Contents

Part II Sentential Logic

1 Sentential Logic **2**
- 1.1 Sentence Letters . . . 2
- 1.2 Sentential Connectives . . . 4
- 1.3 Other Symbolization . . . 13
- 1.4 Recursive Syntax for SL . . . 14

2 Truth Tables **19**
- 2.1 Basic Concepts . . . 19
- 2.2 Complete Truth Tables . . . 20
- 2.3 Using Truth Tables . . . 23
- 2.4 Partial Truth Tables . . . 26
- 2.5 Expressive Completeness . . . 28
- 2.6 Recursive Definition of Truth in SL . . . 29

Part II

Sentential Logic

Chapter 1

Sentential Logic

This chapter introduces a logical language called SL. It is a version of *sentential logic*, because the basic units of the language will represent statements, and a statement is usually given by a complete sentence in English.

1.1 Sentence Letters

In SL, capital letters, called SENTENCE LETTERS are used to represent simple statements. Considered only as a symbol of SL, the letter A could mean any statement. So when translating from English into SL, it is important to provide a symbolization key, or dictionary. The SYMBOLIZATION KEY provides an English language sentence for each sentence letter used in the symbolization.

Consider this argument:

1. There is an apple on the desk.
2. If there is an apple on the desk, then Jenny made it to class.

$\therefore$ Jenny made it to class.

This is obviously a valid argument in English. In symbolizing it, we want to preserve the structure of the argument that makes it valid. What happens if we replace each sentence with a letter? Our symbolization key would look like this:

A: There is an apple on the desk.
B: If there is an apple on the desk, then Jenny made it to class.
C: Jenny made it to class.

We would then symbolize the argument in this way:

1. A
2. B

$\therefore C$

There is no necessary connection between some sentence A, which could be any statement, and some other sentences B and C, which could also be anything. The structure of the argument has been completely lost in this translation.

The important thing about the argument is that the second premise is not merely *any* statement, logically divorced from the other statement in the argument. The second premise contains the first premise and the conclusion *as parts*. Our symbolization key for the argument only needs to include meanings for A and C, and we can build the second premise from those pieces. So we symbolize the argument this way:

1. A
2. If A, then C.

$\therefore C$

This preserves the structure of the argument that makes it valid, but it still makes use of the English expression "If... then...." Although we ultimately want to replace all of the English expressions with logical notation, this is a good start.

The individual sentence letters in SL are called atomic sentences, because they are the basic building blocks out of which more complex sentences can be built. We can identify atomic sentences in English as well. An ATOMIC SENTENCE is one that cannot be broken into parts that are themselves sentences. "There is an apple on the desk" is an atomic sentence in English, because you can't find any proper part of it that forms a complete sentence. For instance "an apple on the desk" is a noun phrase, not a complete sentence. Similarly "on the desk" is a prepositional phrase, and not a sentence, and "is an" is not any kind of phrase at all. This is what you will find no matter how you divide "There is an apple on the desk." On the other hand you can find two proper parts of "If there is an apple on the desk, then Jenny made it to class" that are complete sentences: "There is an apple on the desk" and "Jenny made it to class." As a general rule, we will want to use atomic sentences in SL (that is, the sentence letters) to represent atomic sentences in English. Otherwise, we will lose some of the logical structure of the English sentence, as we have just seen.

There are only 26 letters of the alphabet, but there is no logical limit to the number of atomic sentences. We can use the same letter to symbolize different atomic sentences by adding a subscript, a small number written after the letter. We could have a symbolization key that looks like this:

$\mathbf{A}_1$: The apple is under the armoire.
$\mathbf{A}_2$: Arguments in SL always contain atomic sentences.
$\mathbf{A}_3$: Adam Ant is taking an airplane from Anchorage to Albany.
⋮
$\mathbf{A}_{294}$: Alliteration angers otherwise affable astronauts.

Keep in mind that each of these is a different sentence letter. When there are subscripts in the symbolization key, it is important to keep track of them.

1.2 Sentential Connectives

Logical connectives are used to build complex sentences from atomic components. In SL, our logical connectives are called SENTENTIAL CONNECTIVES because they connect sentence letters. There are five sentential connectives in SL. This table summarizes them, and they are explained below.

Symbol	What it is called	What it means
$\neg$	negation	"It is not the case that..."
$\wedge$	conjunction	"Both ... and ..."
$\vee$	disjunction	"Either ... or ..."
$\rightarrow$	conditional	"If ... then ..."
$\leftrightarrow$	biconditional	"... if and only if ..."

Negation

Consider how we might symbolize these sentences:

1. Mary is in Barcelona.
2. Mary is not in Barcelona.
3. Mary is somewhere other than Barcelona.

In order to symbolize sentence 1, we will need one sentence letter. We can provide a symbolization key:

B: Mary is in Barcelona.

Note that here we are giving B a different interpretation than we did in the previous section. The symbolization key only specifies what B means *in a specific context*. It is vital that we continue to use this meaning of B so long as we are talking about Mary and Barcelona. Later,

when we are symbolizing different sentences, we can write a new symbolization key and use B to mean something else.

Now, sentence 1 is simply B. Sentence 2 is obviously related to sentence 1: it is basically 1 with a "not" added. We could put the sentence partly our symbolic language by writing "Not B." This means we do not want to introduce a different sentence letter for 2. We just need a new symbol for the "not" part. Let's use the symbol '$\neg$,' which we will call NEGATION. Now we can translate 'Not B' to $\neg B$.

Sentence 3 is about whether or not Mary is in Barcelona, but it does not contain the word "not." Nevertheless, it is obviously logically equivalent to sentence 2. They both mean: It is not the case that Mary is in Barcelona. As such, we can translate both sentence 2 and sentence 3 as $\neg B$.

A sentence can be symbolized as $\neg\mathcal{A}$ if it can be paraphrased in English as "It is not the case that $\mathcal{A}$."

Consider these further examples:

4. The widget can be replaced if it breaks.
5. The widget is irreplaceable.
6. The widget is not irreplaceable.

If we let R mean "The widget is replaceable", then sentence 4 can be translated as R.

What about sentence 5? Saying the widget is irreplaceable means that it is not the case that the widget is replaceable. So even though sentence 5 is not negative in English, we symbolize it using negation as $\neg R$.

Sentence 6 can be paraphrased as "It is not the case that the widget is irreplaceable." Using negation twice, we translate this as $\neg\neg R$. The two negations in a row each work as negations, so the sentence means "It is not the case that it is not the case that R." If you think about the sentence in English, it is logically equivalent to sentence 4. So when we define logical equivalence in SL, we will make sure that R and $\neg\neg R$ are logically equivalent.

More examples:

7. Elliott is happy.
8. Elliott is unhappy.

If we let H mean "Elliot is happy", then we can symbolize sentence 7 as H.

However, it would be a mistake to symbolize sentence 8 as $\neg H$. If Elliott is unhappy, then he is not happy—but sentence 8 does not mean the same thing as "It is not the case that Elliott is

happy." It could be that he is not happy but that he is not unhappy either. Perhaps he is somewhere between the two. In order to symbolize sentence 8, we would need a new sentence letter.

For any sentence $\mathcal{A}$: If $\mathcal{A}$ is true, then $\neg\mathcal{A}$ is false. If $\neg\mathcal{A}$ is true, then $\mathcal{A}$ is false. Using T for true and F for false, we can summarize this in a *characteristic truth table* for negation:

$\mathcal{A}$	$\neg\mathcal{A}$
T	F
F	T

We will discuss truth tables at greater length in the next chapter.

Conjunction

Consider these sentences:

9. Adam is athletic.
10. Barbara is athletic.
11. Adam is athletic, and Barbara is also athletic.

We will need separate sentence letters for 9 and 10, so we define this symbolization key:

A: Adam is athletic.
B: Barbara is athletic.

Sentence 9 can be symbolized as A.

Sentence 10 can be symbolized as B.

Sentence 11 can be paraphrased as "A and B." In order to fully symbolize this sentence, we need another symbol. We will use $\wedge$. We translate "A and B" as $A \wedge B$. The logical connective $\wedge$is called the CONJUNCTION, and A and B are each called CONJUNCTS.

Notice that we make no attempt to symbolize "also" in sentence 11. Words like "both" and "also" function to draw our attention to the fact that two things are being conjoined. They are not doing any further logical work, so we do not need to represent them in SL.

Some more examples:

12. Barbara is athletic and energetic.
13. Barbara and Adam are both athletic.
14. Although Barbara is energetic, she is not athletic.

15. Barbara is athletic, but Adam is more athletic than she is.

Sentence 12 is obviously a conjunction. The sentence says two things about Barbara, so in English it is permissible to refer to Barbara only once. It might be tempting to try this when translating the argument: Since B means "Barbara is athletic", one might paraphrase the sentences as "B and energetic." This would be a mistake. Once we translate part of a sentence as B, any further structure is lost. B is an atomic sentence; it is nothing more than true or false. Conversely, "energetic" is not a sentence; on its own it is neither true nor false. We should instead paraphrase the sentence as "B and Barbara is energetic." Now we need to add a sentence letter to the symbolization key. Let E mean "Barbara is energetic." Now the sentence can be translated as $B \wedge E$.

> A sentence can be symbolized as $\mathcal{A} \wedge \mathcal{B}$ if it can be paraphrased in English as 'Both $\mathcal{A}$, and $\mathcal{B}$.' Each of the conjuncts must be a sentence.

Sentence 13 says one thing about two different subjects. It says of both Barbara and Adam that they are athletic, and in English we use the word "athletic" only once. In translating to SL, it is important to realize that the sentence can be paraphrased as, "Barbara is athletic, and Adam is athletic." This translates as $B \wedge A$.

Sentence 14 is a bit more complicated. The word "although" sets up a contrast between the first part of the sentence and the second part. Nevertheless, the sentence says both that Barbara is energetic and that she is not athletic. In order to make each of the conjuncts an atomic sentence, we need to replace "she" with "Barbara."

So we can paraphrase sentence 14 as, "*Both* Barbara is energetic, *and* Barbara is not athletic." The second conjunct contains a negation, so we paraphrase further: "*Both* Barbara is energetic *and it is not the case that* Barbara is athletic." This translates as $E \wedge \neg B$.

Sentence 15 contains a similar contrastive structure. It is irrelevant for the purpose of translating to SL, so we can paraphrase the sentence as "*Both* Barbara is athletic, *and* Adam is more athletic than Barbara." (Notice that we once again replace the pronoun "she" with her name.) How should we translate the second conjunct? We already have the sentence letter A which is about Adam's being athletic and B which is about Barbara's being athletic, but neither is about one of them being more athletic than the other. We need a new sentence letter. Let R mean "Adam is more athletic than Barbara." Now the sentence translates as $B \wedge R$.

> Sentences that can be paraphrased "$\mathcal{A}$, but $\mathcal{B}$" or "Although $\mathcal{A}$, $\mathcal{B}$" are best symbolized using conjunction $\mathcal{A} \wedge \mathcal{B}$.

It is important to keep in mind that the sentence letters A, B, and R are atomic sentences. Considered as symbols of SL, they have no meaning beyond being true or false. We have used

them to symbolize different English language sentences that are all about people being athletic, but this similarity is completely lost when we translate to SL. No formal language can capture all the structure of the English language, but as long as this structure is not important to the argument there is nothing lost by leaving it out.

For any sentences $\mathcal{A}$ and $\mathcal{B}$, $\mathcal{A} \wedge \mathcal{B}$ is true if and only if both $\mathcal{A}$ and $\mathcal{B}$ are true. We can summarize this in the characteristic truth table for conjunction:

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A} \wedge \mathcal{B}$
T	T	T
T	F	F
F	T	F
F	F	F

Conjunction is symmetrical because we can swap the conjuncts without changing the truth value of the sentence. Regardless of what $\mathcal{A}$ and $\mathcal{B}$ are, $\mathcal{A} \wedge \mathcal{B}$ is logically equivalent to $\mathcal{B} \wedge \mathcal{A}$.

Disjunction

Consider these sentences:

16. Either Denison will play golf with me, or he will watch movies.
17. Either Denison or Ellery will play golf with me.

For these sentences we can use this symbolization key:

D: Denison will play golf with me.
E: Ellery will play golf with me.
M: Denison will watch movies.

Sentence 16 is "Either D or M." To fully symbolize this, we introduce a new symbol. The sentence becomes $D \vee M$. The $\vee$connective is called DISJUNCTION, and D and M are called DISJUNCTS.

Sentence 17 is only slightly more complicated. There are two subjects, but the English sentence only gives the verb once. In translating, we can paraphrase it as "Either Denison will play golf with me, or Ellery will play golf with me." Now it obviously translates as $D \vee E$.

A sentence can be symbolized as $\mathcal{A} \vee \mathcal{B}$ if it can be paraphrased in English as "Either $\mathcal{A}$ or $\mathcal{B}$." Each of the disjuncts must be a sentence.

Sometimes in English, the word "or" excludes the possibility that both disjuncts are true. This is called an EXCLUSIVE OR. An *exclusive or* is clearly intended when a restaurant menu says, "Entrees come with either soup or salad." You may have soup; you may have salad; but, if you want *both* soup *and* salad, then you will have to pay extra.

At other times, the word "or" allows for the possibility that both disjuncts might be true. This is probably the case with sentence 17, above. I might play with Denison, with Ellery, or with both Denison and Ellery. Sentence 17 merely says that I will play with *at least* one of them. This is called an INCLUSIVE OR.

The symbol $\vee$ represents an *inclusive or*. So $D \vee E$ is true if D is true, if E is true, or if both D and E are true. It is false only if both D and E are false. We can summarize this with the characteristic truth table for disjunction:

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A} \vee \mathcal{B}$
T	T	T
T	F	T
F	T	T
F	F	F

Like conjunction, disjunction is symmetrical. $\mathcal{A} \vee \mathcal{B}$ is logically equivalent to $\mathcal{B} \vee \mathcal{A}$.

These sentences are somewhat more complicated:

18. Either you will not have soup, or you will not have salad.
19. You will have neither soup nor salad.
20. You get either soup or salad, but not both.

We let S_1 mean that you get soup and S_2 mean that you get salad.

Sentence 18 can be paraphrased in this way: "Either *it is not the case that* you get soup, or *it is not the case that* you get salad." Translating this requires both disjunction and negation. It becomes $\neg S_1 \vee \neg S_2$.

Sentence 19 also requires negation. It can be paraphrased as, "*It is not the case that* either you get soup or you get salad." We need some way of indicating that the negation does not just negate the right or left disjunct, but rather negates the entire disjunction. In order to do this, we put parentheses around the disjunction: "It is not the case that $(S_1 \vee S_2)$." This becomes simply $\neg(S_1 \vee S_2)$.

Notice that the parentheses are doing important work here. The sentence $\neg S_1 \vee S_2$ would mean "Either you will not have soup, or you will have salad."

Sentence 20 is an *exclusive or*. Although $\vee$ is an inclusive or, we can symbolize an exclusive or in SL. We just need more than one connective to do it. We can break the sentence into two parts.

The first part says that you get one or the other. We translate this as $(S_1 \vee S_2)$. The second part says that you do not get both. We can paraphrase this as "It is not the case both that you get soup and that you get salad." Using both negation and conjunction, we translate this as $\neg(S_1 \wedge S_2)$. Now we just need to put the two parts together. As we saw above, "but" can usually be translated as a conjunction. Sentence 20 can thus be translated as $(S_1 \vee S_2) \wedge \neg(S_1 \wedge S_2)$.

Conditional

For the following sentences, let R mean "You will cut the red wire" and B mean "The bomb will explode."

21. If you cut the red wire, then the bomb will explode.
22. The bomb will explode only if you cut the red wire.

Sentence 21 can be translated partially as "If R, then B." We will use the symbol $\rightarrow$ to represent logical entailment. Sentence 21 then becomes $R \rightarrow B$. The connective is called a CONDITIONAL. The sentence on the left-hand side of the conditional (R in this example) is called the ANTECEDENT. The sentence on the right-hand side (B) is called the CONSEQUENT.

Sentence 22 is also a conditional. Since the word "if" appears in the second half of the sentence, it might be tempting to symbolize this in the same way as sentence 21. That would be a mistake.

The conditional $R \rightarrow B$ says that *if* R were true, *then* B would also be true. It does not say that you cutting the red wire is the *only* way that the bomb could explode. Someone else might cut the wire, or the bomb might be on a timer. The sentence $R \rightarrow B$ does not say anything about what to expect if R is false. Sentence 22 is different. It says that the only conditions under which the bomb will explode involve you having cut the red wire; i.e., if the bomb explodes, then you must have cut the wire. As such, sentence 22 should be symbolized as $B \rightarrow R$.

It is important to remember that the connective $\rightarrow$ says only that, if the antecedent is true, then the consequent is true. It says nothing about the *causal* connection between the two events. Translating sentence 22 as $B \rightarrow R$ does not mean that the bomb exploding would somehow have caused you cutting the wire. Both sentence 21 and 22 suggest that, if you cut the red wire, you cutting the red wire would be the cause of the bomb exploding. They differ on the *logical* connection. If sentence 22 were true, then an explosion would tell us—those of us safely away from the bomb—that you had cut the red wire. Without an explosion, sentence 22 tells us nothing.

The paraphrased sentence "$\mathcal{A}$ only if $\mathcal{B}$" is logically equivalent to "If $\mathcal{A}$, then $\mathcal{B}$."

"If $\mathcal{A}$, then $\mathcal{B}$" means that if $\mathcal{A}$ is true, then so is $\mathcal{B}$. So we know that if the antecedent $\mathcal{A}$ is true but the consequent $\mathcal{B}$ is false, then the conditional "If $\mathcal{A}$ then $\mathcal{B}$" is false. What is the truth value of "If $\mathcal{A}$, then $\mathcal{B}$" under other circumstances? Suppose, for instance, that the antecedent $\mathcal{A}$

happened to be false. "If $\mathcal{A}$, then $\mathcal{B}$" would then not tell us anything about the actual truth value of the consequent $\mathcal{B}$, and it is unclear what the truth value of "If $\mathcal{A}$, then $\mathcal{B}$" would be.

In English, the truth of conditionals often depends on what *would* be the case if the antecedent *were true*—even if, as a matter of fact, the antecedent is false. This poses a problem for translating conditionals into SL. Considered as sentences of SL, R and B in the above examples have nothing intrinsic to do with each other. In order to consider what the world would be like if R were true, we would need to analyze what R says about the world. Since R is an atomic symbol of SL, however, there is no further structure to be analyzed. When we replace a sentence with a sentence letter, we consider it merely as some atomic sentence that might be true or false.

In order to translate conditionals into SL, we will not try to capture all the subtleties of the English language "If..., then..." Instead, the symbol $\rightarrow$ will be what logicians call a material conditional. This means that when $\mathcal{A}$ is false, the conditional $\mathcal{A} \rightarrow \mathcal{B}$ is automatically true, regardless of the truth value of $\mathcal{B}$. If both $\mathcal{A}$ and $\mathcal{B}$ are true, then the conditional $\mathcal{A} \rightarrow \mathcal{B}$ is true.

In short, $\mathcal{A} \rightarrow \mathcal{B}$ is false if and only if $\mathcal{A}$ is true and $\mathcal{B}$ is false. We can summarize this with a characteristic truth table for the conditional.

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A} \rightarrow \mathcal{B}$
T	T	T
T	F	F
F	T	T
F	F	T

The conditional is asymmetrical. You cannot swap the antecedent and consequent without changing the meaning of the sentence, because $\mathcal{A} \rightarrow \mathcal{B}$ and $\mathcal{B} \rightarrow \mathcal{A}$ are not logically equivalent.

Not all sentences of the form "If..., then..." are conditionals. Consider this sentence:

23. If anyone wants to see me, then I will be on the porch.

When I say this, it means that I will be on the porch, regardless of whether anyone wants to see me or not—but if someone did want to see me, then they should look for me there. If we let P mean "I will be on the porch," then sentence 23 can be translated simply as P.

Conditionals also occur as of *necessary* and *sufficient* conditions. Consider the following:

24. Being a dog is a sufficient condition for being a mammal.

Which is clearly distinct from

25. Being a mammal is a sufficient condition for being a dog.

Here we understand the phrase "X is a sufficient condition of Y" to mean that "X cannot be true without Y also being true, which is just another way to say "If X is true, then Y is also true." So, 24 amounts to the biologically correct statement that

26. If you are a dog, you are also a mammal.

Now compare 25 with

27. Being a mammal is a necessary condition for being a dog.

25 is clearly false, according to standard biology: being mammal alone isn't enough - sufficient - to be a dog. However, 27 is true - this is because being a mammal, while not sufficient, is a requirement - necessary condition - for being a dog. In few words, if something isn't mammal, then we also know that it's not a dog. To put it yet another way, if something is a dog, then it has got to be a mammal. So 27 ends up being logically equivalent to 24. In sum,

> The paraphrased sentence "$\mathcal{A}$ is a sufficient condition for $\mathcal{B}$" is logically equivalent to "If $\mathcal{A}$, then $\mathcal{B}$."
> The paraphrased sentence "$\mathcal{A}$ is a necessary condition for $\mathcal{B}$" is logically equivalent to "If $\mathcal{B}$, then $\mathcal{A}$."

Biconditional

Consider these sentences:

28. The figure on the board is a triangle only if it has exactly three sides.
29. The figure on the board is a triangle if it has exactly three sides.
30. The figure on the board is a triangle if and only if it has exactly three sides.

Let T mean "The figure is a triangle" and S mean "The figure has three sides."

Sentence 28, for reasons discussed above, can be translated as $T \to S$.

Sentence 29 is importantly different. It can be paraphrased as "If the figure has three sides, then it is a triangle." So it can be translated as $S \to T$.

Sentence 30 says that T is true *if and only if* S is true; we can infer S from T, and we can infer T from S. This is called a BICONDITIONAL, because it entails the two conditionals $S \to T$ and $T \to S$. We will use $\leftrightarrow$to represent the biconditional; sentence 30 can be translated as $S \leftrightarrow T$.

We could abide without a new symbol for the biconditional. Since sentence 30 means "$T \rightarrow S$ and $S \rightarrow T$," we could translate it as $(T \rightarrow S) \wedge (S \rightarrow T)$. We would need parentheses to indicate that $(T \rightarrow S)$ and $(S \rightarrow T)$ are separate conjuncts; the expression $T \rightarrow S \wedge S \rightarrow T$ would be ambiguous.

Because we could always write $(\mathcal{A} \rightarrow \mathcal{B}) \wedge (\mathcal{B} \rightarrow \mathcal{A})$ instead of $\mathcal{A} \leftrightarrow \mathcal{B}$, we do not strictly speaking *need* to introduce a new symbol for the biconditional. Nevertheless, logical languages usually have such a symbol. SL will have one, which makes it easier to translate phrases like "if and only if."

$\mathcal{A} \leftrightarrow \mathcal{B}$ is true if and only if $\mathcal{A}$ and $\mathcal{B}$ have the same truth value. This is the characteristic truth table for the biconditional:

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A} \leftrightarrow \mathcal{B}$
T	T	T
T	F	F
F	T	F
F	F	T

Question: What is connection between biconditionals and necessary/sufficient conditions? (Spoiler: you will be asked about this in the test.)

1.3 Other Symbolization

We have now introduced all of the connectives of SL. We can use them together to translate many kinds of sentences. Consider these examples of sentences that use the English-language connective "unless":

31. Unless you wear a jacket, you will catch cold.
32. You will catch cold unless you wear a jacket.

Let J mean "You will wear a jacket" and let D mean "You will catch a cold."

We can paraphrase sentence 31 as "Unless J, D." This means that if you do not wear a jacket, then you will catch cold; with this in mind, we might translate it as $\neg J \rightarrow D$. It also means that if you do not catch a cold, then you must have worn a jacket; with this in mind, we might translate it as $\neg D \rightarrow J$.

Which of these is the correct translation of sentence 31? Both translations are correct, because the two translations are logically equivalent in SL.

Sentence 32, in English, is logically equivalent to sentence 31. It can be translated as either $\neg J \rightarrow D$ or $\neg D \rightarrow J$.

If a sentence can be paraphrased as "Unless $\mathcal{A}$, $\mathcal{B}$," then it can be symbolized as $\neg\mathcal{A} \to \mathcal{B}$.

1.4 Recursive Syntax for SL

The previous two sections gave you a rough, informal sense of how to create sentences in SL. If I give you an English sentence like "Grass is either green or brown," you should be able to write a corresponding sentence in SL: "$A \vee B$." In this section we want to give a more precise definition of a sentence in SL. When we defined sentences in English, we did so using the concept of truth: Sentences were units of language that can be true or false. In SL, it is possible to define what counts as a sentence without talking about truth. Instead, we can just talk about the structure of the sentence. This is one respect in which a formal language like SL is more precise than a natural language like English.

The structure of a sentence in SL considered without reference to truth or falsity is called its syntax. More generally SYNTAX refers to the study of the properties of language that are there even when you don't consider meaning. Whether a sentence is true or false is considered part of its meaning. In this chapter, we will be giving a purely syntactical definition of a sentence in SL. The contrasting term is SEMANTICS the study of aspects of language that relate to meaning, including truth and falsity. (The word "semantics" comes from the Greek word for "mark")

If we are going to define a sentence in SL just using syntax, we will need to carefully distinguish SL from the language that we use to talk about SL. When you create an artificial language like SL, the language that you are creating is called the OBJECT LANGUAGE. The language that we use to talk about the object language is called the METALANGUAGE. Imagine building a house. The object language is like the house itself. It is the thing we are building. While you are building a house, you might put up scaffolding around it. The scaffolding isn't part of the the house. You just use it to build the house. The metalanguage is like the scaffolding.

The object language in this chapter is SL. For the most part, we can build this language just by talking about it in ordinary English. However we will also have to build some special scaffolding that is not a part of SL, but will help us build SL. Our metalanguage will thus be ordinary English plus this scaffolding.

An important part of the scaffolding are the METAVARIABLES These are the fancy script letters we have been using in the characteristic truth tables for the connectives: $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, etc. These are letters that can refer to any sentence in SL. They can represent sentences like P or Q, or they can represent longer sentences, like $(((A \vee B) \wedge G) \to (P \leftrightarrow Q))$. Just as the sentence letters A, B, etc. are variables that range over any English sentence, the metavariables $\mathcal{A}$, $\mathcal{B}$, etc. are variables that range over any sentence in SL, including the sentence letters A, B, etc.

As we said, in this chapter we will give a syntactic definition for "sentence of SL." The definition itself will be given in mathematical English, the metalanguage.

sentence letters	$A, B, C, \ldots, Z$
with subscripts, as needed	$A_1, B_1, Z_1, A_2, A_{25}, J_{375}, \ldots$
connectives	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
parentheses	(,)

Most random combinations of these symbols will not count as sentences in SL. Any random connection of these symbols will just be called a "string" or "expression" Random strings only become meaningful sentences when the are structured according to the rules of syntax. We saw from the earlier two sections that individual sentence letters, like A and G_{13} counted as sentences. We also saw that we can put these sentences together using connectives so that $\neg A$ and $\neg G_{13}$ is a sentence. The problem is, we can't simply list all the different sentences we can put together this way, because there are infinitely many of them. Instead, we will define a sentence in SL by specifying the process by which they are constructed.

Consider negation: Given any sentence $\mathcal{A}$ of SL, $\neg\mathcal{A}$ is a sentence of SL. It is important here that $\mathcal{A}$ is not the sentence letter A. Rather, it is a metavariable: part of the metalanguage, not the object language. Since $\mathcal{A}$ is not a symbol of SL, $\neg\mathcal{A}$ is not an expression of SL. Instead, it is an expression of the metalanguage that allows us to talk about infinitely many expressions of SL: all of the expressions that start with the negation symbol.

We can say similar things for each of the other connectives. For instance, if $\mathcal{A}$ and $\mathcal{B}$ are sentences of SL, then $(\mathcal{A} \wedge \mathcal{B})$ is a sentence of SL. Providing clauses like this for all of the connectives, we arrive at the following formal definition for a SENTENCE OF SL:

1. Every atomic sentence is a sentence.
2. If $\mathcal{A}$ is a sentence, then $\neg\mathcal{A}$ is a sentence of SL.
3. If $\mathcal{A}$ and $\mathcal{B}$ are sentences, then $(\mathcal{A} \wedge \mathcal{B})$ is a sentence.
4. If $\mathcal{A}$ and $\mathcal{B}$ are sentences, then $(\mathcal{A} \vee \mathcal{B})$ is a sentence.
5. If $\mathcal{A}$ and $\mathcal{B}$ are sentences, then $(\mathcal{A} \rightarrow \mathcal{B})$ is a sentence.
6. If $\mathcal{A}$ and $\mathcal{B}$ are sentences, then $(\mathcal{A} \leftrightarrow \mathcal{B})$ is a sentence.
7. All and only sentences of SL can be generated by applications of these rules.

Why we need recursion

Recursion is absolutely crucial in computability theory and a common design of algorithms. Recursion allows us divide a complex problem into smaller components, so we can conquer them separately. One example is to use recursion to evaluate the factorial function. Consider this *recursive* definition of factorials. For every $a \in \mathbb{N}$, it is such that

1. $0! = 1$

2. $(a+1)! = (a+1) \times a!$

With these two simple lines, we have recursively defined how to compute this function with any a. We do this by working backward to 0!. Take 4! for example:

$$\begin{aligned} 4! &= 4 \times 3! && \text{(by line 2)} \\ &= 4 \times 3 \times 2! && \text{(by line 2)} \\ &= 4 \times 3 \times 2 \times 1! && \text{(by line 2)} \\ &= 4 \times 3 \times 2 \times 1 \times 0! && \text{(by line 2)} \\ &= 4 \times 3 \times 2 \times 1 \times 1 && \text{(by line 1)} \end{aligned}$$

The recursive definition of SL syntax solves an importantly problem for us in a similar way: how can we determine if an *arbitrary* string is a sentence in SL?

To begin, a piece of terminology: any arbitrary string that obeys the syntax of SL is called a well-formed formula (wff). We can apply the definition above to see whether an arbitrary string is a wff.

Suppose we want to know whether or not $\neg\neg\neg D$ is a sentence of SL. Looking at the second clause of the definition, we know that $\neg\neg\neg D$ is a sentence *if* $\neg\neg D$ is a sentence. So now we need to ask whether or not $\neg\neg D$ is a sentence. Again looking at the second clause of the definition, $\neg\neg D$ is a sentence *if* $\neg D$ is. Again, $\neg D$ is a sentence *if* D is a sentence. Now D is a sentence letter, an atomic sentence of SL, so we know that D is a sentence by the first clause of the definition. So for a compound formula like $\neg\neg\neg D$, we must apply the definition repeatedly. Eventually we arrive at the atomic sentences from which the sentence is built up.

The parallel with the factorial example should be obvious. Just as we ask what 4! is by working backward from 3! to 0!, we ask whether $\neg\neg\neg D$ is a wff by working backward to the base case: the atomic sentence D. Eventually we will want to *prove* that this property holds for any given sentence. We will talk about that in later chapters.

Definitions like this are called recursive. RECURSIVE DEFINITIONS begin with some specifiable base elements and define ways to indefinitely compound the base elements. Just as the recursive definition allows complex sentences to be built up from simple parts, you can use it to decompose sentences into their simpler parts. To determine whether or not something meets the definition, you may have to refer back to the definition many times.

When you use a connective to build a longer sentence from shorter ones, the shorter sentences are said to be in the SCOPE of the connective. So in the sentence $(A \wedge B) \rightarrow C$, the scope of the connective $\rightarrow$ includes $(A \wedge B)$ and C. In the sentence $\neg(A \wedge B)$ the scope of the $\neg$ is $(A \wedge B)$. On the other hand, in the sentence $\neg A \wedge B$ the scope of the $\neg$ is just A.

The last connective that you add when you assemble a sentence using the recursive definition is the MAIN CONNECTIVE of that sentence. For example: The main logical operator of

$\neg(E \vee (F \rightarrow G))$ is negation, $\neg$. The main logical operator of $(\neg E \vee (F \rightarrow G))$ is disjunction, $\vee$. The main connective of any sentence will have all the rest of the sentence in its scope.

The recursive structure of sentences in SL will be important when we consider the circumstances under which a particular sentence would be true or false. The sentence $\neg\neg\neg D$ is true if and only if the sentence $\neg\neg D$ is false, and so on through the structure of the sentence until we arrive at the atomic components: $\neg\neg\neg D$ is true if and only if the atomic sentence D is false. We will return to this point in the next chapter.

Notational conventions

A sentence like $(Q \wedge R)$ must be surrounded by parentheses, because we might apply the definition again to use this as part of a more complicated sentence. If we negate $(Q \wedge R)$, we get $\neg(Q \wedge R)$. If we just had $Q \wedge R$ without the parentheses and put a negation in front of it, we would have $\neg Q \wedge R$. It is most natural to read this as meaning the same thing as $(\neg Q \wedge R)$, something very different than $\neg(Q \wedge R)$. The sentence $\neg(Q \wedge R)$ means that it is not the case that both Q and R are true; Q might be false or R might be false, but the sentence does not tell us which. The sentence $(\neg Q \wedge R)$ means specifically that Q is false and that R is true. As such, parentheses are crucial to the meaning of the sentence.

So, strictly speaking, $Q \wedge R$ without parentheses is *not* a sentence of SL. When using SL, however, we will often be able to relax the precise definition so as to make things easier for ourselves. We will do this in several ways.

First, we understand that $Q \wedge R$ means the same thing as $(Q \wedge R)$. As a matter of convention, we can leave off parentheses that occur *around the entire sentence*.

Second, it can sometimes be confusing to look at long sentences with many nested pairs of parentheses. We adopt the convention of using square brackets [and] in place of parentheses. There is no logical difference between $(P \vee Q)$ and $[P \vee Q]$, for example. The unwieldy sentence

$$(((H \rightarrow I) \vee (I \rightarrow H)) \wedge (J \vee K))$$

could be written in this way:

$$\big[(H \rightarrow I) \vee (I \rightarrow H)\big] \wedge (J \vee K)$$

Third, we will sometimes want to translate the conjunction of three or more sentences. For the sentence "Alice, Bob, and Candice all went to the party," suppose we let A mean "Alice went," B mean "Bob went," and C mean "Candice went." The definition only allows us to form a conjunction out of two sentences, so we can translate it as $(A \wedge B) \wedge C$ or as $A \wedge (B \wedge C)$. There is no reason to distinguish between these, since the two translations are logically equivalent. There is no logical difference between the first, in which $(A \wedge B)$ is conjoined with C, and the second, in which A is conjoined with $(B \wedge C)$. So we might as well just write $A \wedge B \wedge C$. As a matter of convention, we can leave out parentheses when we conjoin three or more sentences.

Fourth, a similar situation arises with multiple disjunctions. "Either Alice, Bob, or Candice went to the party" can be translated as $(A \vee B) \vee C$ or as $A \vee (B \vee C)$. Since these two translations are logically equivalent, we may write $A \vee B \vee C$.

These latter two conventions only apply to multiple conjunctions or multiple disjunctions. If a series of connectives includes both disjunctions and conjunctions, then the parentheses are essential; as with $(A \wedge B) \vee C$ and $A \wedge (B \vee C)$. The parentheses are also required if there is a series of conditionals or biconditionals; as with $(A \rightarrow B) \rightarrow C$ and $A \leftrightarrow (B \leftrightarrow C)$.

We have adopted these four rules as notational conventions, not as changes to the definition of a sentence. Strictly speaking, $A \vee B \vee C$ is still not a sentence. Instead, it is a kind of shorthand. We write it for the sake of convenience, but we really mean the sentence $(A \vee (B \vee C))$.

If we had given a different definition for a sentence, then these could count as sentences. We might have written rule 3 in this way: "If $\mathcal{A}$, $\mathcal{B}$, ... $\mathcal{Z}$ are sentences, then $(\mathcal{A} \wedge \mathcal{B} \wedge \ldots \wedge \mathcal{Z})$, is a sentence ." This would make it easier to translate some English sentences, but would have the cost of making our formal language more complicated. We would have to keep the complex definition in mind when we develop truth tables and a proof system. We want a logical language that is expressively simple and allows us to translate easily from English, but we also want a formally simple language. Adopting notational conventions is a compromise between these two desires.

Key Terms

Antecedent

Atomic sentence

Biconditional

Conditional

Conjunct

Conjunction

Consequent

Disjunct

Disjunction

Main connective

Metalanguage

Metavariables

Negation

Object language

Recursive definition

Scope

Semantics

Sentence letter

Sentence of SL

Sentential connective

Symbolization key

Syntax

Chapter 2

Truth Tables

This chapter introduces a way of evaluating sentences and arguments of SL called the truth table method. As we shall see, the truth table method is *semantic* because it involves one aspect of the meaning of sentences, whether those sentences are true or false. As we saw on page 14, semantics is the study of aspects of language related to meaning, including truth and falsity. Although it can be laborious, the truth table method is a purely mechanical procedure that requires no intuition or special insight.

,

2.1 Basic Concepts

A formal language is built from two kinds of elements: logical symbols and nonlogical symbols. LOGICAL SYMBOLS have their meaning fixed by the formal language. In SL, the logical symbols are the sentential connectives and the parentheses. When writing a symbolization key, you are not allowed to change the meaning of the logical symbols. You cannot say, for instance, that the ¬ symbol will mean "not" in one argument and "perhaps" in another. The ¬ symbol always means logical negation. It is used to translate the English language word "not", but it is a symbol of a formal language and is defined by its truth conditions.

The NONLOGICAL SYMBOLS are defined simply as all the symbols that aren't logical. The nonlogical symbols in SL are the sentence letters. When we translate an argument from English to SL, for example, the sentence letter M does not have its meaning fixed in advance; instead, we provide a symbolization key that says how M should be interpreted in that argument. When translating from English to a formal language, we provided symbolization keys which were interpretations of all the nonlogical symbols we used in the translation.

In logic, when we study artificial languages, we investigate their semantics by providing an

interpretation of the nonlogical symbols. An INTERPRETATION is a way of setting up a correspondence between elements of the object language and elements of some other language or logical structure. The symbolization keys we defined in Chapter 1 (p. 2) are a sort of interpretation.

The truth table method will also involve giving an interpretation of sentences, but they will be much simpler than the translation schemes we used in Chapter 2. We will not be concerned with what the individual sentence letters mean. We will only care whether they are true or false. We can do this, because of the way that the meaning of larger sentences is generated by the meaning of their parts.

Any nonatomic sentence of SL is composed of atomic sentences with sentential connectives. The truth value of the compound sentence depends only on the truth value of the atomic sentences that it comprises. In order to know the truth value of $(D \leftrightarrow E)$, for instance, you only need to know the truth value of D and the truth value of E. Connectives that work in this way are called truth functional. More technically, we define a TRUTH-FUNCTIONAL CONNECTIVE as an operator that builds larger sentences out of smaller ones, and fixes the truth value of the resulting sentence based only on the truth value of the component sentences.

Because all of the logical symbols in SL are truth functional, the only aspect of meaning we need to worry about in studying the semantics of SL is truth and falsity. If we want to know about the truth of the sentence $A \wedge B$, the only thing we need to know is whether A and B are true. It doesn't actually matter what else they mean. So if A is false, then $A \wedge B$ is false no matter what false sentence A is used to represent. It could be "I am the Pope" or "Pi is equal to 3.19." The larger sentence $A \wedge B$ is still false. So to give an interpretation of sentences in SL, all we need to do is create a truth assignment. A TRUTH ASSIGNMENT is a function that maps the sentence letters in SL onto our two truth values. In other words, we just need to assign Ts and Fs to all our sentence letters.

It is worth knowing that most languages are not built only out of truth functional connectives. In English, it is possible to form a new sentence from any simpler sentence $\mathcal{X}$ by saying "It is possible that $\mathcal{X}$." The truth value of this new sentence does not depend directly on the truth value of $\mathcal{X}$. Even if $\mathcal{X}$ is false, perhaps in some sense $\mathcal{X}$ *could* have been true—then the new sentence would be true. Some formal languages, called *modal logics*, have an operator for possibility. In a modal logic, we could translate "It is possible that $\mathcal{X}$" as $\Diamond\mathcal{X}$. However, the ability to translate sentences like these comes at a cost: The $\Diamond$ operator is not truth-functional, and so modal logics are not amenable to truth tables.

2.2 Complete Truth Tables

In the last chapter we introduced the characteristic truth tables for the different connectives. To put them all in one place, the truth tables for the connectives of SL are repeated in Table 2.1. Notice that when we did this, we listed all the possible combinations of truth and falsity for the sentence letters in these basic sentences. Each line of the truth table is thus a *truth assignment* for the sentence letters used in the sentence we are giving a truth table for. Thus one line of the

$\mathcal{A}$	$\neg\mathcal{A}$
T	F
F	T

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A}\wedge\mathcal{B}$	$\mathcal{A}\vee\mathcal{B}$	$\mathcal{A}\rightarrow\mathcal{B}$	$\mathcal{A}\leftrightarrow\mathcal{B}$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

Table 2.1: The characteristic truth tables for the connectives of SL.

truth table is all we need to give an *interpretation* of the sentence, and the full table gives all the possible interpretations of the sentence.

The truth value of sentences that contain only one connective is given by the characteristic truth table for that connective. The characteristic truth table for conjunction, for example, gives the truth conditions for any sentence of the form $(\mathcal{A} \wedge \mathcal{B})$. Even if the conjuncts $\mathcal{A}$ and $\mathcal{B}$ are long, complicated sentences, the conjunction is true if and only if both $\mathcal{A}$ and $\mathcal{B}$ are true. Consider the sentence $(H \wedge I) \rightarrow H$. We consider all the possible combinations of true and false for H and I, which gives us four rows. We then copy the truth values for the sentence letters and write them underneath the letters in the sentence.

H	I	$(H$	$\wedge$	$I)$	$\rightarrow$	H
T	T	**T**		**T**		**T**
T	F	**T**		**F**		**T**
F	T	**F**		**T**		**F**
F	F	**F**		**F**		**F**

Now consider the subsentence $H \wedge I$. This is a conjunction $\mathcal{A}\wedge\mathcal{B}$ with H as $\mathcal{A}$ and with I as $\mathcal{B}$. H and I are both true on the first row. Since a conjunction is true when both conjuncts are true, we write a T underneath the conjunction symbol. We continue for the other three rows and get this:

H	I	$(H$	$\wedge$	$I)$	$\rightarrow$	H
		$\mathcal{A}$	$\wedge$	$\mathcal{B}$		
T	T	T	**T**	T		T
T	F	T	**F**	F		T
F	T	F	**F**	T		F
F	F	F	**F**	F		F

The entire sentence is a conditional $\mathcal{A}\rightarrow\mathcal{B}$ with $(H \wedge I)$ as $\mathcal{A}$ and with H as $\mathcal{B}$. On the second row, for example, $(H \wedge I)$ is false and H is true. Since a conditional is true when the antecedent is false, we write a T in the second row underneath the conditional symbol. We continue for the other three rows and get this:

H	I	$(H$	$\wedge$	$I)$	$\rightarrow$	H
			$\mathcal{A}$		$\rightarrow$	$\mathcal{B}$
T	T		T		**T**	T
T	F		F		**T**	T
F	T		F		**T**	F
F	F		F		**T**	F

The column of Ts underneath the conditional tells us that the sentence $(H \wedge I) \rightarrow H$ is true regardless of the truth values of H and I. They can be true or false in any combination, and the compound sentence still comes out true. It is crucial that we have considered all of the possible combinations. If we only had a two-line truth table, we could not be sure that the sentence was not false for some other combination of truth values.

In this example, we have not repeated all of the entries in every successive table. When actually writing truth tables on paper, however, it is impractical to erase whole columns or rewrite the whole table for every step. Although it is more crowded, the truth table can be written in this way:

H	I	$(H$	$\wedge$	$I)$	$\rightarrow$	H
T	T	T	T	T	T	T
T	F	T	F	F	T	T
F	T	F	F	T	T	F
F	F	F	F	F	T	F

Most of the columns underneath the sentence are only there for bookkeeping purposes. When you become more adept with truth tables, you will probably no longer need to copy over the columns for each of the sentence letters. In any case, the truth value of the sentence on each row is just the column underneath the *main connective* (see p. 16) of the sentence, in this case, the column underneath the conditional.

A COMPLETE TRUTH TABLE is a table that gives all the possible interpretations for a sentence or set of sentences in SL. It has a row for all the possible combinations of T and F for all of the sentence letters. The size of the complete truth table depends on the number of different sentence letters in the table. A sentence that contains only one sentence letter requires only two rows, as in the characteristic truth table for negation. This is true even if the same letter is repeated many times, as in this sentence:

$$[(C \leftrightarrow C) \rightarrow C] \wedge \neg(C \rightarrow C).$$

The complete truth table requires only two lines because there are only two possibilities: C can be true, or it can be false. A single sentence letter can never be marked both T and F on the same row. The truth table for this sentence looks like this:

C	$[($	C	$\leftrightarrow$	C	$)$	$\rightarrow$	C	$]$	$\wedge$	$\neg$	$($	C	$\rightarrow$	C	$)$
T		T	T	T		T	T		**F**	F		T	T	T	
F		F	T	F		F	F		**F**	F		F	T	F	

Looking at the column underneath the main connective, we see that the sentence is false on both rows of the table; i.e., it is false regardless of whether C is true or false.

A sentence that contains two sentence letters requires four lines for a complete truth table, as in the characteristic truth tables and the table for $(H \wedge I) \rightarrow I$.

A sentence that contains three sentence letters requires eight lines. For example:

M	N	P	M	$\wedge$	$(N$	$\vee$	$P)$
T	T	T	T	**T**	T	T	T
T	T	F	T	**T**	T	T	F
T	F	T	T	**T**	F	T	T
T	F	F	T	**F**	F	F	F
F	T	T	F	**F**	T	T	T
F	T	F	F	**F**	T	T	F
F	F	T	F	**F**	F	T	T
F	F	F	F	**F**	F	F	F

From this table, we know that the sentence $M \wedge (N \vee P)$ might be true or false, depending on the truth values of M, N, and P.

A complete truth table for a sentence that contains four different sentence letters requires 16 lines. For five letters, 32 lines are required. For six letters, 64 lines, and so on. To be perfectly general: If a complete truth table has n different sentence letters, then it must have 2^n rows.

In order to fill in the columns of a complete truth table, begin with the right-most sentence letter and alternate Ts and Fs. In the next column to the left, write two Ts, write two Fs, and repeat. For the third sentence letter, write four Ts followed by four Fs. This yields an eight line truth table like the one above. For a 16 line truth table, the next column of sentence letters should have eight Ts followed by eight Fs. For a 32 line table, the next column would have 16 Ts followed by 16 Fs. And so on.

2.3 Using Truth Tables

A complete truth table shows us every possible combination of truth assignments on the sentence letters. It tells us every possible way sentences can relate to truth. We can use this to discover all sorts of logical properties of sentences and sets of sentences.

Tautologies, contradictions, and contingent sentences

We defined a tautology as a statement that must be true as a matter of logic, no matter how the world is. A statement like "Either it is raining or it is not raining" is always true, no matter what

the weather is like outside. Something similar goes on in truth tables. With a complete truth table, we consider all of the ways that the world might be. Each line of the truth table corresponds to a way the world might be. This means that if the sentence is true on every line of a complete truth table, then it is true as a matter of logic, regardless of what the world is like.

We can use this fact to create a test for whether a sentence is a tautology: if the column under the main connective of a sentence is a T on every row, the sentence is a tautology. Not every tautology in English will correspond to a tautology in SL. The sentence "All bachelors are unmarried" is a tautology in English, but we cannot represent it as a tautology in SL, because it just translates as a single sentence letter, like B. On the other hand, if something is a tautology in SL, it will also be a tautology in English. No matter how you translate $A \vee \neg A$, if you translate the As consistently, the statement will be a tautology.

Rather than thinking of complete truth tables as an imperfect test for the English notion of a tautology, we can define a separate notion of a tautology in SL based on truth tables. A statement is a SEMANTIC TAUTOLOGY IN SL if and only if the column under the main connective in the complete truth table for the sentence contains only Ts. This is actually the semantic definition of a tautology in SL, because it uses truth tables. Later we will create a separate, syntactic definition and show that it is equivalent to the semantic definition. We will be doing the same thing for all the concepts defined in this section.

Conversely, we defined a contradiction as a sentence that is false no matter how the world is. This means we can define a SEMANTIC CONTRADICTION IN SL as a sentence that has only Fs in the column under them main connective of its complete truth table. Again, this is the semantic definition of a contradiction.

Finally, a sentence is contingent if it is sometimes true and sometimes false. Similarly, a sentence is SEMANTICALLY CONTINGENT IN SL if and only if its complete truth table for has both Ts and Fs under the main connective.

From the truth tables in the previous section, we know that $(H \wedge I) \rightarrow H$ is a tautology (p. 22), that $[(C \leftrightarrow C) \rightarrow C] \wedge \neg(C \rightarrow C)$ is a contradiction (p. 23), and that $M \wedge (N \vee P)$ is contingent (p. 23).

Logical equivalence

Two sentences are logically equivalent in English if they have the same truth value as a matter of logic. Once again, we can use truth tables to define a similar property in SL: Two sentences are SEMANTICALLY LOGICALLY EQUIVALENT IN SL if they have the same truth value on every row of a complete truth table.

Consider the sentences $\neg(A \vee B)$ and $\neg A \wedge \neg B$. Are they logically equivalent? To find out, we construct a truth table.

A	B	$\neg$	$(A$	$\vee$	$B)$	$\neg$	A	$\wedge$	$\neg$	B
T	T	**F**	T	T	T	F	T	**F**	F	T
T	F	**F**	T	T	F	F	T	**F**	T	F
F	T	**F**	F	T	T	T	F	**F**	F	T
F	F	**T**	F	F	F	T	F	**T**	T	F

Look at the columns for the main connectives; negation for the first sentence, conjunction for the second. On the first three rows, both are F. On the final row, both are T. Since they match on every row, the two sentences are logically equivalent.

Consistency

A set of sentences in English is consistent if it is logically possible for them all to be true at once. This means that a sentence is SEMANTICALLY CONSISTENT IN SL if and only if there is at least one line of a complete truth table on which all of the sentences are true. It is semantically inconsistent otherwise.

Validity

Logic is the study of argument, so the most important use of truth tables is to test the validity of arguments. An argument in English is valid if it is logically impossible for the premises to be true and for the conclusion to be false at the same time. So we can define an argument as SEMANTICALLY VALID IN SL if there is no row of a complete truth table on which the premises are all marked "T" and the conclusion is marked "F." An argument is invalid if there is such a row.

Consider this argument:

1. $\neg L \rightarrow (J \vee L)$
2. $\neg L$

$\therefore$ J

Is it valid? To find out, we construct a truth table.

J	L	$\neg$	L	$\rightarrow$	$(J$	$\vee$	$L)$	$\neg$	L	J
T	T	F	T	**T**	T	T	T	**F**	T	**T**
T	F	T	F	**T**	T	T	F	**T**	F	**T**
F	T	F	T	**T**	F	T	T	**F**	T	**F**
F	F	T	F	**F**	F	F	F	**T**	F	**F**

Yes, the argument is valid. The only row on which both the premises are T is the second row, and on that row the conclusion is also T.

In Chapters 1 and 2 we used the three dots $\therefore$ to represent an inference in English. We used this symbol to represent any kind of inference. The truth table method gives us a more specific notion of a valid inference. We will call this semantic entailment and represent it using a new symbol, $\models$, called the "double turnstile." The $\models$ is like the $\therefore$, except for arguments verified by truth tables. When you use the double turnstile, you write the premises as a set, using curly brackets, { and }, which mathematicians use in set theory. The argument above would be written $\{\neg L \rightarrow (J \vee L), \neg L\} \models J$.

More formally, we can define the double turnstile this way: $\{\mathcal{A}_1 \ldots \mathcal{A}_n\} \models \mathcal{B}$ if and only if there is no truth value assignment for which $\mathcal{A}_1 \ldots \mathcal{A}_n$ are true and $\mathcal{B}$ is false. Put differently, it means that $\mathcal{B}$ is true for any and all truth value assignments for which $\mathcal{A}_1 \ldots \mathcal{A}_n$ are true.

We can also use the double turnstile to represent other logical notions. Since a tautology is always true, it is like the conclusion of a valid argument with no premises. The string $\models \mathcal{C}$ means that $\mathcal{C}$ is true for all truth value assignments. This is equivalent to saying that the sentence is entailed by anything. We can represent logical equivalence by writing the double turnstile in both directions: $\mathcal{A} \dashv\models \mathcal{B}$ For instance, if we want to point out that the sentence $A \wedge B$ is equivalent to $B \wedge A$ we would write this: $A \wedge B \dashv\models B \wedge A$.

2.4 Partial Truth Tables

In order to show that a sentence is a tautology, we need to show that it is T on every row. So we need a complete truth table. To show that a sentence is *not* a tautology, however, we only need one line: a line on which the sentence is F. Therefore, in order to show that something is not a tautology, it is enough to provide a one-line *partial truth table*—regardless of how many sentence letters the sentence might have in it.

Consider, for example, the sentence $(U \wedge T) \rightarrow (S \wedge W)$. We want to show that it is *not* a tautology by providing a partial truth table. We fill in F for the entire sentence. The main connective of the sentence is a conditional. In order for the conditional to be false, the antecedent must be true (T) and the consequent must be false (F). So we fill these in on the table:

S	T	U	W	$(U$	$\wedge$	$T)$	$\rightarrow$	$(S$	$\wedge$	$W)$
					T		**F**		F	

In order for the $(U \wedge T)$ to be true, both U and T must be true.

S	T	U	W	$(U$	$\wedge$	$T)$	$\rightarrow$	$(S$	$\wedge$	$W)$
	T	T		T	T	T	**F**		F	

Now we just need to make $(S \wedge W)$ false. To do this, we need to make at least one of S and W false. We can make both S and W false if we want. All that matters is that the whole sentence turns out false on this line. Making an arbitrary decision, we finish the table in this way:

S	T	U	W	$(U$	$\wedge$	$T)$	$\rightarrow$	$(S$	$\wedge$	$W)$
F	T	T	F	T	T	T	**F**	F	F	F

Showing that something is a contradiction requires a complete truth table. Showing that something is *not* a contradiction requires only a one-line partial truth table, where the sentence is true on that one line.

A sentence is contingent if it is neither a tautology nor a contradiction. So showing that a sentence is contingent requires a *two-line* partial truth table: The sentence must be true on one line and false on the other. For example, we can show that the sentence above is contingent with this truth table:

S	T	U	W	$(U$	$\wedge$	$T)$	$\rightarrow$	$(S$	$\wedge$	$W)$
F	T	T	F	T	T	T	**F**	F	F	F
F	T	F	F	F	F	T	**T**	F	F	F

Note that there are many combinations of truth values that would have made the sentence true, so there are many ways we could have written the second line.

Showing that a sentence is *not* contingent requires providing a complete truth table, because it requires showing that the sentence is a tautology or that it is a contradiction. If you do not know whether a particular sentence is contingent, then you do not know whether you will need a complete or partial truth table. You can always start working on a complete truth table. If you complete rows that show the sentence is contingent, then you can stop. If not, then complete the truth table. Even though two carefully selected rows will show that a contingent sentence is contingent, there is nothing wrong with filling in more rows.

Showing that two sentences are logically equivalent requires providing a complete truth table. Showing that two sentences are *not* logically equivalent requires only a one-line partial truth table: Make the table so that one sentence is true and the other false.

Showing that a set of sentences is consistent requires providing one row of a truth table on which all of the sentences are true. The rest of the table is irrelevant, so a one-line partial truth table

Property	Present	Absent
tautology	complete truth table	one-line partial truth table
contradiction	complete truth table	one-line partial truth table
contingent	two-line partial truth table	complete truth table
equivalent	complete truth table	one-line partial truth table
consistent	one-line partial truth table	complete truth table
valid	complete truth table	one-line partial truth table

Table 2.2: Complete or partial truth tables to test for different properties

will do. Showing that a set of sentences is inconsistent, on the other hand, requires a complete truth table: You must show that on every row of the table at least one of the sentences is false.

Showing that an argument is valid requires a complete truth table. Showing that an argument is *invalid* only requires providing a one-line truth table: If you can produce a line on which the premises are all true and the conclusion is false, then the argument is invalid.

Table 2.2 summarizes when a complete truth table is required and when a partial truth table will do.

2.5 Expressive Completeness

We could leave the biconditional ($\leftrightarrow$) out of the language. If we did that, we could still write "$A \leftrightarrow B$" so as to make sentences easier to read, but that would be shorthand for $(A \rightarrow B) \wedge (B \rightarrow A)$. The resulting language would be formally equivalent to SL, since $A \leftrightarrow B$ and $(A \rightarrow B) \wedge (B \rightarrow A)$ are logically equivalent in SL. If we valued formal simplicity over expressive richness, we could replace more of the connectives with notational conventions and still have a language equivalent to SL.

There are a number of equivalent languages with only two connectives. You could do logic with only the negation and the material conditional. Alternately you could just have the negation and the disjunction. You will be asked to prove that these things are true in the last problem set. You could even have a language with only one connective, if you designed the connective right. The *Sheffer stroke* is a logical connective with the following characteristic truth table:

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{A}\|\mathcal{B}$
T	T	F
T	F	T
F	T	T
F	F	T

The Sheffer stroke has the unique property that it is the only connective you need to have a complete system of logic. You will be asked to prove that this is true in the last problem set also.

2.6 Recursive Definition of Truth in SL

In an earlier section, we gave a recursive definition of what it meant to be a sentence in SL. We will also end this chapter with a recursive definition that summarizes the material in the chapter. In this case, we are going to give a recursive definition of truth in SL. This treatment is original to Magnus, *For All X*'s first author..

Formally, what we want is a function that assigns a 1 or 0 to each of the sentences of SL. We can interpret this function as a definition of truth for SL if it assigns 1 to all of the true sentences of SL and 0 to all of the false sentences of SL. Call this function "v" (for "valuation"). We want v to a be a function such that for any sentence $\mathcal{A}$, $v(\mathcal{A}) = 1$ if $\mathcal{A}$ is true and $v(\mathcal{A}) = 0$ if $\mathcal{A}$ is false. Recall that the recursive definition of a wff for SL had two stages: The first step said that atomic sentences (solitary sentence letters) are wffs. The second stage allowed for wffs to be constructed out of more basic wffs. There were clauses of the definition for all of the sentential connectives. For example, if $\mathcal{A}$ is a wff, then $\neg\mathcal{A}$ is a wff. Our strategy for defining the truth function, v, will also be in two steps. The first step will handle truth for atomic sentences; the second step will handle truth for compound sentences.

Truth in SL

How can we define truth for an atomic sentence of SL? Consider, for example, the sentence M. Without an interpretation, we cannot say whether M is true or false. It might mean anything. If we use M to symbolize "The moon orbits the Earth", then M is true. If we use M to symbolize "The moon is a giant turnip", then M is false.

Moreover, the way you would discover whether or not M is true depends on what M means. If M means "It is Monday," then you would need to check a calendar. If M means "Jupiter's moon Io has significant volcanic activity," then you would need to check an astronomy text—and astronomers know because they sent satellites to observe Io.

When we give a symbolization key for SL, we provide an interpretation of the sentence letters that we use. The key gives an English language sentence for each sentence letter that we use. In this way, the interpretation specifies what each of the sentence letters *means*. However, this is not enough to determine whether or not that sentence is true. The sentences about the moon, for instance, require that you know some rudimentary astronomy. Imagine a small child who became convinced that the moon is a giant turnip. She could understand what the sentence "The moon is a giant turnip" means, but mistakenly think that it was true.

Consider another example: If M means "It is morning now", then whether it is true or not depends on when you are reading this. I know what the sentence means, but—since I do not know

when you will be reading this—I do not know whether it is true or false.

So an interpretation alone does not determine whether a sentence is true or false. Truth or falsity depends also on what the world is like. If M meant "The moon is a giant turnip" and the real moon were a giant turnip, then M would be true. To put the point in a general way, truth or falsity is determined by an interpretation *plus* a way that the world is.

$$\text{INTERPRETATION + STATE OF THE WORLD} \Longrightarrow \text{TRUTH/FALSITY}$$

In providing a logical definition of truth, we will not be able to give an account of how an atomic sentence is made true or false by the world. Instead, we will introduce a *truth value assignment*. Formally, this will be a function that tells us the truth value of all the atomic sentences. Call this function "a" (for "assignment"). We define a for all sentence letters $\mathcal{P}$, such that

$$a(\mathcal{P}) = \begin{cases} 1 & \text{if } \mathcal{P} \text{ is true,} \\ 0 & \text{otherwise.} \end{cases}$$

This means that a takes any sentence of SL and assigns it either a one or a zero; one if the sentence is true, zero if the sentence is false. The details of the function a are determined by the meaning of the sentence letters together with the state of the world. If D means "It is dark outside", then $a(D) = 1$ at night or during a heavy storm, while $a(D) = 0$ on a clear day.

You can think of a as being like a row of a truth table. Whereas a truth table row assigns a truth value to a few atomic sentences, the truth value assignment assigns a value to every atomic sentence of SL. There are infinitely many sentence letters, and the truth value assignment gives a value to each of them. When constructing a truth table, we only care about sentence letters that affect the truth value of sentences that interest us. As such, we ignore the rest. Strictly speaking, every row of a truth table gives a *partial* truth value assignment.

It is important to note that the truth value assignment, a, is not part of the language SL. Rather, it is part of the mathematical machinery that we are using to describe SL. It encodes which atomic sentences are true and which are false.

We now define the truth function, v, using the same recursive structure that we used to define a wff of SL.

1. If $\mathcal{A}$ is a sentence letter, then $v(\mathcal{A}) = a(\mathcal{A})$.
2. If $\mathcal{A}$ is $\neg\mathcal{B}$ for some sentence $\mathcal{B}$, then
$$v(\mathcal{A}) = \begin{cases} 1 & \text{if } v(\mathcal{B}) = 0, \\ 0 & \text{otherwise.} \end{cases}$$
3. If $\mathcal{A}$ is $(\mathcal{B} \wedge \mathcal{C})$ for some sentences $\mathcal{B}, \mathcal{C}$, then
$$v(\mathcal{A}) = \begin{cases} 1 & \text{if } v(\mathcal{B}) = 1 \text{ and } v(\mathcal{C}) = 1, \\ 0 & \text{otherwise.} \end{cases}$$

It might seem as if this definition is circular, because it uses the word "and" in trying to define "and." Notice, however, that this is not a definition of the English word "and"; it is a definition of truth for sentences of SL containing the logical symbol "$\wedge$." We define truth for object language sentences containing the symbol "$\wedge$" using the metalanguage word "and." There is nothing circular about that.

4. If $\mathcal{A}$ is $(\mathcal{B} \vee \mathcal{C})$ for some sentences $\mathcal{B}, \mathcal{C}$, then

$$v(\mathcal{A}) = \begin{cases} 0 & \text{if } v(\mathcal{B}) = 0 \text{ and } v(\mathcal{C}) = 0, \\ 1 & \text{otherwise.} \end{cases}$$

5. If $\mathcal{A}$ is $(\mathcal{B} \rightarrow \mathcal{C})$ for some sentences $\mathcal{B}, \mathcal{C}$, then

$$v(\mathcal{A}) = \begin{cases} 0 & \text{if } v(\mathcal{B}) = 1 \text{ and } v(\mathcal{C}) = 0, \\ 1 & \text{otherwise.} \end{cases}$$

6. If $\mathcal{A}$ is $(\mathcal{B} \leftrightarrow \mathcal{C})$ for some sentences $\mathcal{B}, \mathcal{C}$, then

$$v(\mathcal{A}) = \begin{cases} 1 & \text{if } v(\mathcal{B}) = v(\mathcal{C}), \\ 0 & \text{otherwise.} \end{cases}$$

Since the definition of v has the same structure as the definition of a wff, we know that v assigns a value to *every* wff of SL. Since the sentences of SL and the wffs of SL are the same, this means that v returns the truth value of every sentence of SL.

Truth in SL is always truth *relative to* some truth value assignment, because the definition of truth for SL does not say whether a given sentence is true or false. Rather, it says how the truth of that sentence relates to a truth value assignment.

Key Terms

Complete truth table

Interpretation

Logical symbol

Nonlogical symbol

Semantically consistent in SL

Semantically contingent in SL

Semantically logically equivalent in SL

Semantically valid in SL

Semantic contradiction in SL

Semantic tautology in SL

Truth-functional connective

Truth assignment

Bibliography