2 Classical Sentential Logic

1. Introduction

In the present chapter, we briefly review the salient features of the most basic branch of modern symbolic logic – sentential logic (also called propositional logic).

2. Sentences and Connectives

Sentential logic (SL) analyzes sentences in terms of just two grammatical categories:

sentences
connectives

Whereas sentences are a primitive category, connectives are a derivative category, the basic definition being given as follows.

(d1) A ***connective*** is an expression with blanks such that, filling these blanks with sentences results in a sentence.

Every connective has a *degree*, which is a natural number (0, 1, 2, ...); in particular, when the degree of a connective is k, we say that it is an k-place connective. Officially:

(d2) Where k is any natural number (0, 1, 2, ...), A ***k-place connective*** is a connective with k-many blanks (places).[1]

Connectives are a species of the genus *functor*, which is generally defined as follows.

(d3) A ***functor*** is an expression with blanks such that, filling these blanks with expressions of specified categories results in an expression of a specified category.

Once we have the general idea of a functor, we can categorize them according to their syntactic behavior. For example, a one-place connective takes one sentence and produces a sentence, which can be represented as follows.

$\mathsf{S} \to \mathsf{S}$

Similarly, a two-place connective takes two sentences and produces a sentence, which can be depicted as follows.

$(\mathsf{S} \times \mathsf{S}) \to \mathsf{S}$

This notation can also be simplified as follows.

$\mathsf{S}^2 \to \mathsf{S}$

More generally, a k-place connective is depicted as follows.[2]

$\mathsf{S}^k \to \mathsf{S}$

---

[1] 0-place connectives are a technical nicety that do not occur naturally. The most prominent example of such a connective is the contradiction symbol '※' used primarily in derivations. Note, however that '※' has other theoretical uses; for example, one can prove that all truth-functional connectives are definable in terms of '→' and '※'.

[2] These are all the *simple functors* available in sentential logic. However, from an abstract categorial point of view, there are non-simple functors as well, i.e., functors that take functors as input. For example, a functor of category $[(\mathsf{S} \to \mathsf{S}) \to \mathsf{S}]$ takes a one-place connective as input and generates a sentence as output. The present author is unable to think of an example from ordinary language of such a functor. Nevertheless, the category is there in case we need it! For a general presentation of these ideas, please refer to the appendix on categorial grammar.

3. Declarative Sentences and Truth-Values

Ordinary sentential logic is not concerned with the whole class of sentences, but only *declarative sentences*, thus ignoring interrogative, imperative, exclamatory, and performative sentences. The simplest definition of a declarative sentence is that it is a sentence that is capable of being true or false. Basically, a declarative sentence is intended, when uttered, to declare something,[3] which in turn is either true or false.

Associated with the adjectives 'true' and 'false' are the abstract proper nouns 'True' and 'False', which refer to what are known as *truth-values*.

An analogy might be useful here. Consider the difference between the adjective 'blue' and the proper noun 'Blue', as used in the following two sentences.[4]

my favorite shirt is blue
my favorite color is Blue

We can readily hear the difference between these two sentences as soon as we invert them, as follows.

blue is my favorite shirt
Blue is my favorite color

The first one sounds funny (poetic, if you like); the second one sounds rather ordinary (prosaic, if you like).

Notice also that there is a correspondence between the adjective 'blue' and the noun 'Blue', given as follows.

object x is blue
if and only if
the color of object x $\underline{\underline{\text{is}}}$ Blue [or: Blue is the color of object x][5]

Now back to truth-values. There is an analogous relation in logic between the adjectives 'true' and 'false', on the one hand, and the abstract proper nouns 'True' and 'False'.

sentence $\mathbb{S}$ is true/false
if and only if
the truth-value of $\mathbb{S}$ $\underline{\underline{\text{is}}}$ True/False

---

[3] The "something" that a sentence declares or expresses is usually referred to as a *proposition*.
[4] Notice that we capitalize the proper noun to distinguish it from the corresponding adjective.
[5] Note carefully the distinction between the 'is' of predication and the 'is' of identity. This sentence is symbolized in identity logic as follows. $\forall x\{\mathrm{B}x \leftrightarrow [c(x)=\mathrm{B}]\}$

4. Truth-Functional Connectives

Ordinary SL does not deal with sentences in general, but only with declarative sentences. Furthermore, ordinary SL does not deal with all sentence connectives, but only with a special sub-class, known as *truth-functional* connectives. The following is the general definition.

(d4) A connective is **truth-functional** if and only if the truth-value of any compound formula constructed using that connective is a function of the truth-values of the constituent parts.

To be somewhat more specific, the following is the definition of truth-functionality as it applies to two-place connectives.

(d5) A two-place connective © is **truth-functional** if and only if the following obtains:
for any sentences S_1 and S_2, the truth-value of (S_1©S_2)
is a function of the respective truth-values of S_1 and S_2.

Now, the function referred to in this definition is called the *truth-function associated with* connective ©. Note carefully that we use the very same symbol both for the connective and for its associated truth-function; see below for examples.

Traditionally, there are five truth-functional connectives studied by SL.

negation	~	not…
conjunction	&	…and…
disjunction	∨	…or…
conditional	→	if…then…
biconditional	↔	…if and only if…

Whereas negation is a one-place connective that is written in prefix notation, the remaining connectives are two-place connectives, and are written in infix notation.

These are not all the truth-functional connectives; indeed, there are infinitely many of these. On the other hand, all truth-functional connectives are definable in terms of the above connectives; in fact, all truth-functional connectives can be defined in terms of just '~' and '&'.

Being truth-functional, each of the above connectives has associated with it a truth-function, which we denote by the very same symbol. These functions are depicted in the following tables [where the truth-values are abbreviated in the customary manner by 'T' and 'F'].

~T = F ~F = T	T&T = T T&F = F F&T = F F&F = F	T∨T = T T∨F = T F∨T = T F∨F = F	T→T = T T→F = F F→T = T F→F = T	T↔T = T T↔F = F F↔T = F F↔F = T

5. Sentence Forms and Argument Forms

Logic is concerned with evaluating argument validity. Following Aristotle, logicians analyze argument validity in terms of *form*. However, form is not an absolute notion; rather, each logical system offers its own analysis of form.

In the case of sentential logic, the analysis of form is rather simple. First, formulas (i.e., sentence forms) in SL are defined (inductively) as follows.

(f1) Every upper case (Roman) letter (with or without a subscript) is a formula of SL.

(f2) If $\mathcal{A}$ is a formula of SL, then so is: $\sim\mathcal{A}$.

(f3) If $\mathcal{A}$ and $\mathcal{B}$ are formulas of SL, then so are:
- (a) $(\mathcal{A} \,\&\, \mathcal{B})$
- (b) $(\mathcal{A} \vee \mathcal{B})$
- (c) $(\mathcal{A} \rightarrow \mathcal{B})$
- (d) $(\mathcal{A} \leftrightarrow \mathcal{B})$

(f4) Nothing else is a formula of SL.

In addition to sentence forms (formulas), we also have argument forms, which are defined generically as follows.

(d6) An ***argument form*** is a non-empty finite sequence of formulas, the last of which is designated as the *conclusion*, and the remainder of which (if there are any) are designated as the *premises*.

The notation we use to denote argument forms is given as follows.

$$(P_1 \,;\, P_2 \,;\, \ldots \,;\, P_n \,/\, C)$$

Note: when an argument expression stands alone, the outer parentheses are dropped – which yields:

$$P_1 \,;\, P_2 \,;\, \ldots \,;\, P_n \,/\, C$$

Every argument form must have a conclusion, but it need not have any premises. In that special case, it is written thus.

$$/\, C$$

A zero-premise argument, if valid, corresponds to a formula that is logically true – a tautology.

6. Admissible Valuations and Validity

Once we have the notion of argument form, we can attend to the question whether an argument form is valid or not. Validity is generically defined as follows.

(d7) An argument form is ***valid*** if and only if one cannot "make" its premises true without also "making" its conclusion true.

The concept "making true" is fleshed out in terms of valuation-functions, or simply valuations, which are defined as follows.

(d8) Let $\mathcal{L}$ be a formal language, Then a ***valuation*** on $\mathcal{L}$ is, by definition, any function that assigns a truth-value (T or F) to each formula of $\mathcal{L}$.

This only provides the category. Out of all the abstractly possible valuations, a given logical system will specify a subset of *admissible valuations*. This varies from logical system to logical system. In the case of ordinary SL, admissible valuations are defined as follows.

(d9) Let $\mathcal{L}$ be the formal language of SL (as specified above). Let υ be a valuation on $\mathcal{L}$. Then υ is said to be an ***admissible valuation*** (w.r.t. ordinary SL) if and only if it satisfies the following restrictions.

$$\begin{array}{lcl}
\upsilon({\sim}\mathcal{A}) & = & {\sim}\upsilon(\mathcal{A}) \\
\upsilon(\mathcal{A}\&\mathcal{B}) & = & \upsilon(\mathcal{A})\ \&\ \upsilon(\mathcal{B}) \\
\upsilon(\mathcal{A}\vee\mathcal{B}) & = & \upsilon(\mathcal{A}) \vee \upsilon(\mathcal{B}) \\
\upsilon(\mathcal{A}\rightarrow\mathcal{B}) & = & \upsilon(\mathcal{A}) \rightarrow \upsilon(\mathcal{B}) \\
\upsilon(\mathcal{A}\leftrightarrow\mathcal{B}) & = & \upsilon(\mathcal{A}) \leftrightarrow \upsilon(\mathcal{B})
\end{array}$$

Recall that we use every connective symbol ambiguously, to refer both to the connective, and to its associated truth-function. In the identities above, the logic-symbols on the left refer to connectives, whereas the logic symbols on the right refer to truth-functions. The following is an example of applying these rules.

Suppose υ(P)=T, and υ(Q)=F. Then in order for υ to be admissible, it must satisfy the following.

$$\begin{array}{lclclcl}
\upsilon({\sim}\mathrm{P}) & = & {\sim}\upsilon(\mathrm{P}) & = & {\sim}\mathrm{T} & = & \mathrm{F} \\
\upsilon(\mathrm{P}\&\mathrm{Q}) & = & \upsilon(\mathrm{P})\ \&\ \upsilon(\mathrm{Q}) & = & \mathrm{T}\&\mathrm{F} & = & \mathrm{F} \\
\upsilon(\mathrm{P}\vee\mathrm{Q}) & = & \upsilon(\mathrm{P}) \vee \upsilon(\mathrm{Q}) & = & \mathrm{T}\vee\mathrm{F} & = & \mathrm{T} \\
\upsilon(\mathrm{P}\rightarrow\mathrm{Q}) & = & \upsilon(\mathrm{P}) \rightarrow \upsilon(\mathrm{Q}) & = & \mathrm{T}\rightarrow\mathrm{F} & = & \mathrm{F} \\
\upsilon(\mathrm{P}\leftrightarrow\mathrm{Q}) & = & \upsilon(\mathrm{P}) \leftrightarrow \upsilon(\mathrm{Q}) & = & \mathrm{T}\leftrightarrow\mathrm{F} & = & \mathrm{F}
\end{array}$$

Now, once we have the notion of admissible valuation – however it is implemented – we can define validity as follows.

(d10) Let $\mathbb{L}$ be a logical system, and let V($\mathbb{L}$) be the associated class of admissible valuations of $\mathbb{L}$. Let $(P_1\,;P_2\,;\ldots;P_n\,/\,C)$ be an argument form of the associated language of $\mathbb{L}$. Then $(P_1\,;P_2\,;\ldots;P_n\,/\,C)$ is said to be ***valid*** in $\mathbb{L}$ if and only if:

for every υ in V($\mathbb{L}$):
if $\upsilon(P_1)$=T, and $\upsilon(P_2)$=T, and ... , and $\upsilon(P_n)$=T, then $\upsilon(C)$=T.

The following is an immediate corollary about invalidity.

(c) Let $\mathbb{L}$ be a logical system, and let $V(\mathbb{L})$ be the associated class of admissible valuations of $\mathbb{L}$. Let $(P_1\ ;\ P_2\ ;\ ...\ ;\ P_n\ /\ C)$ be an argument form of the associated language of $\mathbb{L}$. Then $(P_1\ ;\ P_2\ ;\ ...\ ;\ P_n\ /\ C)$ is **invalid** in $\mathbb{L}$ if and only if:

there is a υ in $V(\mathbb{L})$ such that:
$\upsilon(P_1)$=T, and $\upsilon(P_2)$=T, and ... , and $\upsilon(P_n)$=T, and $\upsilon(C)$=F.

In other words, an argument is invalid if and only if we can "make" its premises true while "making" its conclusion false. This corollary follows from the definition of validity together with the definition of valuation. The following is a very simple example of an argument form that is valid in SL.

$$\mathrm{P} \to \mathrm{Q}\ ;\ {\sim}\mathrm{Q}\ /\ {\sim}\mathrm{P}$$

Proof: To see that it is valid *by our definition*, we reason as follows. Suppose the above argument form is *not* SL-valid. Then, by (C) above, there is an SL-admissible valuation υ such that: $\upsilon(\mathrm{P}{\to}\mathrm{Q})$=T, $\upsilon({\sim}\mathrm{Q})$=T, and $\upsilon({\sim}\mathrm{P})$=F. Also, since υ is SL-admissible, $\upsilon({\sim}\mathrm{P}) = {\sim}\upsilon(\mathrm{P})$, so $\mathrm{F} = {\sim}\upsilon(\mathrm{P})$, so ${\sim}\mathrm{F} = {\sim}{\sim}\upsilon(\mathrm{P})$, so $\upsilon(\mathrm{P})$=T. By similar reasoning, $\upsilon(\mathrm{Q})$=F. Also, since υ is an SL-admissible valuation, $\upsilon(\mathrm{P}{\to}\mathrm{Q}) = \upsilon(\mathrm{P}){\to}\upsilon(\mathrm{Q}) = \mathrm{T}{\to}\mathrm{F} = \mathrm{F}$. This contradicts our hypothesis that $\upsilon(\mathrm{P}{\to}\mathrm{Q})$=T.

7. Inference Rules for Sentential Logic

The reasoning at the end of the last section can be mechanized/automated in a number of ways. One way is the method of truth-tables, which we will not review here. The other way is the method of formal deduction (derivation).

A formal deduction system is a means of reproducing the class of valid arguments. Such a system counts as "natural" if the rules reflect one's natural reasoning about admissible valuations (truth tables, if you like). There are many such systems – some more "natural" than others. We propose one such system of natural deduction for SL, which is an amplification of the system presented in the author's *Symbolic Logic: A First Course*.

This system has two sorts of rules.

inference rules
strategic rules

Generally, inference-rules are easy to understand: they are simply argument forms that we take for granted. In the proposed system, they are rules that intuitively capture the semantic workings of the connectives, as summarized by the admissible valuations.

First, for each two-place connective, we provide four rules – an introduction-rule, an elimination-rule, a negation-introduction-rule, and a negation-elimination-rule. This results in a total of 16 rules, given as follows.

&I	**&O**		**~&I/O**
$\begin{array}{c}\mathcal{A}\\ \mathcal{B}\\ \hline \mathcal{A}\&\mathcal{B}\end{array}$	$\begin{array}{c}\mathcal{A}\&\mathcal{B}\\ \hline \mathcal{A}\end{array}$	$\begin{array}{c}\mathcal{A}\&\mathcal{B}\\ \hline \mathcal{B}\end{array}$	$\begin{array}{c}{\sim}(\mathcal{A}\&\mathcal{B})\\ \hline\hline \mathcal{A} \to {\sim}\mathcal{B}\end{array}$

vI		**vO**		**~vI/O**
$\begin{array}{c}\mathcal{A} \\ \hline \mathcal{A}\vee\mathcal{B}\end{array}$	$\begin{array}{c}\mathcal{B} \\ \hline \mathcal{A}\vee\mathcal{B}\end{array}$	$\begin{array}{l}\mathcal{A}\vee\mathcal{B} \\ \sim\mathcal{A} \\ \hline \mathcal{B}\end{array}$	$\begin{array}{l}\mathcal{A}\vee\mathcal{B} \\ \sim\mathcal{B} \\ \hline \mathcal{A}\end{array}$	$\begin{array}{c}\sim(\mathcal{A}\vee\mathcal{B}) \\ \hline\hline \sim\mathcal{A} \;\&\; \sim\mathcal{B}\end{array}$

→I		**→O**		**~→I/O**
$\begin{array}{c}\sim\mathcal{A} \\ \hline \mathcal{A}\rightarrow\mathcal{B}\end{array}$	$\begin{array}{c}\mathcal{B} \\ \hline \mathcal{A}\rightarrow\mathcal{B}\end{array}$	$\begin{array}{l}\mathcal{A}\rightarrow\mathcal{B} \\ \mathcal{A} \\ \hline \mathcal{B}\end{array}$	$\begin{array}{l}\mathcal{A}\rightarrow\mathcal{B} \\ \sim\mathcal{B} \\ \hline \sim\mathcal{A}\end{array}$	$\begin{array}{c}\sim(\mathcal{A}\rightarrow\mathcal{B}) \\ \hline\hline \mathcal{A} \;\&\; \sim\mathcal{B}\end{array}$

↔I	**↔O**		**~↔I/O**
$\begin{array}{l}\mathcal{A}\rightarrow\mathcal{B} \\ \mathcal{B}\rightarrow\mathcal{A} \\ \hline \mathcal{A}\leftrightarrow\mathcal{B}\end{array}$	$\begin{array}{c}\mathcal{A}\leftrightarrow\mathcal{B} \\ \hline \mathcal{A}\rightarrow\mathcal{B}\end{array}$	$\begin{array}{c}\mathcal{A}\leftrightarrow\mathcal{B} \\ \hline \mathcal{B}\rightarrow\mathcal{A}\end{array}$	$\begin{array}{c}\sim(\mathcal{A}\leftrightarrow\mathcal{B}) \\ \hline\hline \sim\mathcal{A}\leftrightarrow\mathcal{B}\end{array}$

Notice that the negation rules are collapsed into bi-directional rules, indicated by the long equals sign ‘══’.

In addition to these rules, which pertain to the two-place connectives, we have the following bi-directional rule, **double negation**, which pertains to the lone one-place connective.

DN

$$\begin{array}{c}\sim\sim\mathcal{A} \\ \hline\hline \mathcal{A}\end{array}$$

At this point, it is important to realize that all the inference rules of this particular deductive system – including DN – apply to whole lines, and not to subformulas. For example, the following is most definitely *not* an instance of &O.

&O???

$$\begin{array}{l}(P \;\&\; Q) \rightarrow R \\ \hline P \rightarrow R\end{array}$$

This is not even valid! But, by similar syntactic fastidiousness, the following is *not* an instance of DN.

DN???

$$\begin{array}{l}P\rightarrow Q \\ \hline\hline P\rightarrow\sim\sim Q\end{array}$$

Yes, it is valid, but the point is that the rules pertain to lines of particular forms, and these forms are determined by their principal connectives (or, in the case of negation rules, sub-principal connectives).

Before discussing the strategic rules, we add one further technical nicety. Specifically, for derivational purposes only, there is a special zero-place connective '※', which is semantically defined always to be false,[6] and which is derivationally implemented as follows.

※I	※O
$\mathcal{A}$	※
$\sim\mathcal{A}$	——
——	$\mathcal{A}$
※	

The specific role played by '※' will become evident in the next few sections.

8. Strategic-Rules – The Show-Scheme

In addition to inference rules, we also need *strategic-rules* for demonstrating (showing) different kinds of formulas. In this connection, we opt for a show-scheme that traces to Kalish and Montague.[7]

In selecting a set of strategic-rules, rather than pursue a frugal course, that uses just a few strategic rules, we opt for a *full* set of strategic rules – so that there is a strategic (show) rule for every connective. These are given as follows.

Conditional Derivation (CD)		**Tilde Indirect Derivation (~ID)**	
~~SHOW~~: $\mathcal{A}\rightarrow\mathcal{B}$	ID	~~SHOW~~: $\sim\mathcal{A}$	ID
$\mathcal{A}$	As	$\mathcal{A}$	As
~~SHOW~~: $\mathcal{B}$		~~SHOW~~: ※	

Ampersand Derivation (&D)		**Biconditional Derivation (↔D)**	
~~SHOW~~: $\mathcal{A}\&\mathcal{B}$	&D	~~SHOW~~: $\mathcal{A}\leftrightarrow\mathcal{B}$	↔D
~~SHOW~~: $\mathcal{A}$		~~SHOW~~: $\mathcal{A}\rightarrow\mathcal{B}$	
~~SHOW~~: $\mathcal{B}$		~~SHOW~~: $\mathcal{B}\rightarrow\mathcal{A}$	

Wedge Indirect Derivation (∨ID)	
~~SHOW~~: $\mathcal{D}_1 \vee \mathcal{D}_2 \vee \ldots \vee \mathcal{D}_k$	∨ID
$\sim\mathcal{D}_1$	As
$\sim\mathcal{D}_2$	As
…	
$\sim\mathcal{D}_k$	
~~SHOW~~: ※	

[6] Technically, for any admissible valuation υ, υ(※)=F.

[7] *Logic: Techniques of Formal Reasoning*.

9. Generic Strategic Rules

The rules above are *specific* strategic rules; each one pertains to a specific SL connective. In addition to these, there are two generic rules, called direct derivation (DD) and indirect derivation (ID), given as follows.

Direct Derivation (DD)	**(Generic) Indirect Derivation (ID)**
~~SHOW~~: $\mathcal{A}$ DD $\mathcal{A}$	~~SHOW~~: $\mathcal{A}$ ID $\sim\mathcal{A}$ As ~~SHOW~~: ※

10. An Additional Natural SL Strategy – Separation of Cases

The student familiar with intro logic is aware that one often has to deduce conclusions when one has disjunctive premises. The technique I suggest in my intro book is called the wedge-out strategy. This works, but it does not correspond to the way that most practicing logicians and mathematicians in fact attack "real" deductive problems involving disjunctions.

The technique that is almost always used is called Separation of Cases, which is schematically described as follows. Here, ‘c_1’ means ‘case 1’, ‘c_2’ means ‘case 2’, etc.

Separation of Cases (SC)	
$\mathcal{D}_1 \vee \mathcal{D}_2 \vee ... \vee \mathcal{D}_k$	
~~SHOW~~: C	SC
c1: $\mathcal{D}_1$	As
~~SHOW~~: C	
c2: $\mathcal{D}_2$	As
~~SHOW~~: C	
.	
.	
ck: $\mathcal{D}_k$	As
~~SHOW~~: C	

In other words, if one is trying to show C, and one has a disjunction with k-many disjuncts, then one shows C in each of the k-many cases. One shows C in case (if) $\mathcal{D}_1$; one shows C in case $\mathcal{D}_2$; etc.

Note carefully the box-scheme. For example, as soon as one shows C in case 1, one boxes and cancels C, and one boxes the case 1 sub-derivation as well. It is probably helpful to think of a separation of case argument as a series of conditional derivations, of the form

~~SHOW~~: $\mathcal{D}_1 \to C$
~~SHOW~~: $\mathcal{D}_2 \to C$
.
.
~~SHOW~~: $\mathcal{D}_k \to C$,

in which the conditional show-lines are omitted for the sake of conciseness. Then it is clear where the extra boxes come from.

11. An Example of a Derivation

According to the show-scheme, as the derivation constructor thinks of goals, he/she writes down these goals as lines – specifically, show-lines. This means that the order of lines in a show-style derivation reflects the *strategic order* of a derivation, not its *inferential order*. Its inferential order is more convoluted.

For example, consider the following example.

P → (Q → R) ; P → ~R / P → ~Q

One begins by writing down the premises, as well as the obvious goal – namely, to show the conclusion.

(1)	P → (Q → R)	Pr
(2)	P → ~R	Pr
(3)	SHOW: P → ~Q	

All derivations start in this manner – writing down all the premises, and writing down the conclusion as a show-line.

In this particular example, the show-line involves a conditional, so we appeal to the strategic rule for conditionals, which is →D (a.k.a. conditional derivation – CD). That means we write down two more lines as follows.

(1)	P → (Q → R)	Pr
(2)	P → ~R	Pr
(3)	SHOW: P → ~Q	CD
(4)	P	As
(5)	SHOW: ~Q	

In other words, in order to show a conditional, one assumes its antecedent, and one shows its consequent.

At this point in the derivation, we have set our (intermediate) goal to be showing '~Q'. This is a negation, so we appeal to the strategic-rule for negations, which is ~D (a.k.a. indirect derivation – ID). This means we write down two more lines as follows.

(1)	P → (Q → R)	Pr
(2)	P → ~R	Pr
(3)	SHOW: P → ~Q	CD
(4)	P	As
(5)	SHOW: ~Q	ID
(6)	Q	As
(7)	SHOW: ※	

In other words, in order to show a negation $\sim\Phi$, one assumes the dis-negated formula Φ, and one shows a contradiction, which we formally represent by '※'.

Now, there is no *specific* show-※ strategy, so we must use a generic strategy – either DD or ID. ID is useless here, so we use DD. That means there is nothing more to write down by way of strategy. Instead, we must now start using the inference-rules ($\to$O, ※I, etc.) to deduce (directly) '※'.

At this point it is crucial to recognize the difference between lines we can, and lines we cannot, use. Basically, some lines count as assets, and others count as promises to be paid off using assets. In particular, any uncancelled show-line counts as a promise, not an asset. So, at the present point in the above derivation, lines 1,2,3,6 are available, but lines 3,5,7 are *not* available.

If we apply inference rules to lines that are available, then we can produce the following.

(1)	$P \to (Q \to R)$	Pr
(2)	$P \to \sim R$	Pr
(3)	SHOW: $P \to \sim Q$	CD
(4)	P	As
(5)	SHOW: $\sim Q$	ID
(6)	Q	As
(7)	SHOW: ※	DD
(8)	$Q \to R$	1,4,$\to$O
(9)	R	6,8,$\to$O
(10)	$\sim R$	2,4,$\to$O
(11)	※	9,10,※I

At this point, we have gone down the page as far as we are going, but we are not through yet! We now proceed up the page, cancelling our various promises. First, having obtained ※, we can cancel "SHOW:※", thus obtaining the following.

(1)	$P \to (Q \to R)$	Pr
(2)	$P \to \sim R$	Pr
(3)	SHOW: $P \to \sim Q$	CD
(4)	P	As
(5)	SHOW: $\sim Q$	ID
(6)	Q	As
(7)	~~SHOW~~: ※	DD
(8)	$Q \to R$	1,4,$\to$O
(9)	R	6,8,$\to$O
(10)	$\sim R$	2,4,$\to$O
(11)	※	9,10,※I

Line 7 is cancelled in accordance with the DD show-rule. Now, line 7 is available, but lines 8-11 are not available – they have been paid to buy ※. But now we have deduced ※ assuming Q, so by ID, we are entitled to cancel "SHOW: $\sim$Q", which yields the following.

(1)	$P \to (Q \to R)$	Pr
(2)	$P \to \sim R$	Pr
(3)	SHOW: $P \to \sim Q$	CD
(4)	P	As
(5)	~~SHOW~~: $\sim Q$	ID
(6)	│ Q	As
(7)	│ ~~SHOW~~: ※	DD
(8)	│ │ $Q \to R$	1,4,$\to$O
(9)	│ │ R	6,8,$\to$O
(10)	│ │ $\sim R$	2,4,$\to$O
(11)	│ │ ※	9,10,※I

Next, we see that ~Q has be obtained assuming P, so in accordance with the CD show-rule, we can cancel "SHOW: P→~Q", which yields the following and final position.

(1)	$P \to (Q \to R)$	Pr
(2)	$P \to \sim R$	Pr
(3)	~~SHOW~~: $P \to \sim Q$	CD
(4)	│ P	As
(5)	│ ~~SHOW~~: $\sim Q$	ID
(6)	│ Q	As
(7)	│ ~~SHOW~~: ※	DD
(8)	│ │ $Q \to R$	1,4,$\to$O
(9)	│ │ R	6,8,$\to$O
(10)	│ │ $\sim R$	2,4,$\to$O
(11)	│ │ ※	9,10,※I

At this point, we have paid off our original promise – to show the conclusion – so we are done.

12. More Examples of Derivations in SL

#1: $P \to (Q \to R)$ / $(P \to Q) \to (P \to R)$

(1)	$P \to (Q \to R)$	Pr
(2)	~~SHOW~~: $(P \to Q) \to (P \to R)$	CD
(3)	│ $P \to Q$	As
(4)	│ ~~SHOW~~: $P \to R$	CD
(5)	│ │ P	As
(6)	│ │ ~~SHOW~~: R	DD
(7)	│ │ $Q \to R$	1,5,$\to$O
(8)	│ │ Q	3,5,$\to$O
(9)	│ │ R	7,8,$\to$O

#2: $(P \& Q) \to R$ / $(P \& \sim R) \to \sim Q$

(1)	$(P \& Q) \to R$	Pr
(2)	~~SHOW~~: $(P \& \sim R) \to \sim Q$	CD
(3)	│ $P \& \sim R$	As
(4)	│ ~~SHOW~~: $\sim Q$	ID
(5)	│ │ Q	As
(6)	│ │ ~~SHOW~~: ※	DD
(7)	│ │ │ P	3,&O
(8)	│ │ │ $\sim R$	3,&O
(9)	│ │ │ $P \& Q$	5,6,&I
(10)	│ │ │ R	1,9,$\to$O
(11)	│ │ │ ※	8,9,※I

#3: (P ∨ Q) → (P & Q) / (P & Q) ∨ (~P & ~Q)

(1)	(P ∨ Q) → (P & Q)	Pr
(2)	~~SHOW~~: (P & Q) ∨ (~P & ~Q)	∨D
(3)	~(P & Q)	As
(4)	~(~P & ~Q)	As
(5)	~~SHOW~~: ※	DD
(6)	~(P ∨ Q)	1,3,→O
(7)	~P	6,~∨O
(8)	~Q	6,~∨O
(9)	~P → ~~Q	4,~&O
(10)	~~Q	7,9,→O
(11)	※	8,10,※I

#4: / [(P → Q) → P] → P

(1)	~~SHOW~~: [(P → Q) → P]→P	CD
(2)	(P → Q) → P	As
(3)	~~SHOW~~: P	ID
(4)	~P	As
(5)	~~SHOW~~: ※	DD
(6)	~(P →Q)	2,4,→O
(7)	P & ~Q	6,~→O
(8)	P	7,&O
(9)	※	4,8,※I

13. Counter-Models in SL

The method of formal derivation can demonstrate validity, but it cannot *by itself* demonstrate *in*validity. In order to show invalidity, we return to our original definition of validity, and its corollary theorem about invalidity

(d10) Let $\mathbb{L}$ be a logical system, and let V($\mathbb{L}$) be the associated class of admissible valuations of $\mathbb{L}$. Let $\mathbb{A} = (P_1 ; P_2 ; ... ; P_n / C)$ be an argument form of the associated language of $\mathbb{L}$. Then $\mathbb{A}$ is said to be valid in $\mathbb{L}$ if and only if:

for every υ in V($\mathbb{L}$): if $\upsilon(P_1)$=T, and $\upsilon(P_2)$=T, and ... ,and $\upsilon(P_n)$=T, then $\upsilon(C)$=T.

(C) Let $\mathbb{L}$ be a logical system, and let V($\mathbb{L}$) be the associated class of admissible valuations of $\mathbb{L}$. Let $\mathbb{A} = (P_1 ; P_2 ; ... ; P_n / C)$ be an argument form of the associated language of $\mathbb{L}$. Then $\mathbb{A}$ is ***invalid*** in $\mathbb{L}$ if and only if:

there is a υ in V($\mathbb{L}$) such that $\upsilon(P_1)$=T, and $\upsilon(P_2)$=T and, ... and, $\upsilon(P_n)$=T, and $\upsilon(C)$=F.

In other words, an argument is invalid if and only if we can "make" its premises true while "making" its conclusion false.

The process by which we do this is called the *method of counter-models*. Anyone who has done truth-tables is already familiar with this technique as it applies to SL. The following is a simple example, done in strict accordance with the formal definition and its corollary.

P→Q ; ~P / ~Q

In order to show that this argument form is invalid in SL, it is sufficient to produce an SL-admissible valuation υ such that υ(P→Q)=υ(~P)=T, and υ(~Q)=F. There are infinitely-many, but as it turns out, they all have the following in common: υ(P)=F, υ(Q)=T. For then υ(P→Q) = υ(P)→υ(Q) = F→T = T; and υ(~P) = ~υ(P) = ~F = T; and υ(~Q) = ~υ(Q) = ~T = F.

14. The Relation between Derivations and Counter-Models

We have a method of showing arguments to be valid, and we have a method of showing arguments to be invalid. They are in fact intimately related.

Let us re-examine the argument form at the end of the last section.

P→Q ; ~P / ~Q

What happens if we try to demonstrate this argument using the method of derivation? Well, it goes as follows.

(1)	P→Q	Pr
(2)	~P	Pr
(3)	SHOW: ~Q	ID
(4)	Q	As
(5)	SHOW: ※	??
(6)	??	

At this point, the derivation grinds to a halt. On the other hand, at this point, we can use our incomplete derivation to construct a counter-model. In particular, we have 3 available lines – 1, 2, 4 – each of which gives us information about the counter-model.

(1)	P→Q	T	Pr
(2)	~P	T	Pr
	P	F	from (2) by semantic rule for ~
(4)	Q	T	As
	~Q	F	from (4) by semantic rule for ~

This suggests that we assign F to P and T to Q, to produce our counter-model.

Let us do one more example.

~(P & Q) / ~P & ~Q

(1)	~(P & Q)	Pr
(2)	SHOW: ~P & ~Q	&D
(3)	SHOW: ~P	ID
(4)	P	As
(5)	SHOW: ※	???
(6)	P → ~Q	1,~&O
(7)	~Q	4,6,→O
(8)	???	

Once again, we are stuck, this time at line 8. Once again, however, we can use our incomplete derivation to produce an assignment of truth-values that provides a counter-model.

υ(P) = T	line 4	υ(Q) = F	line 7
υ{~(P & Q)}	=	υ(~P & ~Q)	=
~υ(P & Q)	=	υ(~P) & υ(~Q)	=
~{υ(P) & υ(Q)}	=	~υ(P) & ~υ(Q)	=
~{T & F}	=	~T & ~F	=
~F	=	F & T	=
T		F	