Chapter 1

Sentential Logic: Derivations

1.1 The Derivation System SD

The first derivation rule of *SD* is *Reiteration*

    Reiteration (R)

$$
\begin{array}{c|c}
 & \alpha \\
\rhd & \alpha
\end{array}
$$

The symbol '$\rhd$' indicates the formula we are allowed to derive using the rule. Reiteration does nothing more than simply allowing us to derive a sentence from itself. Here's an example of a a derivation that uses the Reiteration rule.

$$
\begin{array}{c|ll}
1 & P \supset (Q \vee R) & \\
2 & \neg(Q \mathbin{\&} S) & \\ \hline
3 & P \supset (Q \vee R) & \text{R } 1
\end{array}
$$

The first two lines of the derivation are assumptions. Assumptions are set off by a horizontal line from the derived sentences that follow. The vertical line is called a scope line. Notice that the lines of the proof are numbered. To the right of each derived formula will be the rule that was used to derive that formula, and the line numbers to which the rule was applied.

Along with Reiteration, *SD* includes rules for introducing and eliminating each of the five logical connectives in *SL*. The conjunction rules are the simplest, so let's start there.

    Conjunction Introduction (&I)

$$
\begin{array}{c|l} & \alpha \\ & \beta \\ \rhd & \alpha\ \&\ \beta \end{array} \quad \text{or} \quad \begin{array}{c|l} & \alpha \\ & \beta \\ \rhd & \beta\ \&\ \alpha \end{array}
$$

This rule allows us to form a conjunction of any two formulas occurring on previous lines. The order that those formulas appear in the proof doesn't matter. Even if β occurred before α, we could still derive α & β.

Conjunction Elimination (&E)

$$
\begin{array}{c|l} & \alpha\ \&\ \beta \\ \rhd & \alpha \end{array} \quad \text{or} \quad \begin{array}{c|l} & \alpha\ \&\ \beta \\ \rhd & \beta \end{array}
$$

We have two different versions of the same rule—this allows us to derive either of the two conjuncts using the single conjunction elimination rule.

Let's see an example derivation using the conjunction rules.

Derive: W & (X & Y)

$$
\begin{array}{c|l} 1 & W\ \&\ X \\ 2 & Y\ \&\ Z \\ \hline & \\ & \end{array}
$$

A good strategy is to start with the sentence that is to be derived and then think backwards. To form 'W & (X & Y)' using the &I rule, we will need to have 'W' on one line and 'X & Y' on another. We can get 'W' from line 1, so let's start there.

$$
\begin{array}{c|l} 1 & W\ \&\ X \\ 2 & Y\ \&\ Z \\ \hline 3 & W \qquad \text{&E 1} \end{array}
$$

To form 'X & Y', we will need to have a line with each formula. One will come from line 1 and the other from line 2.

1	W & X	
2	Y & Z	
3	W	&E 1
4	X	&E 1
5	Y	&E 2

Now, we can from 'X & Y' using conjunction introduction on lines 4 and 5.

1	W & X	
2	Y & Z	
3	W	&E 1
4	X	&E 1
5	Y	&E 2
6	X & Y	&I 4, 5

Finally, we use lines 1 and 6 to form 'W & (X & Y)'

1	W & X	
2	Y & Z	
3	W	&E 1
4	X	&E 1
5	Y	&E 2
6	X & Y	&I 4, 5

Conditional Elimination ($\supset$ E)

	$\alpha \supset \beta$
	α
$\triangleright$	β

The conditional elimination rule, also called 'Modus Ponens', tells us that the consequent of a conditional can be derived whenever the conditional and its antecedent occur on previous lines. Here's a derivation using conditional elimination.

Derive: H

```
1  │ F      /H
2  │ G
3  │ (F & G) ⊃ H
   ├──────────
4  │ F & G           &I 1, 2
5  │ H               ⊃E 3, 4
```

Conditional Introduction (⊃ I)

```
        │ │ α
        │ ├──
        │ │ β
   ▷    │ α ⊃ β
```

Conditional introduction is often called 'conditional proof'. Notice that it uses a *subderivation*. The subderivation begins with α as an assumption, and then derives β. Since β can be derived from α, the conditional α ⊃ β must be true.

Look at this example of a derivation using conditional proof:

1. If Alan passes Organic Chemistry, then he will graduate.
2. If Alan graduates, then he will move to Kansas.
∴ If Alan passes Organic Chemistry, then he will move to Kansas.

```
1  │ P ⊃ G
2  │ G ⊃ K
   │   ┌──
3  │   │ P
   │   ├──
4  │   │ G      ⊃E 1, 3
5  │   │ K      ⊃E 2, 4
6  │ P ⊃ K      ⊃I 3–5
```

The subderivation is on lines 3–5. It has vertical and horizontal lines just like the derivations that we have been doing so far. The horizontal line distinguishes assumptions from derived formulas; the assumptions of the main derivation are

called primary assumptions, and the assumptions of any subderivations are called auxiliary assumptions. The vertical line is the scope line. The scope line shows the scope of any assumptions. The subderivation shows that, given lines 1 and 2, if we assume 'P', we can derive 'P ⊃ K'. We complete the subderivation by ending the scope line of the subderivation and moving the scope line that is immediately to the left of the subderivation. By ending the subderivation, we discharge the assumption. After the assumption is discharged, the lines in the subderivation are no longer accessible, and thus cannot be used later in the proof. For example, this is not allowed:

1	P ⊃ G	
2	G ⊃ K	
3	P	
4	G	⊃E 1, 3
5	K	⊃E 2, 4
6	K	MISTAKE! R 5

This claims that, given lines 1 and 2, we can derive 'K', but we cannot do that without making the further assumption of line 3. What we can do, though, is appeal to the entire subderivation, as in our original example:

1	P ⊃ G	
2	G ⊃ K	
3	P	
4	G	⊃E 1, 3
5	K	⊃E 2, 4
6	P ⊃ K	⊃I 3–5

Note that the justification for line 6 cites lines 3–5, which is the entire subderivation.

Disjunction Introduction (∨I)

	α		or		α
▷	α ∨ β			▷	β ∨ α

This is an unusual rule for two reasons. First, it starts with a definite claim, α, and derives a vague claim, α ∨ β. Second, β might be completely new to the derivation, giving us a feeling of deriving something from nothing. The rule is obviously truth-preserving, though, since α is true, it has to be the case that α ∨ β is true. Disjunction Introduction is particularly useful when a formula required to complete a derivation contains a sentence-letter that is not found in any of the premises. For example,

1	F & G	
2	(F ∨ H) ⊃ E	
3	F	&E 1
4	F ∨ H	∨I 3
5	E	⊃E 4, 3
▷	E ∨ I	∨I 5

Disjunction Elimination (∨E)

1	α ∨ β	
2		α
3		γ
4		β
5		γ
▷	γ	

We can derive γ from α ∨ β if we can both derive γ from α and derive γ from β. In other words, if we have sentence of the form α ∨ β, a subderivation that begins with the assumption α and ends with γ, and another subderivation that begins with the assumption β and ends with γ, we can derive the sentence γ.

Negation Introduction (¬I)

	α
	β
	¬β
¬α	

¬α can be derived by using a subderivation that assumes α and derives β and ¬β from α Both β and ¬β must be immediately to the right of same scope line as α. Intuitively, anything that entails a contradiction must be false.

Negation Elimination ¬E
Negation Elimination works the same way:

$$
\begin{array}{|l}
\quad\begin{array}{|l} \hline \neg\alpha \\[4pt] \beta \\[4pt] \neg\beta \end{array} \\[14pt]
\alpha
\end{array}
$$

Biconditional Introduction

$$
\begin{array}{ll}
1 & \quad\begin{array}{|l} \hline \alpha \\[4pt] \beta \end{array} \\[14pt]
2 & \\
3 & \quad\begin{array}{|l} \hline \beta \\[4pt] \alpha \end{array} \\[14pt]
4 & \\
\rhd & \alpha \equiv \beta
\end{array}
$$

This rule makes sense if we remember that a biconditional is logically equivalent to the conjunction of two conditionals, that is, α ≡ β is equivalent to (α ⊃ β) & (β ⊃ α).

Biconditional Elimination (≡ E)

$$
\begin{array}{l@{\qquad}c@{\qquad}l}
\begin{array}{|l} \alpha \equiv \beta \\[4pt] \alpha \\[4pt] \rhd\;\beta \end{array}
& \text{or} &
\begin{array}{|l} \alpha \equiv \beta \\[4pt] \beta \\[4pt] \rhd\;\alpha \end{array}
\end{array}
$$

Note that Biconditional Elimination works just like Conditional Elimination, except that either term can be derived from the biconditional and the other term.

1.2   Using the Rules of SD

Here are some important things to keep in mind when using *SD*. First, all of the rules of *SD* are rules of inference, and rules of inference apply only to complete lines of a derivation. For example, consider this simple derivation:

Derive: A

$$\begin{array}{|l} (A \ \& \ B) \ \& \ C \\ \hline \\ \end{array}$$

We are given a conjunction, and we know that all three of those conjuncts have to be true for that conjunction to be true. So, it is tempting to do this:

1	(A & B) & C	
2	A	&E 1

NO!

The rule of inference can only be applied to the entire line, not part of the line. So, we first have to use Conjunction Elimination to break down the first line, then use it again to finish the proof.

1	(A & B) & C	
2	A & B	&E 1
3	A	&E 2

Second, the only lines that can be used for the rules of *SD* are lines that are accessible at that stage of the proof. A formula is accessible at line *n* if and only if it is not within the scope of an assumption that has been discharged before line *n*.

Notice this mistake:

1	F ⊃ G	
2	G ⊃ H	
3	F	
4	G	⊃E 1, 3
5	H	⊃E 2, 4
6	F ⊃ H	⊃I 3–5
7	F	R 3
8	H	⊃E 6, 7

NO!

Lines 1–6 are correct. The subderivation assumes F, and derives G. It is important to remember that when a subderivation is used, the entire subderivation must be cited. Here, line 6 cites Conditional Introduction and lines 3–5. The mistake occurs in line 7. Line 7 reiterates line 3, but line 3 is not accessible at 7. Once the assumption of the subderivation is discharged, none of the lines within that subderivation can be used in the remaining part of the proof.

1.3 SD: Basic Concepts

SD is the basic derivation system for *SL*. Stated precisely, A derivation in *SD* is a series of sentences of *SL*, in which each sentence is either an assumption with a line indicating its scope or is justified by one of the rules of *SD*. Now that we have a basic understanding of the rules of *SD*, it's time to define a few basic concepts. The first is a derivation:

> A **derivation in SD** is a series of sentences of *SL*, in which each sentence is either an assumption with a line indicating its scope or is justified by one of the rules of *SD*.

We define the concept of derivability like this:

> A sentence α of *SL* is **derivable in SD** from a set Γ of sentences of *SL* if and only if there is a derivation in *SD* in which all of the primary assumptions are members of Γ and α occurs in the scope of only those primary assumptions.

We can state that a sentence is derivable from a set using the single turnstile symbol like this:

> $\Gamma \vdash \alpha$

We can say that a sentence is not derivable from a set like this:

> $\Gamma \nvdash \alpha$

Derivability is the fundamental concept of *SD*; other concepts will be defined in terms of derivability. For instance, validity and invalidity in *SD* are defined like this:

> An argument of *SL* is **valid in SD** if and only if the conclusion of the argument is derivable in *SD* from the set consisting of the premises.

> An argument of *SL* is **invalid in SD** if and only if it is not valid in *SD*.

It is possible to derive some sentences from no premises at all, which is to say that those sentences can be derived from the empty set. To do that, we begin with a subderivation. Here is an example:

Derive P ⊃ (Q ⊃ P)

1	P	
2	Q	
3	P	R 1
4	Q ⊃ P	⊃I 2–3
5	P ⊃ (Q ⊃ P)	⊃I 1–3

These sentences are called theorems of *SD*.

> A sentence of *SL* is a **theorem of *SD*** if and only if it is derivable in *SD* from the empty set.

We can also define equivalence in *SD*:

Two sentences α and β of *SL* are **equivalent in *SD*** if and only if β is derivable in *SD* from {α} and α is derivable in *SD* from {β}

> A set Γ of *SL* is **inconsistent in *SD*** if and only if both a sentence α of *SL* and its negation are derivable in *SD* from Γ.

> A set Γ of *SL* is **consistent** if and only if it is not inconsistent.

In **??**, we defined many of those same concepts for *SL*. There, we defined them in terms of truth in a model. Here, they are defined in terms of derivability, not truth. The derivation system operates solely at the level of syntax, and truth is a semantic concept. Nevertheless, we want the concepts of the derivation *SD* to parallel the semantic concepts of *SL*:

- A sentence α of *SL* is derivable in *SD* from a set of sentences Γ in *SL* if and only if α is truth-functionally entailed by Γ.
- An argument of *SL* is valid in *SD* if and only if the argument is truth-functionally valid.
- A sentence α of SL is a theorem in *SD* if and only if α is truth-functionally true.
- Sentences α and β or *SL* are equivalent in *SD* if and only if they are truth-functionally equivalent.

- A set of sentences of *SL* is consistent in *SD* if and only if it is truth-functionally consistent.

This means that if an argument is truth-functionally valid in *SL*, then it will be derivable in *SD*. Unfortunately, this does not mean that every student of logic will be able to construct a derivation for the argument. Constructing a successful derivation is much more likely if approached strategically, rather than haphazardly applying the derivation rules.

1.4 STRATEGIES FOR DERIVATIONS

The key to building derivations in *SD* is to focus on goals and sub-goals. The goal is the sentence that we have been tasked to derive. Strategizing derivations often looks something like this. Say that we have been tasked to derive some sentence α. Looking at the initial assumptions, we see that we can derive α if we can first derive β. We then see that deriving β requires deriving γ, and so on. So, α is our goal, but β abd γ become sub-goals. Let's work through an example:

Show that $\{\neg A, (\neg A \supset B) \& (C \equiv \neg A)\} \vdash (B \vee D) \& C$ in *SD*.

1	$\neg A$	
2	$(\neg A \supset B) \& (C \equiv \neg A)$	
⋮	⋮	
n	$(B \vee D) \& C$	

Now, note that the conclusion is a conjunction. We can use the Conjunction Introduction to form it, if we can derive each conjunct. The second conjunct, C, has to come from the right side of premise 2, which is $C \equiv \neg A$. So, our goal now becomes getting that on a line by itself. Since it's a conjunct, we can just use Conjunction Elimination.

1	$\neg A$	
2	$(\neg A \supset B) \& (C \equiv \neg A)$	
3	$C \equiv \neg A$	&E 2
⋮	⋮	
n	$(B \vee D) \& C$	

Now, we can use Biconditional Elimination to produce C.

1	¬A	
2	(¬A ⊃ B) & (C ≡ ¬A)	
3	C ≡ ¬A	&E 2
4	C	≡E 3, 4
⋮	⋮	
n	(B ∨ D) & C	

That takes care of half of the conclusion. Now, we need to get B ∨ D. There's no D in the premises, which means that we'll have to introduce it using Disjunction Introduction. To do that, we first need to get B. B is the consequent of a conditional, ¬A ⊃ B, so we can get B if we have the antecedent, ¬A. Fortunately, ¬A is the first premise. So our completed proof looks like this:

1	¬A	
2	(¬A ⊃ B) & (C ≡ ¬A)	
3	C ≡ ¬A	&E 2
4	C	≡E 3, 4
5	¬A ⊃ B	&E 2
6	B	⊃E 5, 1
7	B ∨ D	∨I 6
8	(B ∨ D) & C	&I 7, 4

1.5 THE DERIVATION SYSTEM SD+

So far, *SD* has only eleven rules, Reiteration and an introduction and elimination rule for each of the five logical connectives. These eleven rules are enough to derive the conclusion from the premises of every valid argument in *SL*. There is a trade-off, however—the simpler the derivation system in terms of the number of available derivation rules, the more complex the derivations tend to be. For example, consider this argument:

1. α ∨ β
2. ¬α
∴ β

The argument is obviously valid. The first premise tells us that least one of α and β are true, the second tells us that α is false, so β must therefore be true. Proving this with the basic rules of *SD*, however, is trickier than it seems it ought to be:

1	$\alpha \vee \beta$	
2	$\neg\alpha$	
3	α	
4	$\neg\beta$	
5	α	R 3
6	$\neg\alpha$	R 2
7	β	$\neg$E 4–6
8	β	
9	β	R 7
10	β	$\vee$E 3–7, 8–9

We have now proved, using the basic derivation rules of *SD*, that a disjunct can be derived from a disjunction and the negation of the other disjunct. So, we can add it as a new derivation rule to *SD*. There is no theoretical limit to the number of rules that could be added. We could, in principle, add a new rule for every derivation that we successfully complete. The resulting system, however, would be practically unusable. We need a good balance between the simplicity of the system itself and the simplicity of the derivations when using the system. There are two criteria that we can use to determine if an additional rule should be added to the system. First, the circumstances to which the proposed rule applies should occur fairly often, thus saving steps in many different derivations. Second, those circumstances are easily recognizable. Having new rules won't help, if we can't tell when we need to use them.

Rules of Inference

Here are the rules of inference that we will add:

Disjunctive Syllogism (DS)

$$
\begin{array}{c|l}
 & \alpha \lor \beta \\
 & \neg\alpha \\
\rhd & \beta
\end{array}
\quad\text{or}\quad
\begin{array}{c|l}
 & \alpha \lor \beta \\
 & \neg\beta \\
\rhd & \alpha
\end{array}
$$

Hypothetical Syllogism (HS)

$$
\begin{array}{c|l}
 & \alpha \supset \beta \\
 & \beta \supset \gamma \\
\rhd & \alpha \supset \gamma
\end{array}
$$

Modus Tollens (MT)

$$
\begin{array}{c|l}
 & \alpha \supset \beta \\
 & \neg\beta \\
\rhd & \neg\alpha
\end{array}
$$

Constructive Dilemma (CD)

$$
\begin{array}{c|l}
 & \alpha \supset \gamma \\
 & \beta \supset \delta \\
 & \alpha \lor \beta \\
\rhd & \gamma \lor \delta
\end{array}
$$

Destructive Dilemma (DD)

$$
\begin{array}{c|l}
 & \alpha \supset \gamma \\
 & \beta \supset \delta \\
 & \neg\gamma \lor \neg\delta \\
\rhd & \neg\alpha \lor \neg\beta
\end{array}
$$

Biconditional Modus Tollens (BMT)

$$
\begin{array}{c|l}
 & \alpha \equiv \beta \\
 & \neg\alpha \\
\rhd & \neg\beta
\end{array}
$$

Biconditional Hypothetical Syllogism (BHS)

$$
\begin{array}{c|l}
 & \alpha \equiv \beta \\
 & \beta \equiv \gamma \\
\rhd & \alpha \equiv \gamma
\end{array}
$$

RULES OF REPLACEMENT

In addition to rules of inference, *SD+* contains rules of replacement. Rules of replacement let us derive some sentences from other sentences by replacing parts of sentences. Unlike rules of inference, which can be applied only to whole lines, rules of replacement can be applied to parts of lines. For example, using a rule called Contraposition, from the sentence

$(A \lor B) \equiv (A \supset C)$

we can infer

$(A \lor B) \equiv (\neg C \supset \neg A)$

Since every sentence is a part of itself, the rules of replacement can also be applied to entire lines. The rules of replacement are also bidirectional. Using Contraposition, I can not only go from

$A \supset C$

to

$\neg C \supset \neg A$

I can also go from

$\neg C \supset \neg A$

to

$A \supset C$

Here are the rules of replacement of *SD+*:

De Morgan's (DM)

$\neg(\alpha \ \& \ \beta) \Leftrightarrow \neg\alpha \lor \neg\beta$
$\neg(\alpha \lor \beta) \Leftrightarrow \neg\alpha \ \& \ \neg\beta$

Association (Ass)

α ∨ (β ∨ γ) ⇔ (α ∨ β) ∨ γ
α & (β & γ) ⇔ (α & β) & γ

Distribution (Dis)

α & (β ∨ γ) ⇔ (α & β) ∨ (α & γ)
α ∨ (β & γ) ⇔ (α ∨ β) & (α ∨ γ)

Commutativity (Com)

α ∨ β ⇔ β ∨ α
α & β ⇔ β & α

Double Negation (DN)

α ⇔ ¬¬α

Contraposition (Cont)

α ⊃ β ⇔ ¬β ⊃ ¬α

Material Implication (Imp)

α ⊃ β ⇔ ¬α ∨ β

Material Equivalence (ME)

α ≡ β ⇔ (α ⊃ β) & (β ⊃ α)
α ≡ β ⇔ (α & β) ∨ (¬α & ¬β)

Exportation (Exp)

α ⊃ (β ⊃ γ) ⇔ (α & β) ⊃ γ

Tautology (Tau)

$\alpha \Leftrightarrow \alpha \mathbin{\&} \alpha$

$\alpha \Leftrightarrow \alpha \lor \alpha$

Biconditional De Morgan's (BDM)

$\neg(\alpha \equiv \beta) \Leftrightarrow \neg\alpha \equiv \beta$

Biconditional Commutativity (BCom)

$\alpha \equiv \beta \Leftrightarrow \beta \equiv \alpha$

Biconditional Inversion (BI)

$\alpha \equiv \beta \Leftrightarrow \neg\alpha \equiv \neg\beta$

Biconditional Association (BAss)

$\alpha \equiv (\beta \equiv \gamma) \Leftrightarrow (\alpha \equiv \beta) \equiv \gamma$

1	A ⊃ B	
2	B	DM 1