- Σ is **satisfiable iff** there exists a structure ***S*** such that all formulas of Σ are valid in ***S***
- ***S*** is called a **model** of Σ.



## Semantics – Some Notes

- For a given structure ***S*** and a sentence φ, either φ or ¬φ is true in ***S***
- For a given structure ***S*** and a formula ψ, it may be the case that neither ψ nor ¬ψ is valid in ***S***
  - Why?

# Examples

- Structure $S_1$
  - *Domain*: natural numbers
  - *Predicate* R: "less than" (denoted <) relation
- Is the sentence $(\forall x)\ (\exists y)\ R(x, y)$ true in $S_1$?
- Structure $S_2$
  - *Domain*: rational numbers
  - *Predicate* R: "less than" (denoted <) relation
- Is $(\forall x)\ (\forall y)\ \ (R(x, y) \rightarrow ((\exists z)\ (R(x, z) \wedge R(z, y))\ )\ )$ true in $S_2$?
- Is $(\forall x)\ (\forall y)\ \ (R(x, y) \rightarrow ((\exists z)\ (R(x, z) \wedge R(z, y))\ )\ )$ true in $S_1$?