LIN241, Winter 2021
Lecture 6 summary: Predicate Logic

1. Translating English sentences into Predicate Logic

(1) Predicate Logic allows us to think about logical properties of predicates (like verbs) and quantifiers (like **every** or **some**). In this lecture, we put aside quantifiers, which will be discussed in lecture 7, and we restrict our attention to predicates and their arguments.

(2) Our simplified version of Predicate logic will contain three types of expressions:

- Individual constants, which we represent with lower case letters: a, b, c, …
- Predicates, which we represent with upper case words: LIKE, SMOKE, …
- Sentential connectives: ~, &, ∨, →, ↔

(3) We can form formulas of Predicate Logic by combining predicates with individual constants, which will serve as their arguments:

HAPPY(m)

LIKE(j, k)

…

(in lecture 7, we will discuss the use of variables in addition to individual constants)

(4) These formulas can be combined with one another using sentential connectives:

~HAPPY(m)

MEET(j,k) & LIKE(j,k)

SLEEP(m) → HAPPY(m)

…

(5) The individual constants of Predicate Logic are interpreted like proper names. Their denotation is not variable. To illustrate, we can translate sentence (5a) into Predicate Logic as (5b)

a. **Joan is happy**.

b. HAPPY(j)

In (5b), the individual constant j is used to translate the proper name **Joan**.

(6) Note that individual constants don't have to denote people. In principle, individual constants can refer to any kind of entity: individuals, animals, inanimate objects, abstract objects…

In practice though, we will mostly use individual constants to translate proper names and in most cases, these will be the names of people, although we will come across names for other types of entities (e.g., places like **Toronto**).

(7) By convention, the order of arguments of predicates corresponds to the order Subject-Direct Object-Oblique[1] in the active voice:

a. **Lupita introduced Christine to Alberto**.

INTRODUCED(l,c,a)

The translation remains the same when the verb is put in the passive voice:

b. **Christine was introduced to Alberto by Lupita**.

INTRODUCED(l,c,a)

(8) Predicates are used to translate English verbs, as well as non-verbal predicates introduced by the copula **be**:

Alex is happy	HAPPY(a)
Alex is a linguist	LINGUIST(a)

(9) In some cases, a prepositional phrase is used as a predicate of an English sentence, and in that case we translate the preposition as a predicate:

Alex is in Paris	IN(a,p)
Toronto is in Canada	IN(t,c)

[1] An oblique is an argument that is introduced by a preposition, like **Chris** in **Jess introduced Alex to Chris**.

(10) Rules of thumb for translating simple sentences of English (without pronouns and quantifiers) into Predicate logic:

1. If the sentence contains several clauses, break the sentence into simple clauses.

   Example: **[$_{\text{clause 1}}$ Alex fed Cooper] and [$_{\text{clause 2}}$ Cooper is happy]**

2. For each clause:

   a) translate each predicative expression in the English sentence as a predicate. Make sure to use a predicate that has as many arguments as the original predicative expression. In particular, translate:

      - intransitive verbs as unary predicates (one argument)
      - transitive verbs as binary predicates (two arguments)
      - ditransitive verbs as ternary predicates (three argument)

      e.g., translate **fed** as FEED (two arguments) and **is happy** as HAPPY (one argument)

   b) Translate proper names as individual constants.

      e.g., translate **Alex** as a and **Cooper** as c in clause 1, and Cooper as c in clause 2

   c) If the predicate has more than one argument, introduce the arguments in the order Subject-Object-Oblique of the active voice.

      e.g. FEED(a,c) for clause 1 and HAPPY(c) for clause 2

3. If the sentence contains several clauses, connect the clauses using the appropriate connectives.

   e.g. FEED(a,c) & HAPPY(c)

2. Interpreting predicate logic: set theory

(11) In order to interpret formulas of predicate logic, we will need to use some basic notions of set theory. We introduced some notions in lecture 6, and will introduce some more in lecture 7.

(12) **Definition of "set":**

A set S is a collection of objects, called the members or elements of S.

(13) **List notation:**

One can define a set by giving a list of its members between curly brackets:

This is the set that consists of the elements a, b, and c: {a, b, c}

(13) **Predicate notation:**

One can also define a set by stating a condition that all its members must satisfy:
This is the set of every x such that x is green: {x : x is green}

(14) **Set membership:**

The symbol "∈" represents the relation of membership. It means "is a member of."
"a ∈ S" means "a is a member of S"

"a ∉ S" means "a is not a member of S"

(15) Some observations on membership:

- A set can be a member of another set. For instance: {3} ∈ {1, 2, {3}}
- Do not confuse the following two statements:
  - {3} ∈ S (this says that the set {3} is a member of the set S)
  - 3 ∈ S (this says that the number 3 is a member of the set S)
- Note that:
  - {3} ∈ {1, 2, {3}}
  - {3} ∉ {1, 2, 3}
  - 3 ∈ {1, 2, 3}
  - 3 ∉ {1, 2, {3}}

(16) **Identity of sets:**

Two sets are identical (are the same set) iff they have the same members.

(17) Observation on identity of sets:

- Order of members doesn't matter: {a, b, c} is the same set as {c, b, a}
- Repeating a member doesn't matter: {a, b, c} is the same set as {a, a, b, c}

(18) **The empty set**

The empty set is the set that has no members. It is denoted by the symbol ∅.

Another way to write the empty set is {}, so ∅ and {} are the same set.

(19) **Relations**:

Some expressions of English express relations between individuals: **like**, **in**, **give**, …

To illustrate, in the sentence **Jess likes Alex**, **likes** expresses a relation between Jess and Alex.

In set theory, binary relations (relations between two arguments) can be represented as sets of pairs of individuals.

For instance, let us imagine a world where there are only three individuals: Alex, Chris and Jess. Let us assume furthermore that:

- Alex likes Jess
- Jess likes Chris
- Nobody else likes anyone else

In this world, we can represent the relation expressed by **likes** as the following set:

{⟨Alex, Jess⟩, ⟨Jess, Chris⟩}

In this set, ⟨Alex, Jess⟩ is an ordered pair of individuals, whose first member is Alex, and whose second member is Chris.

We can also represent the relation expressed by **likes** as a set using predicate notation for sets:

{⟨x,y⟩ : x likes y} (this is the set of all ordered pairs ⟨x, y⟩ such that x likes y)

In the imaginary world that we were discussing above:

- ⟨Alex, Jess⟩ ∈ {⟨x,y⟩ : x likes y} (since Alex likes Jess)
- ⟨Alex, Chris⟩ ∉ {⟨x,y⟩ : x likes y} (since Alex doesn't like Chris)

3. Semantics of Predicate Logic

(20) We interpret formulas of Predicate Logic using models. We can think of a model as an abstract representation of possible worlds or situations.

(21) A model M is defined by:[2]

- a set of individuals U (the universe of the model)
- an interpretation $[\![\ ]\!]^M$ that gives us the semantic value of basic expressions of Predicate logic (such as individual constants and predicates).

[2] This is a simplified presentation of the notion of models of Predicate Logic.

(22) Here is a simple model M^1 whose universe consists of three individuals:

U = {Dana, Marina, Allan}

$[\![\text{d}]\!]^{M1}$ = Dana $[\![\text{m}]\!]^{M1}$ =Marina $[\![\text{a}]\!]^{M1}$ = Allan

$[\![\text{PERSON}]\!]^{M1}$ = {Dana, Marina, Allan}

$[\![\text{SING}]\!]^{M1}$ = {Dana} $[\![\text{DANCE}]\!]^{M1}$ = {Marina, Allan}

$[\![\text{LIKE}]\!]^{M1}$ = {⟨Marina, Marina⟩, ⟨Allan, Allan⟩, ⟨Dana, Marina⟩, ⟨Dana, Allan⟩}

In this model, Dana signs, Marina and Allan dance, Marina and Allan like themselves, Dana likes Marina, and Dana likes Allan.

(23) We can formulate general rules that allow us to interpret formulas of Predicate Logic in models. Here, we restrict our attention to formulas that do not use quantifiers and variables.

(24) Semantic rules for simple formulas:

- If t is an individual constant and P is a unary predicate:

$$[\![P(t)]\!]^M = T \text{ iff } [\![t]\!]^M \in [\![P]\!]^M$$

- If t_1 , ..., t_n is a sequence of n individual constants ($n \geq 2$) and R is a predicate of n-arguments:

$$[\![R(t_1, ..., t_n)]\!]^M = T \text{ iff } \langle [\![t_1]\!]^M, ..., [\![t_n]\!]^M \rangle \in [\![R]\!]^M$$

(25) The first of the two rules in (24) says that P(t) is true iff the individual denoted by t is a member of the set denoted by the predicate P.

The second rule says that $R(t_1, ..., t_n)$ is true iff the ordered sequence of individuals denoted by t_1 , ..., t_n is a member of the set of ordered sequences of n-individuals denoted by the predicate R.

(26) There are also rules for sentence connectives, which capture the same information as the truth-tables we used define these connectives in Propositional Logic:

Let φ and ψ be formulas of Predicate Logic, then:

$$[\![\neg\varphi]\!]^M = T \text{ iff } [\![\varphi]\!]^M = F$$

$$[\![\varphi \,\&\, \psi]\!]^M = T \text{ iff } [\![\varphi]\!]^M = T \text{ and } [\![\psi]\!]^M = T$$

$$[\![\varphi \vee \psi]\!]^M = T \text{ iff } [\![\varphi]\!]^M = T \text{ or } [\![\psi]\!]^M = T$$

$$[\![\varphi \rightarrow \psi]\!]^M = T \text{ iff } [\![\varphi]\!]^M = F \text{ or } [\![\psi]\!]^M = T$$

$$[\![\varphi \leftrightarrow \psi]\!]^M = T \text{ iff } [\![\varphi]\!]^M = [\![\psi]\!]^M$$

4. Applying model theoretic interpretation directly to English.

(27) We can use models to interpret English sentences directly, without translating them into formulas of predicate logic. To do so, we first need to build models of English, rather than models of Predicate Logic. This means that the expressions that will be interpreted by the function $[\![\ ]\!]^M$ of our model M will be words of English, rather than expressions of predicate logic.

Here is a model M^2, where English words are written in bold font:

U = {Dana, Marina, Allan}

$[\![\textbf{Dana}]\!]^{M2}$ = Dana $[\![\textbf{Marina}]\!]^{M2}$ = Marina $[\![\textbf{Allan}]\!]^{M2}$ = Allan

$[\![\textbf{person}]\!]^{M2}$ = {Dana, Marina, Allan}

$[\![\textbf{sing}]\!]^{M2}$ = {Dana}

$[\![\textbf{dance}]\!]^{M2}$ = {Marina, Allan}

$[\![\textbf{like}]\!]^{M2}$ = {⟨Marina, Marina⟩, ⟨Allan, Allan⟩, ⟨Dana, Marina⟩, ⟨Dana, Allan⟩}

(28) We also need to define rules of interpretation for English. This is more complicated than it is for Predicate Logic. Here, we just consider two simple rules.

(29) The first rule tells us how to interpret a simple sentence S with a subject NP that is not a quantifier:

$$[\![\ [_S \text{ NP VP }]\ ]\!]^M = \text{T iff } [\![\text{NP}]\!]^M \in [\![\text{VP}]\!]^M$$

This rule tells us that a sentence $[_S$ NP VP] is true iff the individual denoted by the subject NP is a member of the set denoted by the VP.

If the VP is just made up of an intransitive verb, like $[_{VP}$ **smokes**], the VP will denote the set denoted by the verb.

However, if the VP is made up of a transitive verb plus its object, we need another rule to interpret the VP.

(30) The second rule tells us how to interpret transitive VPs:

$$[\![\ [_{VP} \text{ V NP }]\ ]\!]^M = \{x: \langle x, [\![\text{NP}]\!]^M\rangle \in [\![\text{V}]\!]^M\}$$

This says that $[_{VP}$ V NP] denotes the set of all individuals x, such that x forms a pair $\langle x, [\![\text{NP}]\!]^M\rangle$ with the individual $[\![\text{NP}]\!]^M$ denoted by the object, and this pair is a member of the set $[\![\text{V}]\!]^M$ denoted by the verb.

(31) We can apply the second rule to a VP like **loves Jess** if we assume that **loves** is the V head and **Jess** is the object NP. Then, according to this rule, the VP denotes the set of individuals who love Jess:

$$[\![\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}\ ]]\ ]\!]^M = \{x: \langle x, [\![\ [_{NP}\ \textbf{Jess}\ ]\ ]\!]^M\rangle \in [\![\textbf{loves}]\!]^M\}$$

(32) Here is an example that uses these two rules to interpret the sentence **Chris loves Jess**:

i. $[\![\ [_S\ [_{NP}\ \textbf{Chris}]\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}\ ]]]\ ]\!]^M = T$ iff $[\![\ [_{NP}\ \textbf{Chris}\ ]\ ]\!]^M \in [\![\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}]]\ ]\!]^M$

ii. $[\![\ [_S\ [_{NP}\ \textbf{Chris}]\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}\ ]]]\ ]\!]^M = T$ iff $[\![\ [_{NP}\ \textbf{Chris}\ ]\ ]\!]^M \in \{x: \langle x, [\![\ [_{NP}\ \textbf{Jess}\ ]\ ]\!]^M\rangle \in [\![\textbf{loves}]\!]^M\}$

iii. $[\![\ [_S\ [_{NP}\ \textbf{Chris}]\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}\ ]]]\ ]\!]^M = T$ iff Chris $\in \{x: \langle x, [\![\ [_{NP}\ \textbf{Jess}\ ]\ ]\!]^M\rangle \in [\![\textbf{loves}]\!]^M\}$

iv. $[\![\ [_S\ [_{NP}\ \textbf{Chris}]\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}\ ]]]\ ]\!]^M = T$ iff Chris $\in \{x: \langle x, \text{Jess}\rangle \in [\![\textbf{loves}]\!]^M\}$

v. $[\![\ [_S\ [_{NP}\ \textbf{Chris}]\ [_{VP}\ \textbf{loves}\ [_{NP}\ \textbf{Jess}\ ]]]\ ]\!]^M = T$ iff $\langle \text{Chris, Jess}\rangle \in [\![\textbf{loves}]\!]^M$

Note that rule (29) is used in step i, and rule (30) is used to go from step i to step ii

In order to go from step ii to iii and from step iii to iv, we just substitute "$[\![\ [_{NP}\ \textbf{Chris}\ ]\ ]\!]^M$" by "Chris" and then "$[\![\ [_{NP}\ \textbf{Jess}\ ]\ ]\!]^M$" by Jess.

In order to go from step iv to step v, we just use set theory to simplify the expression on the right hand of iv.