Atomic Sentences of Predicate Logic

Predicate Logic and Sentence Logic

- Predicate logic is an *Extension* of sentence logic.
  - All items of the vocabulary of sentence logic are items of the vocabulary of predicate logic.
  - All sentences of sentence logic are sentences of predicate logic.
  - All conventions for sentence-logic syntax apply to predicate-logic syntax.
- Predicate logic contains vocabulary items and sentences that are not part of sentence logic.

Syntax Unique to Predicate Logic

- Several new vocabulary items are found in predicate logic.
  - Predicates
  - Names
  - Function symbols
  - Variables
  - Quantifier symbols
- In this segement of the course, we will restrict our attention to the first three new vocabulary items.

Subject and Predicate

- "Predicate logic deals with sentences which say something about someone or something" (II, 2).
- The someone or something about which something is said by a sentence is the **subject** of the sentence.
- The something which is said about someone or something is the **predicate** of the sentence.
- Example: 'Adam is blonde'
  - 'Blond' is the predicate of the sentence.
  - 'Adam' is the subject of the sentence.

Object and Property

- The subject of a subject-predicate sentence refers to an **object**.
  - 'Adam' refers to the person, Adam.
- The predicate of a subject-predicate sentence indicates **property** or a **relation**.
  - 'Blond' refers to the property *being blond*.
- The property indicated by the predicate is attributed to the object referred to by the name.

Predicate		**Subject**
Property	Attributed to	Object
Being blond	Property of	Adam

Relations

- In some sentences, a predicate has multiple subjects.
  - 'Eve loves Adam'.
- Each subject refers to an object.
  - 'Adam' refers to Adam.
  - 'Eve' refers to Eve.
- The **two-place** predicate of such a sentence indicates a relation holding between what the subjects refer to.

Predicate		**Subject**$_1$		**Subject**$_2$
Relation	Holds between	Object$_1$		Object$_2$
Loving	Relation of	Eve	to	Adam

Symbolization of Subjects and Predicates

- Natural-language predicates are symbolized in Predicate Logic using upper-case Roman letters (with or without integer subscripts) which are called *Predicates*.
  - 'B' might stand in for the one-place English predicate 'blond'.
  - 'L' might stand in for the two-place English predicate 'loves'.
- Natural-language subjects are symbolized in Predicate logic using lower-case Roman letters (with or without integer subscripts), which are called *Names*.
  - 'a' might stand in for the English name 'Adam'
  - 'e' might stand in for the English name 'Eve'
- The choice of predicates and names of Predicate logic to stand in for natural-language predicates and names is arbitrary, except that it must be consistent in a given context.

Atomic Sentences of Predicate Logic

- The most basic sentences of Predicate Logic are *Atomic Sentences*.
- There are two types of atomic sentences.
  - An atomic sentence of Sentence Logic, which is a predicate followed by no names.
  - A predicate followed by any number of names (which are called *Arguments*).
- Examples:
  - 'S'
  - 'Ba'
  - 'Lea'

Interpreting Atomic Sentences

- As merely syntactical entities, atomic sentences of Predicate Logic have no meaning.
- The assignment of meanings to sentences is called "semantics".
- Atomic sentences can be given meaning as with Sentence Logic:
  - Stand-ins for (*Transcriptions* of) natural language sentences.
  - Having truth-values t or f.

Transcriptions

- Transcriptions have two components:
  - A *Transcription Guide* which associates vocabulary of Predicate Logic with natural-language expressions.
  - A sentence or sentences of Predicate Logic which is supposed to capture the structure of the natural-language sentence or sentences, or
  - A sentence or sentence in natural language which is supposed to capture the structure of the Predicate Logic sentence or sentences.
- The transcription guide itself contains two components:
  - An object for every name of Predicate Logic used in the transcription.
  - A specification of the predicates of Predicate Logic used in the transcription.

An Example

- Suppose we want to transcribe the sentence ‘Adam is blond, but Eve is not’.
- We choose to let ‘a’ stand for Adam and ‘e’ for Eve.
  - a: Adam e: Eve
- We specify a one-place predicate ‘B’ by writing ‘Bx’ (and a two-place predicate ‘L’ by writing ‘Lxy’, etc.) followed by a quasi-English expression signifying a predicate.
  - Bx: x is blond
- The transcription is: ‘Ba & $\sim$Be’.

Truth-Conditions for Atomic Sentences

- Let us call a “proper” atomic sentence of Predicate Logic one which contains names and therefore is not a sentence of Sentence Logic.
- We could assign proper atomic sentences truth-values in the same way as in Sentence Logic.
- This is done by expanding Sentence-Logic “cases” to *Minimal Interpretations*.

Interpretations of Sentences

- Just as a truth-table only specifies truth-values for the sentences that appear in it, an interpretation in Predicate Logic may be limited in its scope.
- The range of application of an interpretation is determined by three sets it contains:
  - Sentence Letters
  - Names
  - Predicates
- If all the names, predicates, and sentence-letters occuring in a sentence **X** occur in the interpretation, then it is an *Interpretation of the Sentence* **X**.

Minimal Interpretations

- The truth-value of atomic sentences in an interpretation can be assigned just as in sentence logic.
- Consider interpretations which contain ‘B’, ‘e’, and ‘a’, and nothing else.
- There are exactly two atomic sentences to which the interpretations apply: ‘Ba’ and ‘Be’.

- This allows us to list four possible interpretations in the guise of a truth-table.

	Ba	Be
interpretation 1	t	t
interpretation 2	t	f
interpretation 3	f	t
interpretation 4	f	f

State Descriptions

- Each minimal interpretation just listed can be represented in the form of a sentence of Predicate Logic.
- Such a sentence was called a "state description" by Rudolf Carnap.
- A state description is a conjunction of negated or unnegated atomic sentences.
  - The sentence is unnegated if the interpretation assigns it the value true.
  - The sentence is negated if the interpretation assigns it the value false.

	Ba	Be	State Description
interpretation 1	t	t	Ba & Be
interpretation 2	t	f	Ba & $\sim$Be
interpretation 3	f	f	$\sim$Ba & Be
interpretation 4	f	f	$\sim$Ba & $\sim$Be

Domains

- Minimal interpretations treat proper atomic sentences in the same way as atomic sentences of Sentence Logic.
- To make them more useful, we need a way to link the predicates and names occurring in the sentences with objects in the world.
- An interpretation should include "a specification of the objects of which each predicate is true and the objects of which each predicate is false".
- A minimal interpretation says nothing about objects.
- We can make an interpretation more robust by supplementing it with:
  - A *Domain* consisting of at least one object.
  - An association of each name in the interpretation with at least one object in the domain.

An Example

- Suppose the domain of an interpretation consists of Adam and Eve.
- This can be written as: D = {Adam, Eve}.
- The interpretation might assign Adam to ‘a’ and Eve to ‘e’.
- Suppose further that it is the case that Adam is blond and Eve is not blond.
- Given our earlier remarks, we might want to say that ‘B’ stands for the property of being blond.
- Then, we would conclude that on this interpretation, ‘Ba’ is true and ‘Be’ is false.

Intensional and Extensional Interpretations

- An interpretation that makes predicates stand for properties is called “intensional”.
- Intensional interpretations were studied by Leibniz in the eighteenth century.
- Modern logic, however, mostly does not use intensional interpretations.
- Instead, it takes a given predicate to stand for a set of objects that have a given property.
- Such an interpretation is “extensional”.

One-Place Predicates

- Suppose we have interpretations which include the predicate ‘B’ and the names ‘a’ and ‘e’ and nothing else.
- Consider a class of interpretations whose domain is {Adam, Eve}, and where ‘a’ stands for Adam, ‘e’ for Eve.
- There are four possible extensions for ‘B’, and therefore four possible interpretations in this class.

	a	e	B
interpretation 1	Adam	Eve	Adam, Eve
interpretation 2	Adam	Eve	Adam
interpretation 3	Adam	Eve	Eve
interpretation 4	Adam	Eve	Nobody

Determining Truth-Values

- It is easy to see how the truth-values of ‘Ba’ and ‘Be’ are determined given these interpretations.

	a	e	B	Ba	Be
interpretation 1	Adam	Eve	Adam, Eve	t	t
interpretation 2	Adam	Eve	Adam	t	f
interpretation 3	Adam	Eve	Eve	f	t
interpretation 4	Adam	Eve	Nobody	f	f

- The truth-condition for atomic sentences with one-place predicates is: if the object designated by the name is in the extension of the predicate, then the sentence is true; otherwise, it is false.

Extensions of Relation Symbols

- Predicates of two-places or more are called “relation symbols”.
- The extensions of relation symbols are sets of pairs, triples, quadruples, etc., depending on the number of arguments the predicate has.
- We represent these n-tuples with angled brackets.
  - The pair (2-tuple) consisting of Adam and Eve is represented as ⟨Adam, Eve⟩.
  - The triple (3-tuple) consisting of Adam, Eve, and Cid is represented as ⟨Adam, Eve, Cid⟩.
- Strictly speaking, the extension of a one-place predicate is a set of 1-tuples.
  - The 1-tuple consisting of Adam is represented as ⟨Adam⟩.

An Example

- Let an interpretation listing the one-place predicate ‘L’ have a domain consisting of Adam and Eve.
- There are four possible items in the extension of ‘L’:
  - ⟨Adam, Adam⟩
  - ⟨Adam, Eve⟩
  - ⟨Eve, Adam⟩
  - ⟨Eve, Eve⟩
- An interpretation representing the case where Adam and Eve love only themselves would specify the extension of ‘L’ as consisting of ⟨Adam, Adam⟩ and ⟨Eve, Eve⟩.

General Truth-Condition

- With the given interpretation, we can once again see what the truth-values of various sentences are.

a	e	L	Laa	Lae	Lea	Lee
Adam	Eve	⟨Adam, Adam⟩⟨Eve, Eve⟩	t	f	f	t

- An atomic sentence with an n-place predicate is true in an interpretation if the n-tuple consisting of the referents of the arguments (from left to right) is a member of the extension of the predicate; the sentence is false otherwise.

$$\begin{array}{lccc} \text{L} & \text{a} & & \text{a} \\ & \downarrow & & \downarrow \\ & \langle\text{Adam}, & & \text{Adam}\rangle \end{array}$$

Functions

- Predicate Logic may be enhanced by the addition of a way of forming name-like expressions: *Function Symbols*.
- Function symbols consist of italicized lower-case Roman letters, followed by a set of parentheses and containing from 0 to n commas, where n is a finite number.
  - $f(\,)$
  - $g(\,,\,)$
  - $h(\,,\,,\,)$
- Filling the spaces in the function symbols are either names or other filled-in function symbols.
  - f(a)
  - $g(f\text{(a)},f\text{(b)})$
  - $h(g(f\text{(a)},g\text{(b)},\text{c}))$

Interpretation of Function Symbols

- Function symbols are intended to represent functions.
- We illustrate the notion of a function by beginning with one-place function symbols, where what fills the blank is a name, for example, f(a).
- In an interpretation, the name designates exactly one member of the domain, which is an *Argument* of the function that the function symbol signifies.
- The filled-in function symbol itself designates exactly one member of the domain (the *Value* of the function).

An Example

- Consider a domain with two members, Adam and Eve.
- Informally, we might consider 'f( )' to designate the spouse function (which presumes monogamy).
- Adam is the spouse of Eve, and Eve is the spouse of Adam.
- So when Adam is the argument, Eve is the value, and when Eve is the argument, Adam is the value.

a	e	f(a)	f(e)
Adam	Eve	Eve	Adam

Applying the Example

- The interpretation of atomic sentences can be applied to the interpretation of function symbols.
- Specifically, we can determine values of the following sentences.

	a	e	f(a)	f(e)	B	Bf(a)	Bf(e)
interp. 1	Adam	Eve	Eve	Adam	Adam, Eve	t	t
interp. 2	Adam	Eve	Eve	Adam	Adam	f	t
interp. 3	Adam	Eve	Eve	Adam	Eve	t	f
interp. 4	Adam	Eve	Eve	Adam	Nobody	f	f

Many-Place Functions

- Functions of more than one place are treated similarly to functions of one place.
- There always a single value, no matter how many arguments the function might have.
- For example, consider a domain which consists of the positive integers, 1, 2, 3, ...
- An addition function takes any two positive integers as arguments and returns a single value, the sum of the two numbers.
- If 'a' designates 1, designates 'b' 2, and f(,) designates the addition function, then 'f(a,b)' designates 3.

Values of Functions as Arguments of Functions

- Since the value of a function is an object in the domain, it can serve as the argument of a function (even the same one).
- A natural example is the application of the addition function to the result of addition.
  - (1 + 2) + 1
- If we use the interpretations just given, we can represent this as: $f(f(\text{a,b}),\text{a})$.
- 'f(a,b)' represents the number 3, so '$f(f(\text{a,b}),\text{a})$' represents the result of the addition of 3 and 1, i.e., 4.

Constant Terms

- We shall call a name a *Constant* or *Constant Term*.
- A function symbol which is filled in by names is also a constant term.
- More generally, a function symbol which is filled in by constant terms is also a constant term.
- Nothing else is a constant term.

The Identity Predicate

- A special two-place predicate is the *Identity* predicate '='.
- By convention, the identity predicate is placed between two constant terms, rather than in front of them.
  - a = b
  - $f(\text{a}) = g(f(\text{b}),f(\text{c}))$
- The identity predicate is always interpreted in the same way.
- If both constant terms designate the same member of the domain, the identity sentence containing them is true; otherwise, it is false.

An Example

- In the following example, we look at four interpretations that differ in the function designated by the function symbol '$f(\ )$'.

	a	e	f(a)	f(e)	a = e	a = f(e)	f(e) = f(a)
i. 1	Adam	Eve	Adam	Adam	f	t	t
i. 2	Adam	Eve	Adam	Eve	f	f	f
i. 3	Adam	Eve	Eve	Adam	f	t	f
i. 4	Adam	Eve	Eve	Eve	f	f	t

Metavariables for Atomic Sentences

- The metavariables '**s**' and '**t**' designate any constant terms (name or filled-in function symbol).
- Note that in the text, '**s**' and '**t**' are restricted to designating names.
- The metavariables '**P( )**', '**Q( )**', and '**R( )**' indicate sentences of Predicate Logic containing terms.
- '**P(s)**' indicates a sentence of Predicate Logic containing the term designated by '**s**', and '**P(t)**' indicates a sentence containing the term designated by '**t**'.
  - If '**s**' indicates the constant 'a', then 'Lea' can be represented as '**P(s)**'.
- '**P(t/s)**' indicates the result of substituting **t** in all the places where **s** occurs.

Introduction Rule for the Identity Predicate

- Identity Introduction (=I): **t** = **t** may be written on any line of a derivation.
- The rule is truth-preserving because the designation of any constant term is fixed by the interpretation, so that the designation of **t** is always the same, in which case ⟨the designation of **t**, the designation of **t**⟩ is in the extension of '=', no matter what **t** designates.

Elimination Rule for the Identity Predicate

- Identity Elimination (=E): when **s** = **t** occurs on an accessible earlier line, then given **P(s)**, it follows that **P(t/s)**.
  - a = b Lae Lbe
  - a = b Lbe Lae
- The rule is truth-preserving because the truth of **s** = **t** requires that **s** and **t** designate the same individual, and by the semantics, anything holding of what **s** designates holds of what **t** designates.

Summary

- The subject of a subject-predicate English sentence is represented in a transcribed Predicate Logic sentence by a constant term of Predicate Logic, whose designation is determined ny the interpretation of the sentence.
  - A name
  - A filled-in function symbol
- The predicate of a subject-predicate English sentence is represented in a transcribed Predicate Logic sentence by a predicate of Predicate Logic.

  - An n-place predicate whose extension is determined by the interpretation.
  - A two-place identity predicate '=' whose extension is the same on any interpretation.

- An atomic sentence of Predicate Logic consists of an n-place predicate followed by n constant terms.

Objectual Semantics

- The semantics that has been introduced here is sometimes called "objectual".
- The original semantics for Predicate Logic due to Alfred Tarski is objectual in character.
- The semantics is called objectual because the truth-conditions for atomic sentences are based on the designation of objects in the domain by constant terms and on the objects or ordered n-tuples of objects in the extensions of predicates.

Presentation of the Domain

- The formulation of interpretations in Exercises 2-1 and 2-2 differs from that in 2-5, with the latter being closer the the formulation presented here.
- In the earlier exercises, the domain is specified using the names given it by the interpretation
  - D = {b,d}.
- "In practice, we often specify the domain of an interpretation simply by giving the interpretation's name for those objects" (II, 16).
- In the 2-5 exercises, the domain is specified directly,
  - All integers, 1, 2, 3, 4, ...

Determining Truth-Values

- The specification of "which one place predicates apply to which individual objects, which two place predicates apply to which pairs of object, and so on" is presented in the earlier exercises through a state description.
- A state description summarizes the result of the atomic sentences' getting a truth-value.
  - $\sim$Tb & Td & Kbb & $\sim$Kbd & $\sim$Kdb & Kdd.
- The application of predicates to objects is left to the general knowledge of the reader.
  - That 1 is odd, or that not all integers are greater than 17.

Tarski Semantics

- We have developed semantics for atomic sentences in the style of Tarski.
- This style most closely resembles the later exercises.
- The Tarski-style semantics makes an explicit assignment of members of the domain to names.
  - The domain is {Betty, Dylan}, with 'b' designating Betty, and 'd' designating Dylan.
- The extensions of predicates of Predicate Logic, and the meanings of function symbols are given.
  - The extension of 'T' is $\{\langle \text{Dylan} \rangle\}$
- From this we can determine truth-values.
  - Betty is not in the extension of 'T', so 'Tb' is false, and '$\sim$Tb' is true.

From State Descriptions to Explicit Intepretations

- The state-description presentation of the semantics requires that an interpretation have a domain, which is specified by names of Predicate Logic.
- If we specify the domain in this way, we have not said what the items in the domain are, unless we associate the Predicate Logic names with natural-language names (as in a transcription guide).
  - D = {b,d}, b: Betty, d: Dylan.
- State descriptions, along with the transcription guide, can be used to determine which n-tuples of objects are in the extensions of which predicates.
- '$\sim$Tb' indicates that the one-tuple of the object in the domain designated by 'b' (Betty) is not in the extension of 'T'.