Lecture 2: Predicate Logic (23 - 26 May)

Lecturer: Ternovska, Eugenia *Scribe: Lily Li*

2.1 Predicate Calculus (First Order Logic)

A **first-order** vocabulary $\mathcal{L}$ is specified by:

1. For each $n \in \mathbb{N}$ a set of n-ary function symbols (possibly empty). $f, g, h, ..., +, \cdot, s$ are used as meta symbols for function symbols. A zero-ary function symbol is called a **constant symbol**.
2. For each $n \geq 0$, a set of n-ary predicate symbols (must be non-empty for some n). $P, Q, R, ..., <, \leq, =$ are used as meta symbols for predicate symbos. A zero-ary predicate symbol is the same as a propositional atom.

2.1.1 First Order Language

Logical Symbols for formula building include:

1. Parenthesis (,)
2. Connectives $\supset, \neg$
3. Variables $v_1, v_2, ...$
4. Equality Symbol $=$

Parameters include:

1. For all $\forall$ (depends on domain)
2. Predicate symbols: $P_1, P_2, ...$ (have arity)
3. Constant symbols: $c_1, C_2, ...$
4. Function symbols.

Each language mush contain at least one non-logical predicate symbol or equality. Examples of parameters is:

$$L_A = \{0, s, \cdot, +, =\}$$

where 0 is a constant $s, \cdot, +$ are function symbols (first is unary, second and third are binary) and the equality signifies the language of arithmetics. The parameter $\forall$ is usually omitted in the definition and is implicit.

For a language L, The L**-term** is defined as

1. Every variable is a term
2. If f is an n-ary ($n \geq 0$) function symbol, and $t_1, ..., t_n$ are terms then $f(t_1, ..., t_n)$ is a term.

We will just say **term** if the $\mathcal{L}$ is clear from context. Use $e, 0, 1$ as metasymbol for constants. All constants in $\mathcal{L}$ are $\mathcal{L}$-terms.

Theorem 2.1 ***Unique Readability Theorem for Terms:*** *if terms $ft_1 \cdots t_k$ and $fu_1 \cdots u_l$ are syntactically equal, then $k = l$ and $t_i =_{syn} u_i, 1 \leq i \leq k$. This also hold for first-order formulas.*

Proof: Much like the Unique Readability Theorem in propositional logic, we will assign a weight to each symbol (f gets $n-1$ and each variable gets -1) and claim that the total weight is -1. This is quite easy to see by a simple induction proof. Also, for $n \geq 1$, the initial part of the term has positive weight. Suppose i is the first index where $t_i \neq_{syn} u_i$. Observe that either t_i is the initial part of u_i or u_i is the initial part of t_i (since $t_j = u_j$ for all $1 \leq j \leq i-1$). Neither is possible so we must have $t_i =_{syn} u_i$. ■

On Notation: use $r, s, t, ...$ to denote terms.

A L**-formula**, also vocabulary $\mathcal{L}$ or just formula, is defined as

1. $P(t_1, ..., t_n)$ is an **atomic formula** where P is an n-ary predicate.
2. If A, B are L-formulas then are are $\neg A$, $A \wedge B$, and $A \vee B$.
3. If A is an L-formula, x is a variable then $\forall x, A$ and $\exists x A$ are formulas.

The set $free(\phi)$ of **free variables** of formula ϕ satisfies: if ϕ is atomic then $free(\phi) :=$ the set of variable occurring in ϕ. An occurrence of x in A is **bound** iff it is in a subformula of A of the form $\forall x B$ or $\exists x B$. *Note: a variable can have both free and bound occurrences in one formula.* E.g. $Px \wedge \forall x Qx$.

$$\begin{aligned} free(\neg\phi) &:= free(\phi) \\ free(\phi \wedge \psi) &:= free(\phi) \cup free(\psi) \\ &free(\forall) \end{aligned}$$

If a formula A or a term t which does not have free variables (i.e. everything is quantified) is **closed**. A closed formula is a **sentence**.

2.1.2 Semantics of FO Logic

Suppose a FO language L is fixed **structure** give meaning to parameters. An L-structure $\mathcal{M}$ consists of

1. A non-empty set M called the domain (universe) of $\mathcal{M}$
2. Variables of $\mathcal{L}$ range over M.
3. For each n-ary function symbol $f \in \mathcal{L}$, there is an associated function $f^{\mathcal{M}} : M^n \mapsto M$. If $n = 0$, then f is a constant symbol and $f^{\mathcal{M}}$ is simply an element of M.
4. For each n-ary predicate symbol in $\mathcal{L}$, there is an associated relation $P^{\mathcal{M}} \subset M^n$. If $\mathcal{L}$ contains $=$, then $=^{\mathcal{M}}$ must be the true equality relation on M.

Generally, predicate symbols can be interpreted as arbitrary relations of the appropriate arity, but not =. = must be the equality relation.

Every $\mathcal{L}$-sentence becomes either true or false when interpreted by an $\mathcal{L}$-structure $\mathcal{M}$ as follows: if a sentence A becomes true under $\mathcal{M}$, then $\mathcal{M}$ **satisfies** A or $\mathcal{M}$ is a model for A (written $M \vDash A$).

A $\mathcal{M}$ is **finite** if the universe M of $\mathcal{M}$ is finite. If a formula A has free variables, then these variables must be interpreted as specific elements in the universe M before AA gets a truth value under $\mathcal{M}$. An **object assignment (o.a.)** σ for struct. $\mathcal{M}$ is a mapping from variables to the universe M. Denote $\mathcal{M} \vdash A[\sigma]$ to mean struct. $\mathcal{M}$ satisfies formula A when the free variables of A are interpreted using o.a. σ. First we need to define the notion of

E.g. define the normal understanding of natural numbers as our structure $\mathcal{M}$. The short hand for this is: $(\mathbb{N}; \leq, s, 0)$ where s is the successor. Let σ be $P^{\mathcal{M}} = \{\langle u_1, u_2\rangle : u_1 \leq u_2, u_1, u_2 \in M\}$, $c^{\mathcal{M}} = 0$, and $v_i \mapsto i-1$. Remark that the structure is the semantics (meaning) while languages provide the syntax (symbols).

Here is an idea: if you have to deal with binary relations it may useful to think about graphs and graphs structures. For example consider

$$\exists x \forall y : Pxy \vDash \forall y \exists x : Pxy$$

how would one show that this is a valid logical implication? Let x be a vertex, y be an edge, and P be x is incident to edge y. The true of the logical implication is clear.

Next lets axiomatize first order logic. Consider Pagan's theorem: . It axiomatizes NP using second-order existential quantifiers (field of discrete complexity?)

Definition 2.2 *A set of sentences is* ***consistence*** *if there exists a structure which satisfies every sentence in the set.*

Consider the set of sentences: $\forall x, \neg S(x,x)$, $\exists x, P(x)$, $\forall x, \exists y S(x,y)$ (again use the graph interpretation...). You should see that this set Φ is consistent. To prove it, you need to present a model. Some are more detailed than others. The most general model is the best.

Theorem 2.3 *(Compactness Theorem)*

1. *If* $\Gamma \vDash \phi$ *then* $\Gamma_0 \vDash \phi$ *for some finite* $\Gamma_0 \subset \Gamma$.
2. *If every finite subset* Γ_0 *of* Γ *is satisfiable, then* Γ *is satisfiable.*

Observe that this is quite similar to the compactness statement in propositional logic. However compactness does not hold for second order logic. It also fails when restricted to finite models.

Notation: let Σ be a set of sentences. $mod\Sigma$ are all the models of Σ. A class k of structures for language $\mathcal{L}$ is an **elementary class (EC)** iff $K = mod\phi$ for some F.O. sentence ϕ. K is **elementary class in the wider sense (ECΔ)** if $K = mod\Sigma$ for a set of sentences Σ.

Consider the set of graphs with self loops. You can to axiomatize this structure and you will find that this in EC. However, if you only wanted finite such graphs, you are out of luck! You cannot axiomatize these graph using first order logic. This is not in EC and not even in ECΔ.

Theorem 2.4 *(Lowenheim-Skolem) If a set of sentences* Σ *has an arbitrary large finite model then it has an infinite model.*

Proof: Here is what we are going to do: for each natural number $k \geq 2$ we are going to construct a sentence λ_k such λ_k has at least k elements. Let $\lambda_2 := \exists x_1, \exists x_2 \neg(x_1 = x_2)$,

$$\lambda_3 = \exists x_1 \exists x_2 \exists x_3 : \neg(x_1 = x_1) \wedge lnot(x_1 = x_3) \wedge \neg(x_2 = x_3)$$

and so on for all $k \in \mathbb{N}$. Consider $\Phi = \Sigma \cup \{\lambda_2, \lambda_3, ...\}$. Every finite subset of Φ has a model so by compactness theorem Φ, and by extension Σ has a model. Since the model of Φ is larger than any natural number it must be infinite, so Σ has an infinite model. ■

Corollary 2.5 *From the above theorem we can conclude that:*

1. *The class of all finite structures (for a fixed language) is not ECΔ.*
2. *The class of all infinite structures is not EC.*

Proof:

■

2.2 Substitution

Syntactic Definition: where t, u are terms

$t(u/x)$ is t after replacing x with u
$A(u/x)$ is A after replacing all *free* occurances of x with u

Semantic Definition:

Lemma 2.6 *For each struct. $\mathcal{M}$ and object assignment σ, for $m = u^{\mathcal{M}}[\sigma]$*

$$(t(u/x))^{\mathcal{M}}[\sigma] = t^{\mathcal{M}}[\sigma(m/x)]$$

Proof: By structural induction on t. ■

Definition 2.7 *Term t is **freely substitutable** for $x \in A$ iff no free occurrence of $x \in A$ is in a subformula of A that looks like $\forall y B$ or $\exists y B$ where y occurs in t.*

Theorem 2.8 *If t is freely substitutable for $x \in A$ then for all structures $\mathcal{M}$ and all object assignments σ, $\mathcal{M} \vDash A(t/x)[\sigma] \iff \mathcal{M} \vDash A[\sigma(m/x)]$, where $m = t^{\mathcal{M}}[\sigma]$.*

Proof: Wouldn't you believe it, it is *more* structural induction on A. ■