Sentence Logic

- Sentence logic deals with sentences of a natural language that are either true or false (I, 5).
- Sentence logic ignores the internal structure of simple sentences (I, 5).
- Sentence logic is concerned with sentences which are compounded in a certain way.
- A primary goal of sentence logic is to enable the evaluation of a certain class of arguments in natural language (I, 2).
- In an argument, a sentence that is the argument’s conclusion is claimed to be supported by a set of sentences that are its premises.

Deductive Validity

- The kind of support investigated by sentence logic is that of Deductive Validity.
- “Valid Deductive Argument: An argument in which, without fail, if the premises are true, the conclusion will also be true” (I, 3).
- In general, the source of deductive validity in sentence logic lies in the way in which the sentences in the argument are compounded.
- So the main item of business in sentence logic is to investigate the properties of the devices which allow the formation of compound sentences from simpler sentences.

1 Syntax of Sentence Logic I

1.1 Notions of Syntax

Syntax and Sentence Logic
• “A fact of Syntax is a fact which concerns symbols or sentences insofar as the fact can be determined from the form of the symbols or sentences, from the way they are written” (II, 161).

• There are two central syntactical facts investigated by sentence logic (II, 161):
  – Whether or not a string of symbols is a Sentence of sentence logic.
  – The application of Rules of Inference to sentences of sentence logic.

• Sentences of sentence logic are denoted by boldface Metavariables ‘Q’ through ‘Z’ (I, 18).

• Sets of sentences of sentence logic are denoted by italicized boldface metavariables ‘X’ through ‘Z’ (II, 158).

1.2 Formation Rules
Vocabulary of Sentence Logic
• The vocabulary of sentence logic consists of three kinds of items:
  – Sentence Letters (I, 5-6):
    * ‘A’, ‘B’, ‘C’, …, ‘Z’ (with or without integer subscripts), ⊥ (falsum)
  – Connectives (I, 6, 50, 54)
    * ∼ (Sign of Negation)
    * ∨ (Sign of Disjunction)
    * & (Sign of Conjunction)
    * ⊃ (Sign of the Conditional)
    * ≡ (Sign of the Biconditional)
  – Punctuation marks (I, 11-12):
    * ‘(’, ’)’, ‘[’, ’]’, ‘{’, ’}’

Sentences of Sentence Logic
• The sentences (well-formed formulas, or wffs) of sentence logic are determined by the following Formation Rules (I, 16, 55):
  – i) All capital letters ‘A’, ‘B’, ‘C’, …, ‘Z’ (with or without integer subscripts), and ‘⊥’ are wffs (Atomic Sentences).
  – ii) If X is a wff, then so is (∼X) (Negated Sentence).
  – iii) If X and Y are wffs, then so is (X & Y) (Conjunction).
  – iv) If X and Y are wffs, then so is (X ∨ Y) (Disjunction).
  – v) If X and Y are wffs, then so is (X ⊃ Y) (Conditional).
  – vi) If X and Y are wffs, then so is (X ≡ Y) (Biconditional).
vii) Nothing else is a wff of sentence logic.

• Conventions (I, 12-14):
  – Square or curly brackets may replace parentheses.
  – Outermost punctuation marks may be dropped if there is no further compounding.
  – Punctuation marks around negations may be dropped.

2 Semantics of Sentence Logic

2.1 Truth Tables

Semantics and Sentence Logic

• “A fact of Semantics . . . concerns the referents, interpretation, or (insofar as we understand this notion) the meaning of symbols and sentences” (II, 161).

• There are two distinct ways in which we interpret the symbols of sentence logic:
  – Informally: as stand-ins for (Transcriptions of) natural language sentences (I, Chs. 2, 4).
  – Formally: as having one of two Truth Values, true or false (t or f, respectively) (I, 8).

• The formal interpretation of sentence logic will serve as a guide to how to transcribe sentences of natural language into sentence logic.

Truth Values

• We can study the semantical facts about sentence logic without knowing anything about the natural-language sentences for which they might stand in.

• Any atomic sentence may be interpreted either as being true or as being false.

• The assignment of truth values to atomic sentences is called a Case (I, 9).

• The truth value of a compound sentence is strictly determined by the truth values of its component parts.
  – Sentence logic is Truth-Functional.

• The way in which the truth value of a compound sentence is determined can be summarized in a table called a Truth Table (I, 8).
**Truth Table for Falsum**

The symbol ‘⊥,’ which is intended to stand for any sentence that cannot be true, is always assigned the truth value f.

<table>
<thead>
<tr>
<th></th>
<th>⊥</th>
</tr>
</thead>
<tbody>
<tr>
<td>All cases</td>
<td>f</td>
</tr>
</tbody>
</table>

**Truth Table for Negation**

The negation of X takes the opposite of the truth value assigned to X in the given case.

<table>
<thead>
<tr>
<th>X</th>
<th>~X</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

**Truth Table for Conjunction**

A conjunction X & Y is true in a case if and only if both X and Y are true in that case.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X &amp; Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

**Truth Table for Disjunction**

A disjunction X ∨ Y is true in a case if and only either X or Y is true in that case.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X ∨ Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

**Truth Table for the Conditional**

A conditional X ⊃ Y is true in a case if and only if either X is false in that case or Y is true in that case.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X ⊃ Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>
Truth Table for the Biconditional

A biconditional $X \equiv Y$ is true in a case if and only if both $X$ and $Y$ have the same truth value in that case.

<table>
<thead>
<tr>
<th>Case</th>
<th>X</th>
<th>Y</th>
<th>$X \equiv Y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>t</td>
<td>t</td>
<td>t</td>
</tr>
<tr>
<td>2</td>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>3</td>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>4</td>
<td>f</td>
<td>f</td>
<td>t</td>
</tr>
</tbody>
</table>

Deductive Validity in Sentence Logic

- An argument in sentence logic consists of a set $X$ of wffs (premises) and a sentence $Y$ (conclusion).
- “To say that an argument (expressed with sentences of sentence logic) is Valid is to say that any assignment of truth values to sentence letters which makes all of the premises true also makes the conclusion true” (I, 47).
- We symbolize this relation of validity as follows: $X \models Y$.
- An argument from $X$ to $Y$ is invalid ($X \not\models Y$) if and only if it is not valid, i.e., if in some case all the sentences in $X$ are true and $Y$ is false.
- A Counterexample is a case which shows an argument to be invalid by making all the premises true and the conclusion false (I, 47-8).

Other Semantical Properties of Sentence Logic

- Sentences $X$ and $Y$ of sentence logic are Logically Equivalent if and only if in all possible cases they have the same truth value (I, 29-30).
- Sentence $X$ of sentence logic is a Logical Truth (or Tautology) if and only if it is true in all possible cases (I, 38).
- Sentence $X$ of sentence logic is a Contradiction if and only if it is false in all possible cases (I, 38).

2.2 Application of Sentence Logic

Transcription and Connectives

- The semantical facts about compound sentences of sentence logic suggest how to use them to transcribe compound sentences of the natural language.
  - $\sim$: not, it is not the case that
  - $\&$: and, but
  - $\lor$: or (inclusive)
– ⊃: if-then (“material” conditional)
– ≡: if and only if (“material” biconditional)

• The transcriptions for negation, conjunction, and disjunction are less controversial than those for the conditional and biconditional (II, Ch. 4).
• Ordinarily, we do not transcribe natural language sentences as ‘⊥’, as this symbol is useful only within sentence logic itself.

Using the Semantics of Sentence Logic

• The semantics of sentence logic can be used to show the validity or invalidity of some natural-language arguments.
• Validity or invalidity of natural-language arguments can be shown using the semantics only if the sentences making up the argument are adequately transcribed (I, 25).
• If the sentences of the argument are adequately transcribed and the argument in sentence logic is valid, then the natural-language argument is valid.
• If the argument is adequately transcribed, the argument in sentence logic is invalid, and the transcription reveals all of its logical structure, then the natural-language argument is invalid.
• Predicate logic is needed because sentence logic does not reveal all the logical structure of many natural-language arguments (II, 1-2).

3 Syntax of Sentence Logic II

3.1 Rules of Inference

Natural Deduction

• It is possible to determine the validity or invalidity of natural-language arguments using the sentences of sentence logic purely syntactically (i.e., without interpreting them at all).
• This is done using Rules of Inference which relate sets of sentences to a given sentence (I, 60).
• We here use the technique known as Natural Deduction (after Gerhard Gentzen), which was originally formulated in 1929 by Stanislaw Jaśkowski.
• The formulation of natural deduction rules used here is due to Frederick Fitch (1952).
• The distinctive feature of Fitch’s rules is their use of Subderivations (I, 65).
Rules of Inference

- Roughly, a Derivation is the result of the application of inference rules.
- A rule of inference allows one to write down a sentence \( Y \) given that one has already written down some set of sentences \( X \).
  - For example, given ‘A’ and ‘A \( \supset \) B’, one may write down ‘B.’
- We want our rules of inference to be Truth-Preserving (I, 62).
  - A rule is truth-preserving (or sound) if and only if there is no possible case in which all the sentences of \( X \) are true and \( Y \) is false.

Classifying the Rules

- In any system of sentence logic, some rules of inference are “primitive” while others are “derived” (I, 98).
  - A primitive rule is taken as basic.
  - A derived rule is a shortcut, which gives the same result of a more complicated combination of uses of primitive rules.
- For each connective, there is one primitive rule which results in it being “introduced” and another which results in its being “eliminated”.
- There is a further rule which allows any sentence to be repeated, subject to restrictions.

Differences in Primitive and Derived Rules

- It is possible to take a number of different sets of rules as primitive.
- MFLP, LPL and TLB each have different primitive inference rules.
- In this class, we will be using the set of inference rules based on TLB, with the addition of two rules from LPL.

3.2 Some Simple Rules

Conjunction Elimination

- With Conjunction Elimination, if a conjunction \( X \& Y \) occurs at any point of a derivation, either of its two conjuncts may be written down.
• Here is a schematic representation of the rule, which works for either side of the conjunction sign.

\[
\begin{array}{c}
\vdots \\
X & Y \\
\vdots \\
X
\end{array}
\]

Disjunction Introduction

• With Disjunction Introduction, if a sentence $X$ occurs at any point of a derivation, either $X \lor Y$ or $Y \lor X$ may be written.

• Here is a schematic representation of the rule, which works for either side of the disjunction sign.

\[
\begin{array}{c}
\vdots \\
X \\
\vdots \\
X \lor Y
\end{array}
\]

Conditional Elimination

• With Conditional Elimination, if a sentence $X \supset Y$ and $X$ occur, then $Y$ may be written.

• Here is a schematic representation of the rule, where the order of $X$ and $X \supset Y$ is irrelevant.

\[
\begin{array}{c}
\vdots \\
X \\
\vdots \\
X \supset Y \\
\vdots \\
Y
\end{array}
\]

Biconditional Elimination

• With Biconditional Elimination, if a sentence $X \equiv Y$ and $X$ occur, then $Y$ may be written.
Here is a schematic representation of the rule, which works for either side of the biconditional sign.

\[
\begin{array}{c}
\vdots \\
X \\
\vdots \\
X \equiv Y \\
\vdots \\
Y \\
\end{array}
\]

### 3.3 Preservation of Truth

#### Soundness of Conjunction Elimination

- The rule of Conjunction Elimination is truth-preserving because if \( X \& Y \) is true, then \( X \) is true and \( Y \) is true (I, 70).

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>X &amp; Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>t</td>
<td>t ←−</td>
</tr>
<tr>
<td>t</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>t</td>
<td>f</td>
</tr>
<tr>
<td>f</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

- The soundness of the other simple rules given below is established similarly.

### 3.4 Scope Lines

#### Scope Lines

- A *Scope Line* is a device used to keep track of the premises of an argument or of any assumptions made in the course of the argument (I, 62).

- In the following schema, the scope line indicates two premises of an argument.

\[
\begin{array}{c}
X \\
Y \\
\vdots \\
Z \\
\end{array}
\]
Assumptions

- Some rules of inference require making an Assumption which must eventually be Discharged (I, 67).
- When an assumption has been made and discharged in the course of a derivation, the segment of the derivation is called a Subderivation (I, 65).
- The derivation within which a subderivation occurs is called an Outer Derivation of the subderivation.

\[
\begin{array}{c}
  X \\
  Y \\
  ~ \vdash \\
  \vdash \\
  Z \\
  W
\end{array}
\]

Negation Introduction

- The rule of Negation Introduction requires an assumption of a sentence \( X \) and a derivation of a contradiction \( Y \) and \( \sim Y \) from it.
- The assumption can then be discharged and the negation of \( X \) written.

\[
\begin{array}{c}
  X \\
  \vdash \\
  \vdash \\
  \vdash \\
  Y \\
  \sim Y \\
  \sim X
\end{array}
\]

- One of \( Y \) and \( \sim Y \) is false, so given that they both follow by truth-preserving rules from \( X \) (and other premises or assumptions), \( X \) itself must be false and \( \sim X \) true (I, 71).

Negation Elimination

- The rule of Negation Elimination requires an assumption of a sentence \( \sim X \) and a derivation of a contradiction \( Y \) and \( \sim Y \) from it.
The assumption can then be discharged and X written.

\[
\begin{array}{c}
\sim X
\
\vdots
\
Y
\
\sim Y
\end{array}
\]

One of Y and \(\sim Y\) is false, so given they both follow by truth-preserving rules from \(\sim X\) (and other premises or assumptions), X itself must be true.

**Conditional Introduction**

- The rule of Conditional Introduction requires an assumption of a sentence X and a derivation of a sentence Y from it.
- The assumption can then be discharged and the conditional \(X \supset Y\) written.

\[
\begin{array}{c}
X
\
\vdots
\
Y
\end{array}
\]

- Since Y follows from truth-preserving rules from X (and other premises or assumptions), there is no way for X to be true and Y false. (I, 67).

**Biconditional Introduction**

- The rule of Biconditional Introduction requires an assumption of a sentence X and a derivation of a sentence Y from it, and then the assumption of Y and the derivation of X from it, with both assumptions discharged.
**Falsum Introduction**

- The rules for the *falsum* sentence letter are not given in Teller's text, since he does not use the symbol \( \bot \) in the syntax of sentence logic.
- The introduction rule allows that \( \bot \) may be written down any time that a sentence and its negation occur to the immediate right of a given scope line.

\[
\begin{array}{c}
\hline
X \\
\vdots \\
Y \\
\vdots \\
Y \\
\vdots \\
X \\
X \equiv Y
\end{array}
\]

- Since there is no possible case in which both \( X \) and \( \sim X \) are true, there is no possible case in which \( X \) and \( \sim B \) are true and \( \bot \) is false.

**Falsum Elimination**

- The elimination rule allows the introduction of any sentence to the immediate right of a scope line where \( \bot \) appears.

\[
\begin{array}{c}
\vdots \\
\bot \\
\vdots \\
X
\end{array}
\]
• Since there is no possible case in which \( \bot \) is true, there is no possible case in which both \( \bot \) is true and \( X \) is false.

• Perhaps it would be more accurate to say that the rules avoid unwanted falsehood, and hence are “safe,” than to say that they preserve truth.

Derivations

• “A Derivation is a list of which each member is either a sentence or another derivation. . . . Each sentence in a derivation is a premise or assumption, or a reiteration of a previous sentence from the same derivation or an outer derivation, or a sentence which follows from one of the rules of inference from previous sentences or subderivations of the derivation” (I, 88).

• The derivability of \( Y \) from a set of sentences \( X \) is symbolized as \( X \vdash Y \).

• That \( Y \) is not derivable from \( X \) is symbolized as \( X \nvdash Y \).

Soundness and Completeness

• The rules of inference used in this course are both Sound and Complete (I, 72).

• A set of rules is sound if and only if it is not possible using them to derive a false conclusion from a set of true premises.

• A set of rules is complete if and only if there is a derivation using them for every deductively valid argument.

• Proving soundness and completeness requires techniques that cannot be developed in this course.