

Lecture 10

First Order Predicate Logic. Clausal form

Isabela Drămnesc UVT

Computer Science Department,
West University of Timișoara,
Romania

- ▶ We review here (first order) predicate logic:
 - ▶ the syntax,
 - ▶ the semantics,
 - ▶ illustrate some difficulties of the semantic evaluation of truth in first order logic,
 - ▶ review some results that deal with this difficulty.

Syntax of first order predicate logic

- ▶ The **vocabulary** of the language contains the symbols from which expressions of the language are built:
 - ▶ Reserved symbols:
 - ▶ $()$,
 - ▶ $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$,
 - ▶ \forall, \exists .
 - ▶ The set of variables \mathcal{V} (countable set).
 - ▶ The set of language symbols \mathcal{L} :
 - ▶ \mathcal{F} - function symbols (each with their own arity),
 - ▶ \mathcal{P} - predicate symbols (with arity),
 - ▶ \mathcal{C} - constant symbols.
- ▶ Example language (symbols):
 $\mathcal{L} = \{\{+_{/2}, -_{/1}\}, \{<_{/2}, \geq_{/2}\}, \{0, 1\}\}$. We use a notation similar to Prolog to indicate the arity of symbols.

- ▶ The expressions of (first order) predicate logic:
 - ▶ **Terms:**
 - ▶ variables $v \in \mathcal{V}$ are terms,
 - ▶ constants $c \in \mathcal{C}$ are terms,
 - ▶ if $f/n \in \mathcal{F}$ and t_1, \dots, t_n are terms, then so is $f(t_1, \dots, t_n)$.
 - ▶ **Formulae:**
 - ▶ if $p/n \in \mathcal{P}$ and t_1, \dots, t_n are terms, then $p(t_1, \dots, t_n)$ is an atomic formula,
 - ▶ if F, G are formulae, then $\neg F, F \wedge G, F \vee G, F \Rightarrow G, F \Leftrightarrow G$ are (compound) formulae,
 - ▶ if $x \in \mathcal{V}$ and F is a formula then $\forall xF, \exists xF$ are (quantified) formulae (the universally and existentially quantified formulae, respectively).

Semantics of first order predicate logic

- ▶ The semantics of first order logic describe the meaning of expressions in the language.
- ▶ Such a language is used to describe:
 - ▶ a **domain** of objects,
 - ▶ **relations** between the objects (or properties of the objects),
 - ▶ **processes** or **functions** that produce new objects from other objects.
- ▶ To find (compute) the meaning of an expression, one must first define an **interpretation** of the symbols:
 - ▶ constants are interpreted as objects in the domain described by the language,
 - ▶ function symbols are interpreted as processes (functions) in the domain described by the language,
 - ▶ predicate symbols are interpreted as relations/properties between/of objects in the domain described by the language.

- ▶ Consider the language presented previously $\mathcal{L} = \{\{+/_2, -/_1\}, \{</_2, \geq/_2\}, \{0, 1\}\}$ and let's consider two interpretations of this language:
 - ▶ \mathcal{I}_1 an interpretation in the natural numbers:
 - ▶ $\mathcal{I}_1(0) = \text{seven}$,
 - ▶ $\mathcal{I}_1(1) = \text{zero}$,
 - ▶ $\mathcal{I}_1(+)$ = multiplication,
 - ▶ $\mathcal{I}_1(-)$ = factorial,
 - ▶ $\mathcal{I}_1(<)$ = smaller than,
 - ▶ $\mathcal{I}_1(\geq)$ = divides.
 - ▶ \mathcal{I}_2 an interpretation in domain of strings:
 - ▶ $\mathcal{I}_2(0) = \text{" "}$,
 - ▶ $\mathcal{I}_2(1) = \text{"one"}$,
 - ▶ $\mathcal{I}_2(+)$ = concatenation,
 - ▶ $\mathcal{I}_2(-)$ = reverse,
 - ▶ $\mathcal{I}_2(<)$ = substring,
 - ▶ $\mathcal{I}_1(\geq)$ = sorted version.
- ▶ Note that interpretation shows the correspondence between the name of a concept (constant, function symbol, predicate symbol) and the concept described by that name.

- ▶ Once an interpretation has been defined, one can compute **the value of an expression E under interpretation \mathcal{I} , $v_{\mathcal{I}}(E)$** (i.e. the meaning of an expression under interpretation) in the following way:

- ▶ Once an interpretation has been defined, one can compute **the value of an expression E under interpretation \mathcal{I}** , $v_{\mathcal{I}}(E)$ (i.e. the meaning of an expression under interpretation) in the following way:
 - ▶ **The value of terms under interpretation:**

- ▶ Once an interpretation has been defined, one can compute **the value of an expression E under interpretation \mathcal{I}** , $v_{\mathcal{I}}(E)$ (i.e. the meaning of an expression under interpretation) in the following way:
- ▶ **The value of terms under interpretation:**
 - ▶ In general, terms will evaluate to objects in the universe of discourse.

- ▶ Once an interpretation has been defined, one can compute **the value of an expression E under interpretation \mathcal{I}** , $v_{\mathcal{I}}(E)$ (i.e. the meaning of an expression under interpretation) in the following way:
- ▶ **The value of terms under interpretation:**
 - ▶ In general, terms will evaluate to objects in the universe of discourse.
 - ▶ If $c \in \mathcal{C}$, $v_{\mathcal{I}}(c) = \mathcal{I}(c)$.

- ▶ Once an interpretation has been defined, one can compute **the value of an expression E under interpretation \mathcal{I}** , $v_{\mathcal{I}}(E)$ (i.e. the meaning of an expression under interpretation) in the following way:
- ▶ **The value of terms under interpretation:**
 - ▶ In general, terms will evaluate to objects in the universe of discourse.
 - ▶ If $c \in \mathcal{C}$, $v_{\mathcal{I}}(c) = \mathcal{I}(c)$.
 - ▶ If $x \in \mathcal{V}$, $v_{\mathcal{I}}(v)$ is not defined, unless the variable v is assigned a value. I.e. the value of expressions containing free variables cannot be determined unless the variables have values assigned to them.

- ▶ Once an interpretation has been defined, one can compute **the value of an expression E under interpretation \mathcal{I}** , $v_{\mathcal{I}}(E)$ (i.e. the meaning of an expression under interpretation) in the following way:
- ▶ **The value of terms under interpretation:**
 - ▶ In general, terms will evaluate to objects in the universe of discourse.
 - ▶ If $c \in \mathcal{C}$, $v_{\mathcal{I}}(c) = \mathcal{I}(c)$.
 - ▶ If $x \in \mathcal{V}$, $v_{\mathcal{I}}(v)$ is not defined, unless the variable v is assigned a value. I.e. the value of expressions containing free variables cannot be determined unless the variables have values assigned to them.
 - ▶ If $f(t_1, \dots, t_n)$ is a term, then

$$v_{\mathcal{I}}(f(t_1, \dots, t_n)) = \mathcal{I}(f)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ **The value of formulae under interpretation:**

- ▶ **The value of formulae under interpretation:**
 - ▶ Formulae will evaluate to **true** or **false** (but not both).

- ▶ **The value of formulae under interpretation:**
 - ▶ Formulae will evaluate to **true** or **false** (but not both).
 - ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

- ▶ For compound formulae:

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

- ▶ For compound formulae:

▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

- ▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.
- ▶ $v_{\mathcal{I}}(F \vee G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ or $v_{\mathcal{I}}(G) = \mathbf{true}$ (at least one is true).

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.
- ▶ $v_{\mathcal{I}}(F \vee G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ or $v_{\mathcal{I}}(G) = \mathbf{true}$ (at least one is true).
- ▶ $v_{\mathcal{I}}(F \Rightarrow G) = \mathbf{false}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{false}$.

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.
- ▶ $v_{\mathcal{I}}(F \vee G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ or $v_{\mathcal{I}}(G) = \mathbf{true}$ (at least one is true).
- ▶ $v_{\mathcal{I}}(F \Rightarrow G) = \mathbf{false}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \Leftrightarrow G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = v_{\mathcal{I}}(G)$.

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
 - ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.
 - ▶ $v_{\mathcal{I}}(F \vee G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ or $v_{\mathcal{I}}(G) = \mathbf{true}$ (at least one is true).
 - ▶ $v_{\mathcal{I}}(F \Rightarrow G) = \mathbf{false}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{false}$.
 - ▶ $v_{\mathcal{I}}(F \Leftrightarrow G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = v_{\mathcal{I}}(G)$.
- ▶ For quantified formulae:

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.
- ▶ $v_{\mathcal{I}}(F \vee G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ or $v_{\mathcal{I}}(G) = \mathbf{true}$ (at least one is true).
- ▶ $v_{\mathcal{I}}(F \Rightarrow G) = \mathbf{false}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \Leftrightarrow G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = v_{\mathcal{I}}(G)$.

▶ For quantified formulae:

- ▶ $v_{\mathcal{I}}(\forall x F) = \mathbf{true}$ iff for all values of x from the domain, $v_{\mathcal{I}}(F) = \mathbf{true}$.

▶ **The value of formulae under interpretation:**

- ▶ Formulae will evaluate to **true** or **false** (but not both).
- ▶ For atomic formulae,

$$v_{\mathcal{I}}(p(t_1, \dots, t_n)) = \mathcal{I}(p)(v_{\mathcal{I}}(t_1), \dots, v_{\mathcal{I}}(t_n)).$$

▶ For compound formulae:

- ▶ $v_{\mathcal{I}}(\neg F) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \wedge G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{true}$.
- ▶ $v_{\mathcal{I}}(F \vee G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ or $v_{\mathcal{I}}(G) = \mathbf{true}$ (at least one is true).
- ▶ $v_{\mathcal{I}}(F \Rightarrow G) = \mathbf{false}$ iff $v_{\mathcal{I}}(F) = \mathbf{true}$ and $v_{\mathcal{I}}(G) = \mathbf{false}$.
- ▶ $v_{\mathcal{I}}(F \Leftrightarrow G) = \mathbf{true}$ iff $v_{\mathcal{I}}(F) = v_{\mathcal{I}}(G)$.

▶ For quantified formulae:

- ▶ $v_{\mathcal{I}}(\forall x F) = \mathbf{true}$ iff for all values of x from the domain, $v_{\mathcal{I}}(F) = \mathbf{true}$.
- ▶ $v_{\mathcal{I}}(\exists x F) = \mathbf{true}$ iff for some values of x from the domain, $v_{\mathcal{I}}(F) = \mathbf{true}$.

- ▶ For example, consider \mathcal{I}_1 as defined above:

$$\begin{aligned}v_{\mathcal{I}_1}(-(0 + 1))) &= \\ \mathcal{I}_1(-)(v_{\mathcal{I}_1}(0 + 1)) &= \\ \text{factorial}(\mathcal{I}_1(+)(v_{\mathcal{I}_1}(0), v_{\mathcal{I}_1}(1))) &= \\ \text{factorial}(\text{multiplication}(\text{seven}, \text{zero})) &= \\ \text{factorial}(\text{zero}) &= \\ \text{one.} &\end{aligned}$$

Validity, satisfiability, unsatisfiability

- ▶ We are interested in the meaning of formulae, in particular:

Validity, satisfiability, unsatisfiability

- ▶ We are interested in the meaning of formulae, in particular:
 - ▶ Whether a formula is **valid**, i.e. true under all possible interpretations.

Validity, satisfiability, unsatisfiability

- ▶ We are interested in the meaning of formulae, in particular:
 - ▶ Whether a formula is **valid**, i.e. true under all possible interpretations.
 - ▶ Whether a formula is **satisfiable**, i.e. there is an interpretation such that the formula is true.

Validity, satisfiability, unsatisfiability

- ▶ We are interested in the meaning of formulae, in particular:
 - ▶ Whether a formula is **valid**, i.e. true under all possible interpretations.
 - ▶ Whether a formula is **satisfiable**, i.e. there is an interpretation such that the formula is true.
 - ▶ Whether a formula is **unsatisfiable**, i.e. the formula is false under all possible interpretations.

Validity, satisfiability, unsatisfiability

- ▶ We are interested in the meaning of formulae, in particular:
 - ▶ Whether a formula is **valid**, i.e. true under all possible interpretations.
 - ▶ Whether a formula is **satisfiable**, i.e. there is an interpretation such that the formula is true.
 - ▶ Whether a formula is **unsatisfiable**, i.e. the formula is false under all possible interpretations.
 - ▶ Whether two formulae are **logically equivalent**, i.e. the formulae have the same meaning under all possible interpretations (we denote $F_1 \equiv F_2$).

Validity, satisfiability, unsatisfiability

- ▶ We are interested in the meaning of formulae, in particular:
 - ▶ Whether a formula is **valid**, i.e. true under all possible interpretations.
 - ▶ Whether a formula is **satisfiable**, i.e. there is an interpretation such that the formula is true.
 - ▶ Whether a formula is **unsatisfiable**, i.e. the formula is false under all possible interpretations.
 - ▶ Whether two formulae are **logically equivalent**, i.e. the formulae have the same meaning under all possible interpretations (we denote $F_1 \equiv F_2$).
 - ▶ Whether a formula is a **logical consequence** of a set of other formulae, i.e. the formula is true in all interpretations such that all formulae in the set are true (we denote $F_1, \dots, F_n \models G$).

- ▶ Using these notions in practice is very difficult: the number of possible interpretations for a language is infinite. Checking the value of an expression in all possible interpretations is therefore not practical.

- ▶ Using these notions in practice is very difficult: the number of possible interpretations for a language is infinite. Checking the value of an expression in all possible interpretations is therefore not practical.
- ▶ If a formula is (or a set of formulae are) true under an interpretation in a domain, then that domain is called a **model** of the formula(e).

Herbrand Universe

- ▶ Fortunately, the difficulty represented by the immense number of possible interpretations of a language can be overcome.

Herbrand Universe

- ▶ Fortunately, the difficulty represented by the immense number of possible interpretations of a language can be overcome.
- ▶ We will define a domain and an interpretation that “captures” all the properties of all potential domains and interpretation.

Herbrand Universe

- ▶ Fortunately, the difficulty represented by the immense number of possible interpretations of a language can be overcome.
- ▶ We will define a domain and an interpretation that “captures” all the properties of all potential domains and interpretation.
- ▶ Checking satisfiability (and validity) of a formula (set) can be done by just checking the evaluation under a certain interpretation into this special universe.

Herbrand Universe

- ▶ Fortunately, the difficulty represented by the immense number of possible interpretations of a language can be overcome.
- ▶ We will define a domain and an interpretation that “captures” all the properties of all potential domains and interpretation.
- ▶ Checking satisfiability (and validity) of a formula (set) can be done by just checking the evaluation under a certain interpretation into this special universe.
- ▶ Let \mathcal{L} be a language containing the constant symbols \mathcal{C} , function symbols \mathcal{F} and predicate symbols \mathcal{P} . Let F be a formula over \mathcal{L} .

- ▶ The **Herbrand universe** \mathcal{H} corresponding to the language \mathcal{L} (or corresponding to the formula F) is defined in the following way:

- ▶ The **Herbrand universe** \mathcal{H} corresponding to the language \mathcal{L} (or corresponding to the formula F) is defined in the following way:
 - ▶ If $c \in \mathcal{C}$ then $c \in \mathcal{H}$.

- ▶ The **Herbrand universe** \mathcal{H} corresponding to the language \mathcal{L} (or corresponding to the formula F) is defined in the following way:
 - ▶ If $c \in \mathcal{C}$ then $c \in \mathcal{H}$.
 - ▶ If t_1, \dots, t_n and $f \in \mathcal{F}$ then $f(t_1, \dots, t_n) \in \mathcal{H}$.

- ▶ The **Herbrand universe** \mathcal{H} corresponding to the language \mathcal{L} (or corresponding to the formula F) is defined in the following way:
 - ▶ If $c \in \mathcal{C}$ then $c \in \mathcal{H}$.
 - ▶ If t_1, \dots, t_n and $f \in \mathcal{F}$ then $f(t_1, \dots, t_n) \in \mathcal{H}$.
 - ▶ Note that if $\mathcal{C} = \emptyset$ then add an arbitrary constant to the Herbrand universe \mathcal{H} .

- ▶ The **Herbrand universe** \mathcal{H} corresponding to the language \mathcal{L} (or corresponding to the formula F) is defined in the following way:
 - ▶ If $c \in \mathcal{C}$ then $c \in \mathcal{H}$.
 - ▶ If t_1, \dots, t_n and $f \in \mathcal{F}$ then $f(t_1, \dots, t_n) \in \mathcal{H}$.
 - ▶ Note that if $\mathcal{C} = \emptyset$ then add an arbitrary constant to the Herbrand universe \mathcal{H} .
- ▶ The Herbrand universe is the set of ground terms that can be formed from the constants and function symbols of the language.

- ▶ The **Herbrand universe** \mathcal{H} corresponding to the language \mathcal{L} (or corresponding to the formula F) is defined in the following way:
 - ▶ If $c \in \mathcal{C}$ then $c \in \mathcal{H}$.
 - ▶ If t_1, \dots, t_n and $f \in \mathcal{F}$ then $f(t_1, \dots, t_n) \in \mathcal{H}$.
 - ▶ Note that if $\mathcal{C} = \emptyset$ then add an arbitrary constant to the Herbrand universe \mathcal{H} .
- ▶ The Herbrand universe is the set of ground terms that can be formed from the constants and function symbols of the language.
- ▶ The **Herbrand base** \mathcal{B} of the language \mathcal{L} or the formula F is the set of ground atoms that can be formed from the predicate symbols in \mathcal{P} and terms in \mathcal{H} .

- ▶ A **Herbrand interpretation** $\mathcal{I}_{\mathcal{H}}$ for the language \mathcal{L} is an interpretation whose domain is the Herbrand universe \mathcal{H} whose symbols are interpreted to themselves:

- ▶ A **Herbrand interpretation** $\mathcal{I}_{\mathcal{H}}$ for the language \mathcal{L} is an interpretation whose domain is the Herbrand universe \mathcal{H} whose symbols are interpreted to themselves:
 - ▶ If $c \in \mathcal{C}$, $\mathcal{I}_{\mathcal{H}}(c) = c$.

- ▶ A **Herbrand interpretation** $\mathcal{I}_{\mathcal{H}}$ for the language \mathcal{L} is an interpretation whose domain is the Herbrand universe \mathcal{H} whose symbols are interpreted to themselves:
 - ▶ If $c \in \mathcal{C}$, $\mathcal{I}_{\mathcal{H}}(c) = c$.
 - ▶ If $f \in \mathcal{F}$, $\mathcal{I}_{\mathcal{H}}(f) = f$.

- ▶ A **Herbrand interpretation** $\mathcal{I}_{\mathcal{H}}$ for the language \mathcal{L} is an interpretation whose domain is the Herbrand universe \mathcal{H} whose symbols are interpreted to themselves:
 - ▶ If $c \in \mathcal{C}$, $\mathcal{I}_{\mathcal{H}}(c) = c$.
 - ▶ If $f \in \mathcal{F}$, $\mathcal{I}_{\mathcal{H}}(f) = f$.
 - ▶ If $f \in \mathcal{P}$, $\mathcal{I}_{\mathcal{H}}(p) = p$.

- ▶ A **Herbrand interpretation** $\mathcal{I}_{\mathcal{H}}$ for the language \mathcal{L} is an interpretation whose domain is the Herbrand universe \mathcal{H} whose symbols are interpreted to themselves:
 - ▶ If $c \in \mathcal{C}$, $\mathcal{I}_{\mathcal{H}}(c) = c$.
 - ▶ If $f \in \mathcal{F}$, $\mathcal{I}_{\mathcal{H}}(f) = f$.
 - ▶ If $p \in \mathcal{P}$, $\mathcal{I}_{\mathcal{H}}(p) = p$.
- ▶ A **Herbrand model** for a formula (set) F is a Herbrand interpretation that satisfies F . A Herbrand model can be identified with a subset of the Herbrand base, namely the subset for which

$$v_{\mathcal{I}_{\mathcal{H}}}(p(t_1, \dots, t_n)) = \mathbf{true}.$$

Herbrand's Theorem

Several remarkable results hold.

Herbrand's Theorem

Several remarkable results hold.

Theorem

Let F be a formula. F has a model iff it has a Herbrand model.

Herbrand's Theorem

Several remarkable results hold.

Theorem

Let F be a formula. F has a model iff it has a Herbrand model.

Theorem (Herbrand's theorem (semantic form))

*Let F be a formula (set). F is unsatisfiable iff a formula built from a **finite** set of ground instances of subformulae of F is unsatisfiable.*

Herbrand's Theorem

Several remarkable results hold.

Theorem

Let F be a formula. F has a model iff it has a Herbrand model.

Theorem (Herbrand's theorem (semantic form))

*Let F be a formula (set). F is unsatisfiable iff a formula built from a **finite** set of ground instances of subformulae of F is unsatisfiable.*

Theorem (Herbrand's theorem (syntactic form))

A formula F is provable iff a formula built from a finite set of ground instances of subformulas of F is provable in propositional logic.

- ▶ Herbrand's theorem (semantic form) tells us that we can reduce the question of unsatisfiability in predicate logic to the question of unsatisfiability in propositional logic.

- ▶ Herbrand's theorem (semantic form) tells us that we can reduce the question of unsatisfiability in predicate logic to the question of unsatisfiability in propositional logic.
- ▶ For propositional logic, the **resolution method** is used to decide the question of satisfiability. See [Crăciun, 2010] for details.

- ▶ Herbrand's theorem (semantic form) tells us that we can reduce the question of unsatisfiability in predicate logic to the question of unsatisfiability in propositional logic.
- ▶ For propositional logic, the **resolution method** is used to decide the question of satisfiability. See [Crăciun, 2010] for details.
- ▶ For using resolution in propositional logic, propositional formulas are written in **Conjunctive Normal Form (CNF)**.

- ▶ Herbrand's theorem (semantic form) tells us that we can reduce the question of unsatisfiability in predicate logic to the question of unsatisfiability in propositional logic.
- ▶ For propositional logic, the **resolution method** is used to decide the question of satisfiability. See [Crăciun, 2010] for details.
- ▶ For using resolution in propositional logic, propositional formulas are written in **Conjunctive Normal Form (CNF)**.
- ▶ To use Herbrand's theorem in and propositional resolution, one would need a similar transformation for predicate logic.

- ▶ A **literal** in predicate logic is an atomic formula or the negation of an atomic formula.
- ▶ A formula of predicate logic is in **conjunctive normal form (CNF)** iff it is a conjunction of disjunctions of literals.
- ▶ A formula of predicate logic is in **prenex conjunctive normal form (PCNF)** iff it is of the form

$$Q_1x_1 \dots Q_nx_nM,$$

where Q_i is a quantifier (either \forall , \exists), for $i = 1 \dots n$, and M is a quantifier-free formula in CNF. $Q_1x_1 \dots Q_nx_n$ is called the **prefix** and M is called the **matrix**.

- ▶ A formula is **closed** iff it has no free variables (i.e. all variables are bound by a quantifier).
- ▶ A closed formula is in **clausal form** iff it is in PCNF and its prefix consists only of universal quantifiers.
- ▶ A **clause** is a disjunction of literals.

- Example: The following formula is in clausal normal form:

$$\forall x \forall y \forall z \left((p(f(x)) \vee \neg q(y, z)) \wedge \right. \\ \left. (\neg p(x) \vee q(y, f(z)) \vee r(x, y)) \wedge \right. \\ \left. (q(x, f(z)) \vee \neg r(f(y), f(z))) \right).$$

- ▶ Example: The following formula is in clausal normal form:

$$\forall x \forall y \forall z \left((p(f(x)) \vee \neg q(y, z)) \wedge (\neg p(x) \vee q(y, f(z)) \vee r(x, y)) \wedge (q(x, f(z)) \vee \neg r(f(y), f(z))) \right).$$

- ▶ Notation: Since the matrix only consists of universal quantifiers, these can be omitted. The clausal form can be represented in the following manner (clauses as **sets of literals**, formulae in clausal form as **sets of clauses**):

$$\left\{ \begin{array}{l} \{p(f(x)), \neg q(y, z)\}, \\ \{\neg p(x), q(y, f(z)), r(x, y)\}, \\ \{q(x, f(z)), \neg r(f(y), f(z))\} \end{array} \right\}.$$

- ▶ Example: The following formula is in clausal normal form:

$$\forall x \forall y \forall z \left((p(f(x)) \vee \neg q(y, z)) \wedge (\neg p(x) \vee q(y, f(z)) \vee r(x, y)) \wedge (q(x, f(z)) \vee \neg r(f(y), f(z))) \right).$$

- ▶ Notation: Since the matrix only consists of universal quantifiers, these can be omitted. The clausal form can be represented in the following manner (clauses as **sets of literals**, formulae in clausal form as **sets of clauses**):

$$\left\{ \begin{array}{l} \{p(f(x)), \neg q(y, z)\}, \\ \{\neg p(x), q(y, f(z)), r(x, y)\}, \\ \{q(x, f(z)), \neg r(f(y), f(z))\} \end{array} \right\}.$$

- ▶ Notation: Let F, G be formulas. We denote $F \approx G$ if F and G are equisatisfiable (i.e. F is satisfiable iff G is satisfiable).

Theorem (Skolem)

Let F be a closed formula. Then there exists a formula F' in clausal form such that $F \approx F'$.

- ▶ Skolem's theorem can be used to decide if a formula is unsatisfiable if a method for deciding unsatisfiability of formulas in clausal exists. This is the subject of the next Chapter.

Skolemization Algorithm

IN: closed formula F .

OUT: formula F' in clausal form such that $F \approx F'$.

Running example:

$$\forall x(p(x) \Rightarrow q(x)) \Rightarrow (\forall x p(x) \Rightarrow \forall x q(x))$$

- 1: Rename the bound variables such that no variable appears in the scope of two different quantifiers.

$$\forall x(p(x) \Rightarrow q(x)) \Rightarrow (\forall y p(y) \Rightarrow \forall z q(z))$$

- 2: Eliminate all the equivalence and implication connectives ($\Leftrightarrow, \Rightarrow$).

$$\neg \forall x(\neg p(x) \vee q(x)) \vee (\neg \forall y p(y) \vee \forall z q(z))$$

- 3: Push the negations inside the parantheses, until negations apply only to atomic formulae. Use the equivalences

$$\begin{aligned}\neg(\neg F) &\equiv F, \\ \neg(F \wedge G) &\equiv (\neg F \vee \neg G), \\ \neg(F \vee G) &\equiv (\neg F \wedge \neg G), \\ \neg(\forall x F[x]) &\equiv \exists x \neg F[x], \\ \neg(\exists x F[x]) &\equiv \forall x \neg F[x].\end{aligned}$$

$$\exists x(p(x) \wedge \neg q(x)) \vee \exists y \neg p(y) \vee \forall z q(z)$$

- 4: Extract the quantifiers from the matrix. Since the variables have been renamed, the following equivalences can be applied: $A \text{ op } Qx B[x] \equiv Qx(A \text{ op } B[X])$ and $Qx B[x] \text{ op } A \equiv Qx(B[X] \text{ op } A)$ where Q is one of \forall, \exists and op is one of \wedge, \vee .

$$\exists x \exists y \forall z ((p(x) \wedge \neg q(x)) \vee \neg p(y) \vee q(z))$$

5: Use the distributive laws $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$, $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ to transform the matrix into CNF.

Note that steps 1-5 preserve logical consequence. It is relatively easy to show that step 6 preserves satisfiability

- 5: Use the distributive laws $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$, $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ to transform the matrix into CNF.

$$\exists x \exists y \forall z ((p(x) \vee \neg p(y) \vee q(z)) \wedge (\neg q(x) \vee \neg p(y) \vee q(z)))$$

Note that steps 1-5 preserve logical consequence. It is relatively easy to show that step 6 preserves satisfiability

- 5: Use the distributive laws $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$, $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ to transform the matrix into CNF.

$$\exists x \exists y \forall z ((p(x) \vee \neg p(y) \vee q(z)) \wedge (\neg q(x) \vee \neg p(y) \vee q(z)))$$

6: Skolemization

Note that steps 1-5 preserve logical consequence. It is relatively easy to show that step 6 preserves satisfiability

- 5: Use the distributive laws $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$, $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ to transform the matrix into CNF.

$$\exists x \exists y \forall z ((p(x) \vee \neg p(y) \vee q(z)) \wedge (\neg q(x) \vee \neg p(y) \vee q(z)))$$

6: Skolemization

- ▶ If the prefix is of the form $\forall y_1 \dots \forall y_n \exists x$, let f be a new n -ary function symbol. Delete $\exists x$ from the prefix, replace all occurrences of x in the matrix by $f(y_1, \dots, y_n)$. The function f is called a **Skolem function**.

Note that steps 1-5 preserve logical consequence. It is relatively easy to show that step 6 preserves satisfiability

- 5: Use the distributive laws $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$, $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ to transform the matrix into CNF.

$$\exists x \exists y \forall z ((p(x) \vee \neg p(y) \vee q(z)) \wedge (\neg q(x) \vee \neg p(y) \vee q(z)))$$

6: Skolemization

- ▶ If the prefix is of the form $\forall y_1 \dots \forall y_n \exists x$, let f be a new n -ary function symbol. Delete $\exists x$ from the prefix, replace all occurrences of x in the matrix by $f(y_1, \dots, y_n)$. The function f is called a **Skolem function**.
- ▶ If there are no universal quantifiers before $\exists x$ in the prefix, let a be a new constant. Eliminate $\exists x$ from the prefix and replace every occurrence of x in the matrix with a . The constant a is a **Skolem constant**.

Note that steps 1-5 preserve logical consequence. It is relatively easy to show that step 6 preserves satisfiability.

- 5: Use the distributive laws $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$, $(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ to transform the matrix into CNF.

$$\exists x \exists y \forall z ((p(x) \vee \neg p(y) \vee q(z)) \wedge (\neg q(x) \vee \neg p(y) \vee q(z)))$$

6: Skolemization

- ▶ If the prefix is of the form $\forall y_1 \dots \forall y_n \exists x$, let f be a new n -ary function symbol. Delete $\exists x$ from the prefix, replace all occurrences of x in the matrix by $f(y_1, \dots, y_n)$. The function f is called a **Skolem function**.
- ▶ If there are no universal quantifiers before $\exists x$ in the prefix, let a be a new constant. Eliminate $\exists x$ from the prefix and replace every occurrence of x in the matrix with a . The constant a is a **Skolem constant**.

$$\forall z ((p(a) \vee \neg p(b) \vee q(z)) \wedge (\neg q(a) \vee \neg p(b) \vee q(z)))$$

Note that steps 1-5 preserve logical consequence. It is relatively easy to show that step 6 preserves satisfiability.

- ▶ Read: Chapter 7, sections 7.1-7.4 of [Ben-Ari, 2001].
- ▶ Items of interest (no proofs required):
 - ▶ Predicate logic language: syntax, semantics(interpretation, model).
 - ▶ Herbrand's universe, Herbrand base, Herbrand interpretation
 - ▶ Herbrand's theorem, the significance of Herbrand's theorem.
 - ▶ Clausal form of first order formulae: Skolemization (Skolem constants, Skolem functions), transformation algorithm.



Ben-Ari, M. (2001).

Mathematical Logic for Computer Science.

Springer Verlag, London, 2nd edition.



Crăciun, A. (2005-2010).

Logic for Computer Science.