forallχ

Agnes Scott College 2019-2020

P.D. Magnus
University at Albany, State University of New York

Modified for the Cambridge course by:
Tim Button
University of Cambridge

Modified for the Agnes Scott College course by:
Jared Millson
Agnes Scott College

P.D. Magnus would like to thank the people who made this project possible. Notable among these are Cristyn Magnus, who read many early drafts; Aaron Schiller, who was an early adopter and provided considerable, helpful feedback; and Bin Kang, Craig Erb, Nathan Carter, Wes McMichael, Selva Samuel, Dave Krueger, Brandon Lee, and the students of Introduction to Logic, who detected various errors in previous versions of the book.

Tim Button would like to thank P.D. Magnus for his extraordinary act of generosity, in making `forall`χ available to everyone. Thanks also to Alfredo Manfredini Böhm, Sam Brain, Felicity Davies, Emily Dyson, Phoebe Hill, Richard Jennings, Justin Niven, and Igor Stojanovic for noticing errata in earlier versions.

Jared would like to thank P.D. Magnus for his extraordinary act of generosity, in making `forall`χ available to everyone. Thanks also Tim Button for is wonderful development of that work.

Typesetting was carried out entirely in LaTeX2ε. The style for typesetting proofs is based on fitch.sty (v0.4) by Peter Selinger, University of Ottawa.

Contents

Chapter 1

Key notions

Arguments 1

Logic is the business of evaluating arguments; sorting the good from the bad.

In everyday language, we sometimes use the word 'argument' to talk about belligerent shouting matches. Logic is not concerned with such teeth-gnashing and hair-pulling. They are not arguments, in our sense; they are disagreements.

An argument, as we shall understand it, is something more like this:

> It is raining heavily.
> If you do not take an umbrella, you will get soaked.
> So: You should take an umbrella.

We here have a series of sentences. The word 'So' on the third line indicates that the final sentence expresses the *conclusion* of the argument. The two sentences before that express *premises* of the argument. If you believe the premises, then the argument (perhaps) provides you with a reason to believe the conclusion.

This is the sort of thing that logicians are interested in. We shall say that an argument is any collection of premises, together with a conclusion.

In the example just given, we used individual sentences to express both of the argument's premises, and we used a third sentence to express the argument's conclusion. Many arguments are expressed in this way. But a single sentence can contain a complete argument. Consider:

> I was wearing my sunglasses; so it must have been sunny.

This argument has one premise followed by a conclusion.

Many arguments start with premises, and end with a conclusion. But not all of them. The argument with which this section began might equally have been presented with the conclusion at the beginning, like so:

> You should take an umbrella. After all, it is raining heavily. And if you do not take an umbrella, you will get soaked.

Equally, it might have been presented with the conclusion in the middle:

> It is raining heavily. Accordingly, you should take an umbrella, given that if you do not take an umbrella, you will get soaked.

When approaching an argument, we want to know whether or not the conclusion follows from the premises. So the first thing to do is to separate out the conclusion from the premises. As a guideline, the following words are often used to indicate an argument's conclusion:

so, therefore, hence, thus, accordingly, consequently

And these expressions often indicate that we are dealing with a premise, rather than a conclusion

since, because, given that

But in analysing an argument, there is no substitute for a good nose.

Practice exercises

At the end of some sections, there are problems that review and explore the material covered in the chapter. There is no substitute for actually working through some problems, because logic is more about a way of thinking than it is about memorising facts.

Highlight the phrase which expresses the conclusion of each of these arguments:

1. It is sunny. So I should take my sunglasses.
2. It must have been sunny. I did wear my sunglasses, after all.
3. No one but you has had their hands in the cookie-jar. And the scene of the crime is littered with cookie-crumbs. You're the culprit!
4. Miss Scarlett and Professor Plum were in the study at the time of the murder. And Reverend Green had the candlestick in the ballroom, and we know that there is no blood on his hands. Hence Colonel Mustard did it in the kitchen with the lead-piping. Recall, after all, that the gun had not been fired.

Valid arguments 2

In §1, we gave a very permissive account of what an argument is. To see just how permissive it is, consider the following:

> There is a bassoon-playing dragon in the *Cathedra Romana*.
> So: Salvador Dali was a poker player.

We have been given a premise and a conclusion. So we have an argument. Admittedly, it is a *terrible* argument. But it is still an argument.

2.1 Two ways that arguments can go wrong

It is worth pausing to ask what makes the argument so weak. In fact, there are two sources of weakness. First: the argument's (only) premise is obviously false. The Pope's throne is only ever occupied by a hat-wearing man. Second: the conclusion does not follow from the premise of the argument. Even if there were a bassoon-playing dragon in the Pope's throne, we would not be able to draw any conclusion about Dali's predilection for poker.

What about the main argument discussed in §1? The premises of this argument might well be false. It might be sunny outside; or it might be that you can avoid getting soaked without taking an umbrella. But even if both premises were true, it does not necessarily show you that you should take an umbrella. Perhaps you enjoy walking in the rain, and you would like to get soaked. So, even if both premises were true, the conclusion might nonetheless be false.

The general point is as follows. For any argument, there are two ways that it might go wrong:

- One or more of the premises might be false.
- The conclusion might not follow from the premises.

To determine whether or not the premises of an argument are true is often a very important matter. But that is normally a task best left to experts in the field: as it might be, historians, scientists, or whomever. In our role as *logicians*, we are more concerned with arguments *in general*. So we are (usually) more concerned with the second way in which arguments can go wrong.

So: we are interested in whether or not a conclusion *follows from* some premises. Don't, though, say that the premises *infer* the conclusion. Entailment is a relation between premises and conclusions; inference is something we do. (So if you want to mention inference when the conclusion follows from the premises, you could say that *one may infer* the conclusion from the premises.)

2.2 Validity

As logicians, we want to be able to determine when the conclusion of an argument follows from the premises. One way to put this is as follows. We want to know whether, if all the premises were true, the conclusion would also have to be true. This motivates a definition:

> An argument is VALID if and only if it is impossible for all of the premises to be true and the conclusion false.

The crucial thing about a valid argument is that it is impossible for the premises to be true whilst the conclusion is false. Consider this example:

Oranges are either fruits or musical instruments.
Oranges are not fruits.
So: Oranges are musical instruments.

The conclusion of this argument is ridiculous. Nevertheless, it follows from the premises. *If* both premises were true, *then* the conclusion just has to be true. So the argument is valid.

This highlights that valid arguments do not need to have true premises or true conclusions. Conversely, having true premises and a true conclusion is not enough to make an argument valid. Consider this example:

London is in England.
Beijing is in China.
So: Paris is in France.

The premises and conclusion of this argument are, as a matter of fact, all true. But the argument is invalid. If Paris were to declare independence from the rest of France, then the conclusion would be false, even though both of the premises would remain true. Thus, it is *possible* for the premises of this argument to be true and the conclusion false. The argument is therefore invalid.

The important thing to remember is that validity is not about the actual truth or falsity of the sentences in the argument. It is about whether it is *possible* for all the premises to be true and the conclusion false. Nonetheless, we shall say that an argument is SOUND if and only if it is both valid and all of its premises are true.

2.3 Inductive arguments

Many good arguments are invalid. Consider this one:

In January 1997, it rained in London.
In January 1998, it rained in London.
In January 1999, it rained in London.
In January 2000, it rained in London.
So: It rains every January in London.

This argument generalises from observations about several cases to a conclusion about all cases. Such arguments are called INDUCTIVE arguments. The argument could be made stronger by adding additional premises before drawing the conclusion: In January 2001, it rained in London; In January 2002....

But, however many premises of this form we add, the argument will remain invalid. Even if it has rained in London in every January thus far, it remains *possible* that London will stay dry next January.

The point of all this is that inductive arguments—even good inductive arguments—are not (deductively) valid. They are not *watertight*. Unlikely though it might be, it is *possible* for their conclusion to be false, even when all of their premises are true. In this book, we shall set aside (entirely) the question of what makes for a good inductive argument. Our interest is simply in sorting the (deductively) valid arguments from the invalid ones.

Practice exercises

A. Which of the following arguments are valid? Which are invalid?

1. Socrates is a man.
2. All men are carrots.

So: Therefore, Socrates is a carrot.

1. Abe Lincoln was either born in Illinois or he was once president.
2. Abe Lincoln was never president.

So: Abe Lincoln was born in Illinois.

1. If I pull the trigger, Abe Lincoln will die.
2. I do not pull the trigger.

So: Abe Lincoln will not die.

1. Abe Lincoln was either from France or from Luxemborg.
2. Abe Lincoln was not from Luxemborg.

So: Abe Lincoln was from France.

1. If the world were to end today, then I would not need to get up tomorrow morning.
2. I will need to get up tomorrow morning.

So: The world will not end today.

1. Joe is now 19 years old.
2. Joe is now 87 years old.

So: Bob is now 20 years old.

B. Could there be:

1. A valid argument that has one false premise and one true premise?
2. A valid argument that has only false premises?
3. A valid argument with only false premises and a false conclusion?
4. A sound argument with a false conclusion?
5. An invalid argument that can be made valid by the addition of a new premise?
6. A valid argument that can be made invalid by the addition of a new premise?

In each case: if so, give an example; if not, explain why not.

Other logical notions 3

In §2, we introduced the idea of a valid argument. We will want to introduce some more ideas that are important in logic.

3.1 Truth values

As we said in §1, arguments consist of premises and a conclusion. Note that many kinds of English sentence cannot be used to express premises or conclusions of arguments. For example:

- **Questions**, e.g. 'are you feeling sleepy?'
- **Imperatives**, e.g. 'Wake up!'
- **Exclamations**, e.g. 'Ouch!'

The common feature of these three kinds of sentence is that they are not *assertoric*: they cannot be true or false. It does not even make sense to ask whether a *question* is true (it only makes sense to ask whether the *answer* to a question is true).

The general point is that, the premises and conclusion of an argument must be capable of having a TRUTH VALUE. And the two truth values that concern us are just True and False.

3.2 Consistency

Consider these two sentences:

B1. Jane's only brother is shorter than her.
B2. Jane's only brother is taller than her.

Logic alone cannot tell us which, if either, of these sentences is true. Yet we can say that *if* the first sentence (B1) is true, *then* the second sentence (B2) must be false. And if B2 is true, then B1 must be false. It is impossible that both sentences are true together. These sentences are inconsistent with each other. And this motivates the following definition:

> Sentences are JOINTLY CONSISTENT if and only if it is possible for them all to be true together.

Conversely, B1 and B2 are *jointly inconsistent*.

We can ask about the consistency of any number of sentences. For example, consider the following four sentences:

G1. There are at least four giraffes at the wild animal park.
G2. There are exactly seven gorillas at the wild animal park.
G3. There are not more than two martians at the wild animal park.
G4. Every giraffe at the wild animal park is a martian.

G1 and G4 together entail that there are at least four martian giraffes at the park. This conflicts with G3, which implies that there are no more than two martian giraffes there. So the sentences G1–G4 are jointly inconsistent. They cannot all be true together. (Note that the sentences G1, G3 and G4 are jointly inconsistent. But if sentences are already jointly inconsistent, adding an extra sentence to the mix will not make them consistent!)

3.3 Necessity and contingency

In assessing arguments for validity, we care about what would be true *if* the premises were true. But some sentences just *must* be true. Consider these sentences:

1. It is raining.
2. Either it is raining here, or it is not.
3. It is both raining here and not raining here.

In order to know if sentence 1 is true, you would need to look outside or check the weather channel. It might be true; it might be false.

Sentence 2 is different. You do not need to look outside to know that it is true. Regardless of what the weather is like, it is either raining or it is not. That is a NECESSARY TRUTH.

Equally, you do not need to check the weather to determine whether or not sentence 3 is true. It must be false, simply as a matter of logic. It might be raining here and not raining across town; it might be raining now but stop raining even as you finish this sentence; but it is impossible for it to be both raining and not raining in the same place and at the same time. So, whatever the world is like, it is not both raining here and not raining here. It is a NECESSARY FALSEHOOD.

Something which is capable of being true or false, but which is neither necessarily true nor necessarily false, is CONTINGENT.

Practice exercises

A. For each of the following: Is it necessarily true, necessarily false, or contingent?

1. Caesar crossed the Rubicon.
2. Someone once crossed the Rubicon.
3. No one has ever crossed the Rubicon.
4. If Caesar crossed the Rubicon, then someone has.
5. Even though Caesar crossed the Rubicon, no one has ever crossed the Rubicon.
6. If anyone has ever crossed the Rubicon, it was Caesar.

B. Look back at the sentences G1–G4 in this section (about giraffes, gorillas and martians in the wild animal park), and consider each of the following:

1. G2, G3, and G4
2. G1, G3, and G4
3. G1, G2, and G4
4. G1, G2, and G3

Which are jointly consistent? Which are jointly inconsistent?

C. Could there be:

1. A valid argument, the conclusion of which is necessarily false?
2. An invalid argument, the conclusion of which is necessarily true?
3. Jointly consistent sentences, one of which is necessarily false?
4. Jointly inconsistent sentences, one of which is necessarily true?

In each case: if so, give an example; if not, explain why not.

Chapter 2

Truth-functional logic

First steps to symbolisation 4

4.1 Validity in virtue of form

Consider this argument:

Jenny is raining outside.
If it is raining outside, then Jenny is miserable.
So: Jenny is miserable.

and another argument:

Jenny is an anarcho-syndicalist.
If Jenny is an anarcho-syndicalist, then Dipan is an avid reader of Tolstoy.
So: Dipan is an avid reader of Tolstoy.

Both arguments are valid, and there is a straightforward sense in which we can say that they share a common structure. We might express the structure thus:

A
If A, then C
So: C

This looks like an excellent argument *structure*. Indeed, surely any argument with this *structure* will be valid. And this is not the only good argument structure. Consider an argument like:

Jenny is either happy or sad.
Jenny is not happy.
So: Jenny is sad.

Again, this is a valid argument. The structure here is something like:

A or B
not-A
So: B

A superb structure! And here is a final example:

It's not the case that Jim both studied hard and acted in lots of plays.
Jim studied hard
So: Jim did not act in lots of plays.

This valid argument has a structure which we might represent thus:

not-(A and B)
A
So: not-B

The examples illustrate an important idea, which we might describe as *validity in virtue of form*. The validity of the arguments just considered has nothing very much to do with the meanings of English expressions like 'Jenny is miserable', 'Dipan is an avid reader of Tolstoy', or 'Jim acted in lots of plays'. If it has to do with meanings at all, it is with the meanings of phrases like 'and', 'or', 'not,' and 'if..., then...'.

In this chapter, we are going to develop a formal language which allows us to symbolise many arguments in such a way as to show that they are valid in virtue of their form. That language will be *truth-functional logic*, or TFL.

4.2 Validity for special reasons

There are plenty of arguments that are valid, but not for reasons relating to their form. Take an example:

Juanita is a vixen
So: Juanita is a fox

It is impossible for the premise to be true and the conclusion false. So the argument is valid. But the validity is not related to the form of the argument. Here is an invalid argument with the same form:

Juanita is a vixen
So: Juanita is a cathedral

This might suggest that the validity of the first argument *is* keyed to the meaning of the words 'vixen' and 'fox'. But, whether or not that is right, it is not simply the *shape* of the argument that makes it valid. Equally, consider the argument:

The sculpture is green all over.
So: The sculpture is not red all over.

Again, it seems impossible for the premise to be true and the conclusion false, for nothing can be both green all over and red all over. So the argument is valid. But here is an invalid argument with the same form:

The sculpture is green all over.
So: The sculpture is not shiny all over.

The argument is invalid, since it is possible to be green all over and shiny all over. (I might paint my nails with an elegant shiny green varnish.) Plausibly, the validity of the first argument is keyed to the way that colours (or colour-words) interact. But, whether or not that is right, it is not simply the *shape* of the argument that makes it valid.

The important moral can be stated as follows. *At best, TFL will help us to understand arguments that are valid due to their form.*

4.3 Atomic sentences

I started isolating the form of an argument, in §4.1, by replacing *subsentences* of sentences with individual letters. Thus in the first example of this section, 'it is raining outside' is a subsentence of 'If it is raining outside, then Jenny is miserable', and we replaced this subsentence with 'A'.

Our artificial language, TFL, pursues this idea absolutely ruthlessly. We start with some *atomic sentences*. These will be the basic building blocks out of which more complex sentences are built. We will use uppercase italic letters for atomic sentences of TFL. There are only twenty-six letters of the alphabet, but there is no limit to the number of atomic sentences that we might want to consider. By adding subscripts to letters, we obtain new atomic sentences. So, here are five different atomic sentences of TFL:

$$A, P, P_1, P_2, A_{234}$$

We shall use atomic sentence to represent, or symbolise, certain English sentences. To do this, we provide a SYMBOLISATION KEY, such as the following:

> A: It is raining outside
> C: Jenny is miserable

In doing this, we are not fixing this symbolisation *once and for all*. We are just saying that, for the time being, we shall think of the atomic sentence of TFL, 'A', as symbolising the English sentence 'It is raining outside', and the atomic sentence of TFL, 'C', as symbolising the English sentence 'Jenny is miserable. Later, when we are dealing with different sentences or different arguments, we can provide a new symbolisation key; as it might be:

> A: Jenny is an anarcho-syndicalist
> C: Dipan is an avid reader of Tolstoy

But it is important to understand that whatever structure an English sentence might have is lost when it is symbolised by an atomic sentence of TFL. From the point of view of TFL, an atomic sentence is just a letter. It can be used to build more complex sentences, but it cannot be taken apart.

Connectives 5

In the previous section, we considered symbolising fairly basic English sentences with atomic sentences of TFL. This leaves us wanting to deal with the English expressions 'and', 'or', 'not', and so forth. These are *connectives*—they can be used to form new sentences out of old ones. And in TFL, we shall make use of logical connectives to build complex sentences from atomic components. There are five logical connectives in TFL. This table summarises them, and they are explained throughout this section.

symbol	**what it is called**	**rough meaning**
$\neg$	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 ...'

5.1 Negation

Consider how we might symbolise these sentences:

1. Mary is in Barcelona.
2. It is not the case that Mary is in Barcelona.
3. Mary is not in Barcelona.

In order to symbolise sentence 1, we will need an atomic sentence. We might offer this symbolisation key:

B: Mary is in Barcelona.

Since sentence 2 is obviously related to the sentence 1, we shall not want to symbolise it with a completely different sentence. Roughly, sentence 2 means something like 'It is not the case that B'. In order to symbolise this, we need a symbol for negation. We will use '$\neg$'. Now we can symbolise sentence 2 with '$\neg B$'.

Sentence 3 also contains the word 'not'. And it is obviously equivalent to sentence 2. As such, we can also symbolise it with '$\neg B$'.

> A sentence can be symbolised as $\neg\phi$ if it can be paraphrased in English as 'It is not the case that...'.

It will help to offer a few more examples:

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

Let us use the following representation key:

R: The widget is replaceable

Sentence 4 can now be symbolised by 'R'. Moving on to sentence 5: saying the widget is irreplaceable means that it is not the case that the widget is replaceable. So even though sentence 5 does not contain the word 'not', we shall symbolise it as follows: '$\neg R$'.

Sentence 6 can be paraphrased as 'It is not the case that the widget is irreplaceable.' Which can again be paraphrased as 'It is not the case that it is not the case that the widget is replaceable'. So we might symbolise this English sentence with the TFL sentence '$\neg\neg R$'.

But some care is needed when handling negations. Consider:

7. Jane is happy.
8. Jane is unhappy.

If we let the TFL-sentence 'H' symbolise 'Jane is happy', then we can symbolise sentence 7 as 'H'. However, it would be a mistake to symbolise sentence 8 with '$\neg H$'. If Jane is unhappy, then she is not happy; but sentence 8 does not mean the same thing as 'It is not the case that Jane is happy'. Jane might be neither happy nor unhappy; she might be in a state of blank indifference. In order to symbolise sentence 8, then, we would need a new atomic sentence of TFL.

5.2 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 atomic sentences of TFL to symbolise sentences 9 and 10; perhaps

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

Sentence 9 can now be symbolised as 'A', and sentence 10 can be symbolised as 'B'. Sentence 11 roughly says 'A and B'. We need another symbol, to deal with 'and'. We will use '$\wedge$'. Thus we will symbolise it as '$(A \wedge B)$'. This connective is called CONJUNCTION. We also say that 'A' and 'B' are the two CONJUNCTS of the conjunction '$(A \wedge B)$'.

Notice that we make no attempt to symbolise the word 'also' in sentence 11. Words like 'both' and 'also' function to draw our attention to the fact that two things are being conjoined. Maybe they affect the emphasis of a sentence. But we will not (and cannot) symbolise such things in TFL.

Some more examples will bring out this point:

12. Barbara is athletic and energetic.
13. Barbara and Adam are both athletic.
14. Although Barbara is energetic, she is not athletic.
15. Adam is athletic, but Barbara is more athletic than him.

Sentence 12 is obviously a conjunction. The sentence says two things (about Barbara). In English, it is permissible to refer to Barbara only once. It *might* be tempting to think that we need to symbolise sentence 12 with something along the lines of 'B and energetic'. This would be a mistake. Once we symbolise part of a sentence as 'B', any further structure is lost. 'B' is an atomic sentence of TFL. Conversely, 'energetic' is not an English sentence at all. What we are aiming for is something like 'B and Barbara is energetic'. So we need to add another sentence letter to the symbolisation key. Let 'E' symbolise 'Barbara is energetic'. Now the entire sentence can be symbolised as '$(B \wedge E)$'.

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. The sentence can be paraphrased as 'Barbara is athletic, and Adam is athletic'. We can symbolise this in TFL as '$(B \wedge A)$', using the same symbolisation key that we have been using.

Sentence 14 is slightly more complicated. The word 'although' sets up a contrast between the first part of the sentence and the second part. Nevertheless, the sentence tells us 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'. And now we can symbolise this with the TFL sentence '$(E \wedge \neg B)$'. Note that we have lost all sorts of nuance in this symbolisation. There is a distinct difference in tone between sentence 14 and 'Both Barbara is energetic and it is not the case that Barbara is athletic'. TFL does not (and cannot) preserve these nuances.

Sentence 15 raises similar issues. There is a contrastive structure, but this is not something that TFL can deal with. So we can paraphrase the sentence as '*Both* Adam is athletic, *and* Barbara is more athletic than Adam'. (Notice that we once again replace the pronoun 'him' with 'Adam'.) How should we deal with the second conjunct? We already have the sentence letter 'A', which is being used to symbolise 'Adam is athletic', and the sentence 'B' which is being used to symbolise 'Barbara is athletic'; but neither of these concerns their relative athleticity. So, to to symbolise the entire sentence, we need a new sentence letter. Let the TFL sentence 'R' symbolise the English sentence 'Barbara is more athletic than Adam'. Now we can symbolise sentence 15 by '$(A \wedge R)$'.

> A sentence can be symbolised as $(\phi \wedge \psi)$ if it can be paraphrased in English as 'Both..., and...', or as '..., but ...', or as 'although ..., ...'.

You might be wondering why I am putting brackets around the conjunctions. The reason for this is brought out by considering how negation might interact with conjunction. Consider:

16. It's not the case that you will get both soup and salad.
17. You will not get soup but you will get salad.

Sentence 16 can be paraphrased as 'It is not the case that: both you will get soup and you will get salad'. Using this symbolisation key:

S_1: You get soup.
S_2: You get salad.

We would symbolise 'both you will get soup and you will get salad' as '$(S_1 \wedge S_2)$'. To symbolise sentence 16, then, we simply negate the whole sentence, thus: '$\neg(S_1 \wedge S_2)$'.

Sentence 17 is a conjunction: you *will not* get soup, and you *will* get salad. 'You will not get soup' is symbolised by '$\neg S_1$'. So to symbolise sentence 17 itself, we offer '$(\neg S_1 \wedge S_2)$'.

These English sentences are very different, and their symbolisations differ accordingly. In one of them, the entire conjunction is negated. In the other, just one conjunct is negated. Brackets help us to keep track of things like the *scope* of the negation.

5.3 Disjunction

Consider these sentences:

18. Either Denison will play golf with me, or he will watch movies.
19. Either Denison or Ellery will play golf with me.

For these sentences we can use this symbolisation key:

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

However, we shall again need to introduce a new symbol. Sentence 18 is symbolised by '$(D \vee M)$'. The connective is called DISJUNCTION. We also say that 'D' and 'M' are the DISJUNCTS of the disjunction '$(D \vee M)$'.

Sentence 19 is only slightly more complicated. There are two subjects, but the English sentence only gives the verb once. However, we can paraphrase sentence 19 as 'Either Denison will play golf with me, or Ellery will play golf with me'. Now we can obviously symbolise it by '$(D \vee E)$' again.

> A sentence can be symbolised as $(\phi \vee \psi)$ if it can be paraphrased in English as 'Either..., or....' 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 it says, on a restaurant menu, '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 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 19, above. I might

play golf with Denison, with Ellery, or with both Denison and Ellery. Sentence 19 merely says that I will play with *at least* one of them. This is called an INCLUSIVE OR. The TFL symbol '$\vee$' always symbolises an *inclusive or*.

It might help to see negation interact with disjunction. Consider:

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

Using the same symbolisation key as before, sentence 20 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'. To symbolise this in TFL, we need both disjunction and negation. 'It is not the case that you get soup' is symbolised by '$\neg S_1$'. 'It is not the case that you get salad' is symbolised by '$\neg S_2$'. So sentence 20 itself is symbolised by '$(\neg S_1 \vee \neg S_2)$'.

Sentence 21 also requires negation. It can be paraphrased as, '*It is not the case that* either you get soup or you get salad'. Since this negates the entire disjunction, we symbolise sentence 21 with '$\neg(S_1 \vee S_2)$'.

Sentence 22 is an *exclusive or*. We can break the sentence into two parts. The first part says that you get one or the other. We symbolise 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 symbolise this with '$\neg(S_1 \wedge S_2)$'. Now we just need to put the two parts together. As we saw above, 'but' can usually be symbolised with '$\wedge$'. Sentence 22 can thus be symbolised as '$((S_1 \vee S_2) \wedge \neg(S_1 \wedge S_2))$'.

This last example shows something important. Although the TFL symbol '$\vee$' always symbolises *inclusive or*, we can symbolise an *exclusive or* in TFL. We just have to use a few of our other symbols as well.

5.4 Conditional

Consider these sentences:

23. If Jean is in Paris, then Jean is in France.
24. Jean is in France only if Jean is in Paris.

Let's use the following symbolisation key:

P: Jean is in Paris.
F: Jean is in France

Sentence 23 is roughly of this form: 'if P, then F'. We will use the symbol '$\rightarrow$' to symbolise this 'if..., then...' structure. So we symbolise sentence 23 by '$(P \rightarrow F)$'. The connective is called THE CONDITIONAL. Here, 'P' is called the ANTECEDENT of the conditional '$(P \rightarrow F)$', and 'F' is called the CONSEQUENT.

Sentence 24 is also a conditional. Since the word 'if' appears in the second half of the sentence, it might be tempting to symbolise this in the same way as sentence 23. That would be a mistake. My knowledge of geography tells me that sentence 23 is unproblematically true: there is no way for Jean to be in Paris that doesn't involve Jean being in France. But sentence 24 is not so straightforward: were Jean in Dieppe, Lyons, or Toulouse, Jean would be

in France without being in Paris, thereby rendering sentence 24 false. Since geography alone dictates the truth of sentence 23, whereas travel plans (say) are needed to know the truth of sentence 24, they must mean different things.

In fact, sentence 24 can be paraphrased as 'If Jean is in France, then Jean is in Paris'. So we can symbolise it by '$(F \to P)$'.

> A sentence can be symbolised as $\phi \to \psi$ if it can be paraphrased in English as 'If A, then B' or 'A only if B'.

In fact, many English expressions can be represented using the conditional. Consider:

25. For Jean to be in Paris, it is necessary that Jean be in France.
26. It is a necessary condition on Jean's being in Paris that she be in France.
27. For Jean to be in France, it is sufficient that Jean be in Paris.
28. It is a sufficient condition on Jean's being in France that she be in Paris.

If we think really hard, all four of these sentences mean the same as 'If Jean is in Paris, then Jean is in France'. So they can all be symbolised by '$P \to F$'.

It is important to bear in mind that the connective '$\to$' tells us only that, if the antecedent is true, then the consequent is true. It says nothing about a *causal* connection between two events (for example). In fact, we lose a huge amount when we use '$\to$' to symbolise English conditionals. We shall return to this in §§9.3 and 11.5.

5.5 Biconditional

Consider these sentences:

29. Shergar is a horse only if it he is a mammal
30. Shergar is a horse if he is a mammal
31. Shergar is a horse if and only if he is a mammal

We shall use the following symbolisation key:

H: Shergar is a horse
M: Shergar is a mammal

Sentence 29, for reasons discussed above, can be symbolised by '$H \to M$'.

Sentence 30 is importantly different. It can be paraphrased as, 'If Shergar is a mammal then Shergar is a horse'. So it can be symbolised by '$M \to H$'.

Sentence 31 says something stronger than either 29 or 30. It can be paraphrased as 'Shergar is a horse if he is a mammal, and Shergar is a horse only if Shergar is a mammal'. This is just the conjunction of sentences 29 and 30. So we can symbolise it as '$(H \to M) \wedge (M \to H)$'. We call this a BICONDITIONAL, because it entails the conditional in both directions.

We could treat every biconditional this way. So, just as we do not need a new TFL symbol to deal with *exclusive or*, we do not really need a new TFL symbol to deal with biconditionals. However, we will use '$\leftrightarrow$' to symbolise the biconditional. So we can symbolise sentence 31 with the TFL sentence '$H \leftrightarrow M$'.

The expression 'if and only if' occurs a lot in philosophy and logic. For brevity, we can abbreviate it with the snappier word 'iff'. I shall follow this practice. So 'if' with only *one* 'f' is the English conditional. But 'iff' with *two* 'f's is the English biconditional. Armed with this we can say:

> A sentence can be symbolised as $\phi \leftrightarrow \psi$ if it can be paraphrased in English as 'A iff B'; that is, as 'A if and only if B'.

A word of caution. Ordinary speakers of English often use 'if . . . , then. . . ' when they really mean to use something more like '. . . if and only if . . . '. Perhaps your parents told you, when you were a child: 'if you don't eat your greens, you won't get any pudding'. Suppose you ate your greens, but that your parents refused to give you any pudding, on the grounds that they were only committed to the *conditional* (roughly 'if you get pudding, then you will have eaten your greens'), rather than the biconditional (roughly, 'you get pudding iff you eat your greens'). Well, a tantrum would rightly ensue. So, be aware of this when interpreting people; but in your own writing, make sure you use the biconditional iff you mean to.

5.6 Unless

We have now introduced all of the connectives of TFL. We can use them together to symbolise many kinds of sentences. But a typically nasty case is when we use the English-language connective 'unless':

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

These two sentences are clearly equivalent. To symbolise them, we shall use the symbolisation key:

J: You will wear a jacket.
D: You will catch a cold.

Both sentences mean that if you do not wear a jacket, then you will catch cold. With this in mind, we might symbolise them as '$\neg J \rightarrow D$'.

Equally, both sentences mean that if you do not catch a cold, then you must have worn a jacket. With this in mind, we might symbolise them as '$\neg D \rightarrow J$'.

Equally, both sentences mean that either you will wear a jacket or you will catch a cold. With this in mind, we might symbolise them as '$J \vee D$'.

All three are correct symbolisations. Indeed, in chapter 3 we shall see that all three symbolisations are equivalent in TFL.

> If a sentence can be paraphrased as 'Unless A, B,' then it can be symbolised as '$\phi \vee \psi$'.

Again, though, there is a little complication. 'Unless' can be symbolised as a conditional; but as I said above, people often use the conditional (on its own) when they mean to use the biconditional. Equally, 'unless' can be symbolised as a disjunction; but there are two kinds of disjunction (exclusive and inclusive). So it will not surprise you to discover that ordinary speakers of English often

use ‘unless’ to mean something more like the biconditional, or like exclusive disjunction. Suppose I say: ‘I shall go running unless it rains’. I probably mean something like ‘I shall go running iff it does not rain’ (i.e. the biconditional), or ‘either I shall go running or it will rain, but not both’ (i.e. exclusive disjunction). Again: be aware of this when interpreting what other people have said, but be precise in your writing, unless you want to be deliberately ambiguous.

Practice exercises

A. Using the symbolisation key given, symbolise each English sentence in TFL.

M: Those creatures are men in suits.
C: Those creatures are chimpanzees.
G: Those creatures are gorillas.

1. Those creatures are not men in suits.
2. Those creatures are men in suits, or they are not.
3. Those creatures are either gorillas or chimpanzees.
4. Those creatures are neither gorillas nor chimpanzees.
5. If those creatures are chimpanzees, then they are neither gorillas nor men in suits.
6. Unless those creatures are men in suits, they are either chimpanzees or they are gorillas.

B. Using the symbolisation key given, symbolise each English sentence in TFL.

A: Mister Ace was murdered.
B: The butler did it.
C: The cook did it.
D: The Duchess is lying.
E: Mister Edge was murdered.
F: The murder weapon was a frying pan.

1. Either Mister Ace or Mister Edge was murdered.
2. If Mister Ace was murdered, then the cook did it.
3. If Mister Edge was murdered, then the cook did not do it.
4. Either the butler did it, or the Duchess is lying.
5. The cook did it only if the Duchess is lying.
6. If the murder weapon was a frying pan, then the culprit must have been the cook.
7. If the murder weapon was not a frying pan, then the culprit was either the cook or the butler.
8. Mister Ace was murdered if and only if Mister Edge was not murdered.
9. The Duchess is lying, unless it was Mister Edge who was murdered.
10. If Mister Ace was murdered, he was done in with a frying pan.
11. Since the cook did it, the butler did not.
12. Of course the Duchess is lying!

C. Using the symbolisation key given, symbolise each English sentence in TFL.

E_1: Ava is an electrician.

E_2: Harrison is an electrician.
F_1: Ava is a firefighter.
F_2: Harrison is a firefighter.
S_1: Ava is satisfied with her career.
S_2: Harrison is satisfied with his career.

1. Ava and Harrison are both electricians.
2. If Ava is a firefighter, then she is satisfied with her career.
3. Ava is a firefighter, unless she is an electrician.
4. Harrison is an unsatisfied electrician.
5. Neither Ava nor Harrison is an electrician.
6. Both Ava and Harrison are electricians, but neither of them find it satisfying.
7. Harrison is satisfied only if he is a firefighter.
8. If Ava is not an electrician, then neither is Harrison, but if she is, then he is too.
9. Ava is satisfied with her career if and only if Harrison is not satisfied with his.
10. If Harrison is both an electrician and a firefighter, then he must be satisfied with his work.
11. It cannot be that Harrison is both an electrician and a firefighter.
12. Harrison and Ava are both firefighters if and only if neither of them is an electrician.

D. Give a symbolisation key and symbolise the following English sentences in TFL.

1. Alice and Bob are both spies.
2. If either Alice or Bob is a spy, then the code has been broken.
3. If neither Alice nor Bob is a spy, then the code remains unbroken.
4. The German embassy will be in an uproar, unless someone has broken the code.
5. Either the code has been broken or it has not, but the German embassy will be in an uproar regardless.
6. Either Alice or Bob is a spy, but not both.

E. Give a symbolisation key and symbolise the following English sentences in TFL.

1. If there is food to be found in the pridelands, then Rafiki will talk about squashed bananas.
2. Rafiki will talk about squashed bananas unless Simba is alive.
3. Rafiki will either talk about squashed bananas or he won't, but there is food to be found in the pridelands regardless.
4. Scar will remain as king if and only if there is food to be found in the pridelands.
5. If Simba is alive, then Scar will not remain as king.

F. For each argument, write a symbolisation key and symbolise all of the sentences of the argument in TFL.

1. If Dorothy plays the piano in the morning, then Roger wakes up cranky. Dorothy plays piano in the morning unless she is distracted. So if Roger does not wake up cranky, then Dorothy must be distracted.
2. It will either rain or snow on Tuesday. If it rains, Neville will be sad. If it snows, Neville will be cold. Therefore, Neville will either be sad or cold on Tuesday.
3. If Zoog remembered to do his chores, then things are clean but not neat. If he forgot, then things are neat but not clean. Therefore, things are either neat or clean; but not both.

G. We symbolised an *exclusive or* using '$\vee$', '$\wedge$', and '$\neg$'. How could you symbolise an *exclusive or* using only two connectives? Is there any way to symbolise an *exclusive or* using only one connective?

Sentences of TFL 6

The sentence 'either apples are red, or berries are blue' is a sentence of English, and the sentence '$(A \vee B)$' is a sentence of TFL. Although we can identify sentences of English when we encounter them, we do not have a formal definition of 'sentence of English'. But in this chapter, we shall offer a complete *definition* of what counts as a sentence of TFL. This is one respect in which a formal language like TFL is more precise than a natural language like English.

6.1 Expressions

We have seen that there are three kinds of symbols in TFL:

Atomic sentences	$A, B, C, \ldots, Z$
with subscripts, as needed	$A_1, B_1, Z_1, A_2, A_{25}, J_{375}, \ldots$
Connectives	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
Brackets	(,)

We define an EXPRESSION OF TFL as any string of symbols of TFL. Take any of the symbols of TFL and write them down, in any order, and you have an expression of TFL.

6.2 Sentences

Of course, many expressions of TFL will be total gibberish. We want to know when an expression of TFL amounts to a *sentence*.

Obviously, individual atomic sentences like 'A' and 'G_{13}' should count as sentences. We can form further sentences out of these by using the various connectives. Using negation, we can get '$\neg A$' and '$\neg G_{13}$'. Using conjunction, we can get '$(A \wedge G_{13})$', '$(G_{13} \wedge A)$', '$(A \wedge A)$', and '$(G_{13} \wedge G_{13})$'. We could also apply negation repeatedly to get sentences like '$\neg\neg A$' or apply negation along with conjunction to get sentences like '$\neg(A \wedge G_{13})$' and '$\neg(G_{13} \wedge \neg G_{13})$'. The possible combinations are endless, even starting with just these two sentence letters, and there are infinitely many sentence letters. So there is no point in trying to list all the sentences one by one.

Instead, we will describe the process by which sentences can be *constructed*. Consider negation: Given any sentence ϕ of TFL, $\neg\phi$ is a sentence of TFL. (Why the funny fonts? I return to this in §7.)

We can say similar things for each of the other connectives. For instance, if ϕ and ψ are sentences of TFL, then $(\phi \wedge \psi)$ is a sentence of TFL. Providing clauses like this for all of the connectives, we arrive at the following formal definition for a SENTENCE OF TFL:

1. Every atomic sentence is a sentence.
2. If ϕ is a sentence, then $\neg\phi$ is a sentence.
3. If ϕ and ψ are sentences, then $(\phi \wedge \psi)$ is a sentence.
4. If ϕ and ψ are sentences, then $(\phi \vee \psi)$ is a sentence.
5. If ϕ and ψ are sentences, then $(\phi \rightarrow \psi)$ is a sentence.
6. If ϕ and ψ are sentences, then $(\phi \leftrightarrow \psi)$ is a sentence.
7. Nothing else is a sentence.

Definitions like this are called *recursive*. Recursive definitions begin with some specifiable base elements, and then present ways to generate indefinitely many more elements by compounding together previously established ones. To give you a better idea of what a recursive definition is, we can give a recursive definition of the idea of *an ancestor of mine*. We specify a base clause.

- My parents are ancestors of mine.

and then offer further clauses like:

- If x is an ancestor of mine, then x's parents are ancestors of mine.
- Nothing else is an ancestor of mine.

Using this definition, we can easily check to see whether someone is my ancestor: just check whether she is the parent of the parent of. . . one of my parents. And the same is true for our recursive definition of sentences of TFL. Just as the recursive definition allows complex sentences to be built up from simpler parts, the definition allows us to decompose sentences into their simpler parts. And if we get down to atomic sentences, then we are ok.

Let's consider some examples.

Suppose we want to know whether or not '$\neg\neg\neg D$' is a sentence of TFL. 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 an atomic sentence of TFL, so we know that 'D' is a sentence by the first clause of the definition. So for a compound sentence 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.

Next, consider the example '$\neg(P \wedge \neg(\neg Q \vee R))$'. Looking at the second clause of the definition, this is a sentence if '$(P \wedge \neg(\neg Q \vee R))$' is. And this is a sentence if *both* 'P' *and* '$\neg(\neg Q \vee R)$' are sentences. The former is an atomic sentence, and the latter is a sentence if '$(\neg Q \vee R)$' is a sentence. It is. Looking at the fourth clause of the definition, this is a sentence if both '$\neg Q$' and 'R' are sentences. And both are!

Ultimately, every sentence is constructed nicely out of atomic sentences. When we are dealing with a *sentence* other than an atomic sentence, we can see that there must be some sentential connective that was introduced *last*, when constructing the sentence. We call that the MAIN LOGICAL OPERATOR of the sentence. In the case of '$\neg\neg\neg D$', the main logical operator is the very first '$\neg$' sign. In the case of '$(P \wedge \neg(\neg Q \vee R))$', the main logical operator is '$\wedge$'. In the case of '$((\neg E \vee F) \rightarrow \neg\neg G)$', the main logical operator is '$\rightarrow$'.

As a general rule, you can find the main logical operator for a sentence by using the following method:

- If the first symbol in the sentence is '$\neg$', then that is the main logical operator
- Otherwise, start counting the brackets. For each open-bracket, i.e. '(', add 1; for each closing-bracket, i.e. ')', subtract 1. When your count is at exactly 1, the first operator you hit (*apart* from a '$\neg$') is the main logical operator.

(Note: if you do use this method, then make sure to include *all* the brackets in the sentence, rather than omitting some as per the conventions of §6.3!)

The inductive or "recursive" structure of sentences in TFL 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. We will return to this point in chapter 3.

The recursive structure of sentences in TFL also allows us to give a formal definition of the *scope* of a negation (mentioned in §5.2). The scope of a '$\neg$' is the subsentence for which '$\neg$' is the main logical operator. So in a sentence like:

$$(P \wedge (\neg(R \wedge B) \leftrightarrow Q))$$

this was constructed by conjoining 'P' with '$(\neg(R \wedge B) \leftrightarrow Q)$'. This last sentence was constructed by placing a biconditional between '$\neg(R \wedge B)$' and 'Q'. And the former of these sentences—a subsentence of our original sentence—is a sentence for which '$\neg$' is the main logical operator. So the scope of the negation is just '$\neg(R \wedge B)$'. More generally:

> The SCOPE of a connective (in a sentence) is the subsentence for which that connective is the main logical operator.

6.3 Bracketing conventions

Strictly speaking, the brackets in '$(Q \wedge R)$' are an indispensable part of the sentence. Part of this is because we might use '$(Q \wedge R)$' as a subsentence in a more complicated sentence. For example, we might want to negate '$(Q \wedge R)$', obtaining '$\neg(Q \wedge R)$'. If we just had '$Q \wedge R$' without the brackets 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 '$(\neg Q \wedge R)$'. But as we saw in §5.2, this is very different from '$\neg(Q \wedge R)$'.

Strictly speaking, then, '$Q \wedge R$' is *not* a sentence. It is a mere *expression*.

When working with TFL, however, it will make our lives easier if we are sometimes a little less than strict. So, here are some convenient conventions.

First, we allow ourselves to omit the *outermost* brackets of a sentence. Thus we allow ourselves to write '$Q \wedge R$' instead of the sentence '$(Q \wedge R)$'. However, we must remember to put the brackets back in, when we want to embed the sentence into a more complicated sentence!

Second, it can be a bit painful to stare at long sentences with many nested pairs of brackets. To make things a bit easier on the eyes, we shall allow ourselves to use square brackets, '[' and ']', instead of rounded ones. So there is no logical difference between '$(P \vee Q)$' and '$[P \vee Q]$', for example.

Combining these two conventions, we can rewrite the unwieldy sentence

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

rather more simply as follows:

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

The scope of each connective is now much clearer.

Practice exercises

A. For each of the following: (a) Is it a sentence of TFL, strictly speaking? (b) Is it a sentence of TFL, allowing for our relaxed bracketing conventions?

1. (A)
2. $J_{374} \vee \neg J_{374}$
3. $\neg\neg\neg\neg F$
4. $\neg \wedge S$
5. $(G \wedge \neg G)$
6. $(A \to (A \wedge \neg F)) \vee (D \leftrightarrow E)$
7. $[(Z \leftrightarrow S) \to W] \wedge [J \vee X]$
8. $(F \leftrightarrow \neg D \to J) \vee (C \wedge D)$

B. Are there any sentences of TFL that contain no atomic sentences? Explain your answer.

C. What is the scope of each connective in the sentence

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

Use and mention 7

In this chapter, I have talked a lot *about* sentences. So I need to pause to explain an important, and very general, point.

7.1 Quotation conventions

Consider these two sentences:

- David Cameron is the Prime Minister.
- The expression 'David Cameron' is composed of two uppercase letters and ten lowercase letters

When we want to talk about the Prime Minister, we *use* his name. When we want to talk about the Prime Minister's name, we *mention* that name. And we do so by putting it in quotation marks.

There is a general point here. When we want to talk about things in the world, we just *use* words. When we want to talk about words, we typically have to *mention* those words. We need to indicate that we are mentioning them, rather than using them. To do this, some convention is needed. We can put them in quotation marks, or display them centrally in the page (say). So this sentence:

- 'David Cameron' is the Prime Minister.

says that some *expression* is the Prime Minister. And that's false. The *man* is the Prime Minister; his *name* isn't. Conversely, this sentence:

- David Cameron is composed of two uppercase letters and ten lowercase letters.

also says something false: David Cameron is a man, made of meat rather than letters. One final example:

- "'David Cameron'" is the name of 'David Cameron'.

On the left-hand-side, here, we have the name of a name. On the right hand side, we have a name. Perhaps this kind of sentence only occurs in logic textbooks, but it is true.

Those are just general rules for quotation, and you should observe them carefully in all your work! To be clear, the quotation-marks here do not indicate indirect speech. They indicate that you are moving from talking about an object, to talking about the name of that object.

7.2 Object language and metalanguage

These general quotation conventions are of particular importance for us. After all, we are describing a formal language here, TFL, and so we are often *mentioning* expressions from TFL.

When we talk about a language, the language that we are talking about is called the OBJECT LANGUAGE. The language that we use to talk about the object language is called the METALANGUAGE.

For the most part, the object language in this chapter has been the formal language that we have been developing: TFL. The metalanguage is English. Not conversational English exactly, but English supplemented with some additional vocabulary which helps us to get along.

Now, I have used italic uppercase letters for atomic sentences of TFL:

$$A, B, C, Z, A_1, B_4, A_{25}, J_{375}, \ldots$$

These are sentences of the object language (TFL). They are not sentences of English. So I must not say, for example:

- D is an atomic sentence of TFL.

Obviously, I am trying to come out with an English sentence that says something about the object language (TFL). But 'D' is a sentence of TFL, and no part of English. So the preceding is gibberish, just like:

- Schnee ist weiß is a German sentence.

What we surely meant to say, in this case, is:

- 'Schnee ist weiß' is a German sentence.

Equally, what we meant to say above is just:

- 'D' is an atomic sentence of TFL.

The general point is that, whenever we want to talk in English about some specific expression of TFL, we need to indicate that we are *mentioning* the expression, rather than *using* it. We can either deploy quotation marks, or we can adopt some similar convention, such as placing it centrally in the page.

7.3 Greek letters and Quine quotes

However, we do not just want to talk about *specific* expressions of TFL. We also want to be able to talk about *any arbitrary* sentence of TFL. Indeed, I had to do this in §6, when I presented the recursive definition of a sentence of TFL. I used lower-case letters from the Greek alphabet to do this, namely:

$$\phi, \psi, \omega, \delta, \ldots$$

These symbols do not belong to TFL. Rather, they are part of our (augmented) metalanguage that we use to talk about *any* expression of TFL. To repeat the second clause of the recursive definition of a sentence of TFL, we said:

3. If ϕ is a sentence, then $\neg\phi$ is a sentence.

This talks about *arbitrary* sentences. If we had instead offered:

- If 'A' is a sentence, then '$\neg A$' is a sentence.

this would not have allowed us to determine whether '$\neg B$' is a sentence. To emphasise, then:

> 'ϕ' is a symbol in augmented English, which we use to talk about any TFL expression. 'A' is a particular atomic sentence of TFL.

But this last example raises a further complications for our quotation conventions. I have not included any quotation marks in the third clause of our recursive definition. Should I have done so?

The problem is that the expression on the right-hand-side of this rule is not a sentence of English, since it contains '$\neg$'. So we might try to write:

3′. If ϕ is a sentence, then '$\neg\phi$' is a sentence.

But this is no good: '$\neg\phi$' is not a TFL sentence, since 'ϕ' is a symbol of (augmented) English rather than a symbol of TFL.

What we really want to say is something like this:

3″. If ϕ and ψ are sentences, then the result of concatenating the symbol '$\neg$' with the sentence ϕ is a sentence.

This is impeccable, but rather long-winded. But we can avoid long-windedness by creating our own conventions. We can perfectly well stipulate that an expression like '$\neg\phi$' should simply be read *directly* in terms of rules for concatenation. So, *officially*, the metalanguage expression '$\neg\phi$' simply abbreviates:

> the result of concatenating the symbol '$\neg$' with the sentence ϕ

and similarly, for expressions like '$(\phi \wedge \psi)$', '$(\phi \vee \psi)$', etc.

7.4 Quotation conventions for arguments

One of our main purposes for using TFL is to study arguments, and that will be our concern in chapter 3. In English, the premises of an argument are often expressed by individual sentences, and the conclusion by a further sentence. Since we can symbolise English sentences, we can symbolise English arguments using TFL. Thus we might ask whether the argument whose premises are the TFL sentences 'A' and '$A \rightarrow B$', and whose conclusion is the TFL sentence 'C', is valid. However, it is quite a mouthful to write that every time. So instead I shall introduce another bit of abbreviation. This:

$$\phi_1, \phi_2, \ldots, \phi_n \therefore \psi$$

abbreviates:

> the argument with premises $\phi_1, \phi_2, \ldots, \phi_n$ and conclusion ψ

To avoid unnecessary clutter, we shall not regard this as requiring quotation marks around it. (Note, then, that '$\therefore$' is a symbol of our augmented *metalanguage*, and not a new symbol of TFL.)

Chapter 3

Truth tables

Characteristic truth tables 8

Any non-atomic sentence of TFL 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 comprise it. In order to know the truth value of '$(D \wedge E)$', for instance, you only need to know the truth value of 'D' and the truth value of 'E'.

We introduced five connectives in chapter 2. So we simply need to explain how they map between truth values. For convenience, we shall abbreviate 'True' with 'T' and 'False' with 'F'. (But just to be clear, the two truth values are True and False; the truth values are not *letters*!)

Negation. For any sentence ϕ: If ϕ is true, then $\neg\phi$ is false. If $\neg\phi$ is true, then ϕ is false. We can summarize this in the *characteristic truth table* for negation:

ϕ	$\neg\phi$
T	F
F	T

Conjunction. For any sentences ϕ and ψ, $\phi\wedge\psi$ is true if and only if both ϕ and ψ are true. We can summarize this in the characteristic truth table for conjunction:

ϕ	ψ	$\phi \wedge \psi$
T	T	T
T	F	F
F	T	F
F	F	F

Note that conjunction is *symmetrical*. The truth value for $\phi \wedge \psi$ is always the same as the truth value for $\psi \wedge \phi$.

Disjunction. Recall that '$\vee$' always represents inclusive or. So, for any sentences ϕ and ψ, $\phi \vee \psi$ is true if and only if either ϕ or ψ is true. We can summarize this in the characteristic truth table for disjunction:

ϕ	ψ	$\phi \vee \psi$
T	T	T
T	F	T
F	T	T
F	F	F

Like conjunction, disjunction is symmetrical.

Conditional. I'm just going to come clean and admit it. Conditionals are a right old mess in TFL. Exactly how much of a mess they are is a matter of *philosophical* contention. I shall discuss a few of the subtleties in §§9.3 and 11.5. For now, I am going to stipulate the following: $\phi \to \psi$ is false if and only if ϕ is true and ψ is false. We can summarize this with a characteristic truth table for the conditional.

ϕ	ψ	$\phi \to \psi$
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 $\phi \to \psi$ has a very different truth table from $\psi \to \phi$.

Biconditional. Since a biconditional is to be the same as the conjunction of a conditional running in each direction, we shall want the truth table for the biconditional to be:

ϕ	ψ	$\phi \leftrightarrow \psi$
T	T	T
T	F	F
F	T	F
F	F	T

Unsurprisingly, the biconditional is symmetrical.

Truth-functional connectives 9

9.1 The idea of truth-functionality

I want to introduce an important idea.

> A connective is TRUTH-FUNCTIONAL iff the truth value of a sentence with that connective as its main logical operator is uniquely determined by the truth value(s) of the constituent sentence(s).

Every connective in TFL is truth-functional. The truth value of a negation is uniquely determined by the truth value of the unnegated sentence. The truth value of a conjunction is uniquely determined by the truth value of both conjuncts. The truth value of a disjunction is uniquely determined by the truth value of both disjuncts. And so on. To determine the truth value of some TFL sentence, we only need to know the truth value of its components.

This is what gives TFL its name: it is *truth-functional logic*.

In plenty of languages there are connectives that are not truth-functional. In English, for example, we can form a new sentence from any simpler sentence by prefixing it with 'It is necessarily the case that...'. The truth value of this new sentence is not fixed solely by the truth value of the original sentence. For consider two true sentences:

1. $2 + 2 = 4$
2. Shostakovich wrote fifteen string quartets

Whereas it is necessarily the case that $2 + 2 = 4$, it is not *necessarily* the case that Shostakovich wrote fifteen string quartets. If Shostakovich had died earlier, he would have failed to finish Quartet no. 15; if he had lived longer, he might have written a few more. So 'It is necessarily the case that...' is a connective of English, but it is not *truth-functional*.

9.2 Symbolising versus translating

All of the connectives of TFL are truth-functional. But more than that: they really do nothing *but* map us between truth values.

When we symbolise a sentence or an argument in TFL, we ignore everything *besides* the contribution that the truth values of a component might make to the truth value of the whole. There are subtleties to our ordinary claims that far outstrip their mere truth values. Sarcasm; poetry; snide implicature; emphasis; these are important parts of everyday discourse. But none of this is retained

in TFL. As remarked in §5, TFL cannot capture the subtle differences between the following English sentences:

1. Jon is fat and Jon is quick
2. Although Jon is fat, Jon is quick
3. Despite being fat, Jon is quick
4. Jon is quick, albeit fat
5. Jon's fatness notwithstanding, he is quick

All of the above sentences will be symbolised with the same TFL sentence, perhaps '$F \wedge Q$'.

I keep saying that we use TFL sentences to *symbolise* English sentences. Many other textbooks talk about *translating* English sentences into TFL. But a good translation should preserve certain facets of meaning, and—as I have just pointed out—TFL just cannot do that. This is why I shall speak of *symbolising* English sentences, rather than of *translating* them.

This affects how we should understand our symbolisation keys. Consider a key like:

> F: Jon is fat.
> Q: Jon is quick.

Other textbooks will understand this as a stipulation that the TFL sentence 'F' should *mean* that Jon is fat, and that the TFL sentence 'Q' should *mean* that Jon is quick. But TFL just is totally unequipped to deal with *meaning*. The preceding symbolisation key is doing no more nor less than stipulating that the TFL sentence 'F' should take the same truth value as the English sentence 'Jon is fat' (whatever that might be), and that the TFL sentence 'Q' should take the same truth value as the English sentence 'Jon is quick' (whatever that might be).

> When we treat a TFL sentence as *symbolising* an English sentence, we are stipulating that the TFL sentence is to take the same truth value as that English sentence.

9.3 Indicative versus subjunctive conditionals

I want to bring home the point that TFL can *only* deal with truth functions by considering the case of the conditional. When I introduced the characteristic truth table for the material conditional in §8, I did not say anything to justify it. Let me now offer a justification, which follows Dorothy Edgington.[1]

Suppose that Lara has drawn some shapes on a piece of paper, and coloured some of them in. I have not seen them, but I claim:

> If any shape is grey, then that shape is also circular.

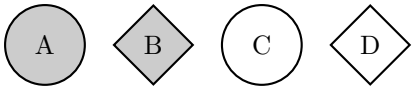
As it happens, Lara has drawn the following:

[1] Dorothy Edgington, 'Conditionals', 2006, in the *Stanford Encyclopedia of Philosophy* (http://plato.stanford.edu/entries/conditionals/).

In this case, my claim is surely true. Shapes C and D are not grey, and so can hardly present *counterexamples* to my claim. Shape A *is* grey, but fortunately it is also circular. So my claim has no counterexamples. It must be true. And that means that each of the following *instances* of my claim must be true too:

- If A is grey, then it is circular (true antecedent, true consequent)
- If C is grey, then it is circular (false antecedent, true consequent)
- If D is grey, then it is circular (false antecedent, false consequent)

However, if Lara had drawn a fourth shape, thus:



then my claim would have be false. So it must be that this claim is false:

- If B is grey, then it is a circular (true antecedent, false consequent)

Now, recall that every connective of TFL has to be truth-functional. This means that the mere truth value of the antecedent and consequent must uniquely determine the truth value of the conditional as a whole. Thus, from the truth values of our four claims—which provide us with all possible combinations of truth and falsity in antecedent and consequent—we can read off the truth table for the material conditional.

What this argument shows is that '→' is the *only* candidate for a truth-functional conditional. Otherwise put, *it is the best conditional that TFL can provide*. But is it any good, as a surrogate for the conditionals we use in everyday language? Consider two sentences:

1. If Mitt Romney had won the 2012 election, then he would have been the 45th President of the USA.
2. If Mitt Romney had won the 2012 election, then he would have turned into a helium-filled balloon and floated away into the night sky.

Sentence 1 is true; sentence 2 is false. But both have false antecedents and false consequents. So the truth value of the whole sentence is not uniquely determined by the truth value of the parts. Do not just blithely assume that you can adequately symbolise an English 'if..., then...' with TFL's '→'.

The crucial point is that sentences 1 and 2 employ *subjunctive* conditionals, rather than *indicative* conditionals. They ask us to imagine something contrary to fact—Mitt Romney lost the 2012 election—and then ask us to evaluate what *would* have happened in that case. Such considerations just cannot be tackled using '→'.

I shall say more about the difficulties with conditionals in §11.5. For now, I shall content myself with the observation that '→' is the only candidate for a truth-functional conditional, but that many English conditionals cannot be represented adequately using '→'. TFL is an intrinsically limited language.

Complete truth tables 10

So far, we have considered assigning truth values to TFL sentences indirectly. We have said, for example, that a TFL sentence such as 'B' is to take the same truth value as the English sentence 'Big Ben is in London' (whatever that truth value may be). But we can also assign truth values *directly*. We can simply stipulate that 'B' is to be true, or stipulate that it is to be false.

> A VALUATION is any assignment of truth values to particular atomic sentences of TFL.

The power of truth tables lies in the following. Each row of a truth table represents a possible valuation. The entire truth table represents all possible valuations. And the truth table provides us with a means to calculate the truth value of complex sentences, on each possible valuation. This is easiest to explain by example.

10.1 A worked example

Consider the sentence '$(H \wedge I) \rightarrow H$'. There are four possible ways to assign True and False to the atomic sentence 'H' and 'I'—four possible valuations—which we can represent as follows:

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

To calculate the truth value of the entire sentence '$(H \wedge I) \rightarrow H$', we first copy the truth values for the atomic sentences 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, $(\phi \wedge \psi)$, with 'H' as ϕ and with 'I' as ψ. The characteristic truth table for conjunction gives the truth conditions for *any* sentence of the form $(\phi \wedge \psi)$, whatever ϕ and ψ might be. It summarises the point that a conjunction is true iff both conjuncts

are true. In this case, our conjuncts are just 'H' and 'I'. They are both true on (and only on) the first line of the truth table. Accordingly, we can calculate the truth value of the conjunction on all four rows.

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

Now, the entire sentence that we are dealing with is a conditional, $\phi \rightarrow \psi$, with '$(H \wedge I)$' as ϕ and with 'H' as ψ. 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:

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

The conditional is the main logical connective of the sentence. And the column of 'T's 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. Since we have considered all four possible assignments of truth and falsity to 'H' and 'I'—since, that is, we have considered all the different *valuations*—we can say that '$(H \wedge I) \rightarrow H$' is true on every valuation.

In this example, I have not repeated all of the entries in every column 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. The column that matters most is the column underneath the *main logical operator* for the sentence, since this tells you the truth value of the entire sentence. I have emphasised this, by putting this column in bold. When you work through truth tables yourself, you should similarly emphasise it (perhaps by underlining).

10.2 Building complete truth tables

A COMPLETE TRUTH TABLE has a line for every possible assignment of True and False to the relevant atomic sentences. Each line represents a *valuation*, and a complete truth table has a line for all the different valuations.

The size of the complete truth table depends on the number of different atomic sentences in the table. A sentence that contains only one atomic sentence 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 the 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. 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 logical operator, we see that the sentence is false on both rows of the table; i.e., the sentence is false regardless of whether 'C' is true or false. It is false on every valuation.

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

A sentence that contains three atomic sentences requires eight lines:

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 '$M \wedge (N \vee P)$' can 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 atomic sentences requires 16 lines. Five letters, 32 lines. Six letters, 64 lines. And so on. To be perfectly general: If a complete truth table has n different atomic sentences, then it must have 2^n lines.

In order to fill in the columns of a complete truth table, begin with the rightmost atomic sentence and alternate between 'T' and 'F'. In the next column to the left, write two 'T's, write two 'F's, and repeat. For the third atomic sentence, write four 'T's followed by four 'F's. This yields an eight line truth table like the one above. For a 16 line truth table, the next column of atomic sentences should have eight 'T's followed by eight 'F's. For a 32 line table, the next column would have 16 'T's followed by 16 'F's. And so on.

10.3 More bracketing conventions

Consider these two sentences:

$$((A \wedge B) \wedge C)$$
$$(A \wedge (B \wedge C))$$

These have the same truth table. Consequently, it will never make any difference from the perspective of truth value – which is all that TFL cares about (see §9) – which of the two sentences we assert (or deny). And since the order of the brackets does not matter, I shall allow us to drop them. In short, we can save some ink and some eyestrain by writing:

$$A \wedge B \wedge C$$

The general point is that, if we just have a long list of conjunctions, we can drop the inner brackets. (I already allowed us to drop outermost brackets in §6.) The same observation holds for disjunctions. Since the following sentences have exactly the same truth table:

$$((A \vee B) \vee C)$$
$$(A \vee (B \vee C))$$

we can simply write:

$$A \vee B \vee C$$

And generally, if we just have a long list of disjunctions, we can drop the inner brackets. *But be careful.* These two sentences have *different* truth tables:

$$((A \rightarrow B) \rightarrow C)$$
$$(A \rightarrow (B \rightarrow C))$$

So if we were to write:

$$A \rightarrow B \rightarrow C$$

it would be dangerously ambiguous. So we must not do the same with conditionals. Equally, these sentences have different truth tables:

$$((A \vee B) \wedge C)$$
$$(A \vee (B \wedge C))$$

So if we were to write:

$$A \vee B \wedge C$$

it would be dangerously ambiguous. *Never write this.* The moral is: you can drop brackets when dealing with a long list of conjunctions, or when dealing with a long list of disjunctions. But that's it.

Practice exercises

A. Offer complete truth tables for each of the following:

1. $A \to A$
2. $C \to \neg C$
3. $(A \leftrightarrow B) \leftrightarrow \neg(A \leftrightarrow \neg B)$
4. $(A \to B) \vee (B \to A)$
5. $(A \wedge B) \to (B \vee A)$
6. $\neg(A \vee B) \leftrightarrow (\neg A \wedge \neg B)$
7. $\big[(A \wedge B) \wedge \neg(A \wedge B)\big] \wedge C$
8. $\big[(A \wedge B) \wedge C\big] \to B$
9. $\neg\big[(C \vee A) \vee B\big]$

B. Check all the claims made in introducing the new notational conventions in §10.3, i.e. show that:

1. '$((A \wedge B) \wedge C)$' and '$(A \wedge (B \wedge C))$' have the same truth table
2. '$((A \vee B) \vee C)$' and '$(A \vee (B \vee C))$' have the same truth table
3. '$((A \vee B) \wedge C)$' and '$(A \vee (B \wedge C))$' do not have the same truth table
4. '$((A \to B) \to C)$' and '$(A \to (B \to C))$' do not have the same truth table

Also, check whether:

5. '$((A \leftrightarrow B) \leftrightarrow C)$' and '$(A \leftrightarrow (B \leftrightarrow C))$' have the same truth table

If you want additional practice, you can construct truth tables for any of the sentences and arguments in the exercises for the previous chapter.

Semantic Properties 11

In the previous section, we introduced the idea of a valuation and showed how to determine the truth value of any TFL sentence, on any valuation, using a truth table. In this section, we shall introduce some related ideas, and show how to use truth tables to test whether or not they apply.

11.1 Tautologies and contradictions

In §3, I explained *necessary truth* and *necessary falsity*. Both notions have surrogates in TFL. We shall start with a surrogate for necessary truth.

> ϕ is a TAUTOLOGY iff it is true on every valuation.

We can determine whether a sentence is a tautology just by using truth tables. If the sentence is true on every line of a complete truth table, then it is true on every valuation, so it is a tautology. In the example of §10, '$(H \wedge I) \rightarrow H$' is a tautology.

This is only, though, a surrogate for necessary truth. There are some necessary truths that we cannot adequately symbolise in TFL. An example is '$2 + 2 = 4$'. This *must* be true, but if we try to symbolise it in TFL, the best we can offer is an atomic sentence, and no atomic sentence is a tautology. Still, if we can adequately symbolise some English sentence using a TFL sentence which is a tautology, then that English sentence expresses a necessary truth.

We have a similar surrogate for necessary falsity:

> ϕ is a CONTRADICTION iff it is false on every valuation.

We can determine whether a sentence is a contradiction just by using truth tables. If the sentence is false on every line of a complete truth table, then it is false on every valuation, so it is a contradiction. In the example of §10, '$[(C \leftrightarrow C) \rightarrow C] \wedge \neg(C \rightarrow C)$' is a contradiction.

11.2 Tautological equivalence

Here is a similar, useful notion:

> ϕ and ψ are TAUTOLOGICALLY EQUIVALENT iff they have the same truth value on every valuation.

We have already made use of this notion, in effect, in §10.3; the point was that '$(A \wedge B) \wedge C$' and '$A \wedge (B \wedge C)$' are tautologically equivalent. Again, it is easy

to test for tautological equivalence using truth tables. Consider the sentences '$\neg(P \vee Q)$' and '$\neg P \wedge \neg Q$'. Are they tautologically equivalent? To find out, we construct a truth table.

P	Q	$\neg\ (P \vee Q)$	$\neg P \wedge \neg Q$
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 logical operators; negation for the first sentence, conjunction for the second. On the first three rows, both are false. On the final row, both are true. Since they match on every row, the two sentences are tautologically equivalent.

11.3 Consistency

In §3, I said that sentences are jointly consistent iff it is possible for all of them to be true at once. We can offer a surrogate for this notion too:

> $\phi_1, \phi_2, \ldots, \phi_n$ are JOINTLY TAUTOLOGICALLY CONSISTENT iff there is some valuation which makes them all true.

Derivatively, sentences are jointly tautologically inconsistent if there is no valuation that makes them all true. Again, it is easy to test for joint tautological consistency using truth tables.

11.4 Tautological entailment and validity

The following idea is closely related to that of joint consistency:

> The sentences $\phi_1, \phi_2, \ldots, \phi_n$ TAUTOLOGICALLY ENTAIL the sentence ψ if there is no valuation of the atomic sentences which makes all of $\phi_1, \phi_2, \ldots, \phi_n$ true and ψ false.

Again, it is easy to test this with a truth table. Let us check whether '$\neg L \rightarrow (J \vee L)$' and '$\neg L$' tautologically entail 'J', we simply need to check whether there is any valuation which makes both '$\neg L \rightarrow (J \vee L)$' and '$\neg L$' true whilst making '$J$' false. So we use 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**

The only row on which both'$\neg L \rightarrow (J \vee L)$' and '$\neg L$' are true is the second row, and that is a row on which 'J' is also true. So '$\neg L \rightarrow (J \vee L)$' and '$\neg L$' tautologically entail 'J'.

We now make an important observation:

> If $\phi_1, \phi_2, \ldots, \phi_n$ tautologically entail ψ, then $\phi_1, \phi_2, \ldots, \phi_n \therefore \psi$ is valid.

Here's why. If $\phi_1, \phi_2, \ldots, \phi_n$ tautologically entail ψ, then there is no valuation which makes all of $\phi_1, \phi_2, \ldots, \phi_n$ true whilst making ψ false. This means that it is *logically impossible* for $\phi_1, \phi_2, \ldots, \phi_n$ all to be true whilst ψ is false. But this is just what it takes for an argument, with premises $\phi_1, \phi_2, \ldots, \phi_n$ and conclusion ψ, to be valid!

In short, we have a way to test for the validity of English arguments. First, we symbolise them in TFL, as having premises $\phi_1, \phi_2, \ldots, \phi_n$, and conclusion ψ. Then we test for tautological entailment using truth tables.

11.5 The limits of these tests

We have reached an important milestone: a test for the validity of arguments! But, we should not get carried away just yet. It is important to understand the *limits* of our achievement. I shall illustrate these limits with three examples.

First, consider the argument:

1. Daisy has four legs. So Daisy has more than two legs.

To symbolise this argument in TFL, we would have to use two different atomic sentences – perhaps 'F' and 'T' – for the premise and the conclusion respectively. Now, it is obvious that 'F' does not tautologically entail 'T'. But the English argument surely seems valid!

Second, consider the sentence:

2. Jan is neither bald nor not-bald.

To symbolise this sentence in TFL, we would offer something like '$\neg J \wedge \neg\neg J$'. This a contradiction (check this with a truth-table). But sentence 2 does not itself seem like a contradiction; for we might have happily go on to add 'Jan is on the borderline of baldness'!

Third, consider the following sentence:

3. It's not the case that, if God exists, She answers malevolent prayers.

Symbolising this in TFL, we would offer something like '$\neg(G \rightarrow M)$'. Now, '$\neg(G \rightarrow M)$' tautologically entails 'G' (again, check this with a truth table). So if we symbolise sentence 3 in TFL, it seems to entail that God exists. But that's strange: surely even the atheist can accept sentence 3, without contradicting herself!

In different ways, these three examples highlight some of the limits of working with a language (like TFL) that can *only* handle truth-functional connectives. Moreover, these limits give rise to some interesting questions in philosophical logic. The case of Jan's baldness (or otherwise) raises the general question of what logic we should use when dealing with *vague* discourse. The case of the atheist raises the question of how to deal with the (so-called) *paradoxes of material implication*. Part of the purpose of this course is to equip you with the tools to explore these questions of *philosophical logic*. But we have to walk before we can run; we have to become proficient in using TFL, before we can adequately discuss its limits, and consider alternatives.

11.6 The double-turnstile

We are going to use the notion of tautological entailment rather a lot in this course. It will help us, then, to introduce a symbol that abbreviates it. Rather than saying that the TFL sentences $\phi_1, \phi_2, \ldots$ and ϕ_n together tautologically entail ψ, we shall abbreviate this by:

$$\phi_1, \phi_2, \ldots, \phi_n \vDash \psi$$

The symbol '$\vDash$' is known as *the double-turnstile*, since it looks like a turnstile with two horizontal beams.

But let me be clear. '$\vDash$' is not a symbol of TFL. Rather, it is a symbol of our metalanguage, augmented English (recall the difference between object language and metalanguage from §7). So the metalanguage sentence:

- $P, P \to Q \vDash Q$

is just an abbreviation for the English sentence:

- The TFL sentences 'P' and '$P \to Q$' tautologically entail 'Q'

Note that there is no limit on the number of TFL sentences that can be mentioned before the symbol '$\vDash$'. Indeed, we can even consider the limiting case:

$$\vDash \psi$$

This says that there is no valuation which makes all the sentences mentioned on the left side of '$\vDash$' true whilst making ψ false. Since *no* sentences are mentioned on the left side of '$\vDash$' in this case, this just means that there is no valuation which makes ψ false. Otherwise put, it says that every valuation makes ψ true. Otherwise put, it says that ψ is a tautology. Equally:

$$\phi \vDash$$

says that ϕ is a contradiction.

11.7 '$\vDash$' versus '$\to$'

I now want to compare and contrast '$\vDash$' and '$\to$'.

Observe: $\phi \vDash \psi$ iff there is no valuation of the atomic sentences that makes ϕ true and ψ false.

Observe: $\phi \to \psi$ is a tautology iff there is no valuation of the atomic sentences that makes $\phi \to \psi$ false. Since a conditional is true except when its antecedent is true and its consequent false, $\phi \to \psi$ is a tautology iff there is no valuation that makes ϕ true and ψ false.

Combining these two observations, we see that $\phi \to \psi$ is a tautology iff $\phi \vDash \psi$. But there is a really, really important difference between '$\vDash$' and '$\to$':

'$\to$' is a sentential connective of TFL.
'$\vDash$' is a symbol of augmented English.

Indeed, when '$\to$' is flanked with two TFL sentences, the result is a longer TFL sentence. By contrast, when we use '$\vDash$', we form a metalinguistic sentence that *mentions* the surrounding TFL sentences.

Practice exercises

A. Revisit your answers to §10**A**. Determine which sentences were tautologies, which were contradictions, and which were neither tautologies nor contradictions.

B. Use truth tables to determine whether these sentences are jointly consistent, or jointly inconsistent:

1. $A \to A$, $\neg A \to \neg A$, $A \wedge A$, $A \vee A$
2. $A \vee B$, $A \to C$, $B \to C$
3. $B \wedge (C \vee A)$, $A \to B$, $\neg(B \vee C)$
4. $A \leftrightarrow (B \vee C)$, $C \to \neg A$, $A \to \neg B$

C. Use truth tables to determine whether each argument is valid or invalid.

1. $A \to A \therefore A$
2. $A \to (A \wedge \neg A) \therefore \neg A$
3. $A \vee (B \to A) \therefore \neg A \to \neg B$
4. $A \vee B, B \vee C, \neg A \therefore B \wedge C$
5. $(B \wedge A) \to C, (C \wedge A) \to B \therefore (C \wedge B) \to A$

D. Answer each of the questions below and justify your answer.

1. Suppose that ϕ and ψ are tautologically equivalent. What can you say about $\phi \leftrightarrow \psi$?
2. Suppose that $(\phi \wedge \psi) \to \omega$ is neither a tautology nor a contradiction. What can you say about whether $\phi, \psi \therefore \omega$ is valid?
3. Suppose that ϕ, ψ and ω are jointly tautologically inconsistent. What can you say about $(\phi \wedge \psi \wedge \omega)$?
4. Suppose that ϕ is a contradiction. What can you say about whether $\phi, \psi \vDash \omega$?
5. Suppose that ψ is a tautology. What can you say about whether $\phi, \psi \vDash \omega$?
6. Suppose that ϕ and ψ are tautologically equivalent. What can you say about $(\phi \vee \psi)$?
7. Suppose that ϕ and ψ are *not* tautologically equivalent. What can you say about $(\phi \vee \psi)$?

E. Consider the following principle:

- Suppose ϕ and ψ are tautologically equivalent. Suppose an argument contains ϕ (either as a premise, or as the conclusion). The validity of the argument would be unaffected, if we replaced ϕ with ψ.

Is this principle correct? Explain your answer.

Truth table shortcuts 12

With practice, you will quickly become adept at filling out truth tables. In this section, I want to give you some permissible shortcuts to help you along the way.

12.1 Working through truth tables

You will quickly find that you do not need to copy the truth value of each atomic sentence, but can simply refer back to them. So you can speed things up by writing:

P	Q	$(P \vee Q) \leftrightarrow \neg P$
T	T	**F** F
T	F	**F** F
F	T	**T** T
F	F	**F** T

You also know for sure that a disjunction is true whenever one of the disjuncts is true. So if you find a true disjunct, there is no need to work out the truth values of the other disjuncts. Thus you might offer:

P	Q	$(\neg P \vee \neg Q) \vee \neg P$
T	T	F F **F** F
T	F	F T T **T** F
F	T	**T** T
F	F	**T** T

Equally, you know for sure that a conjunction is false whenever one of the conjuncts is false. So if you find a false conjunct, there is no need to work out the truth value of the other conjunct. Thus you might offer:

P	Q	$\neg (P \wedge \neg Q) \wedge \neg P$
T	T	**F** F
T	F	**F** F
F	T	T F **T** T
F	F	T F **T** T

A similar short cut is available for conditionals. You immediately know that a conditional is true if either its consequent is true, or its antecedent is false. Thus you might present:

P	Q	$((P \to Q) \to P) \to P$
T	T	**T**
T	F	**T**
F	T	T F **T**
F	F	T F **T**

So '$((P \to Q) \to P) \to P$' is a tautology. In fact, it is an instance of *Peirce's Law*, named after Charles Sanders Peirce.

12.2 Testing for validity and entailment

When we use truth tables to test for validity or entailment, we are checking for *bad* lines: lines where the premises are all true and the conclusion is false. Note:

- Any line where the conclusion is true is not a bad line.
- Any line where some premise is false is not a bad line.

Since *all* we are doing is looking for bad lines, we should bear this in mind. So: if we find a line where the conclusion is true, we do not need to evaluate anything else on that line: that line definitely isn't bad. Likewise, if we find a line where some premise is false, we do not need to evaluate anything else on that line.

With this in mind, consider how we might test the following for validity:

$$\neg L \to (J \vee L), \neg L \therefore J$$

The *first* thing we should do is evaluate the conclusion. If we find that the conclusion is *true* on some line, then that is not a bad line. So we can simply ignore the rest of the line. So at our first stage, we are left with something like:

J	L	$\neg L \to (J \vee L)$	$\neg L$	J
T	T			T
T	F			T
F	T	?	?	F
F	F	?	?	F

where the blanks indicate that we are not going to bother doing any more investigation (since the line is not bad) and the question-marks indicate that we need to keep investigating.

The easiest premise to evaluate is the second, so we next do that:

J	L	$\neg L \to (J \vee L)$	$\neg L$	J
T	T			T
T	F			T
F	T		F	F
F	F	?	T	F

Note that we no longer need to consider the third line on the table: it will not be a bad line, because (at least) one of premises is false on that line. And finally, we complete the truth table:

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

The truth table has no bad lines, so the argument is valid. (Any valuation on which all the premises are true is a valuation on which the conclusion is true.)

It might be worth illustrating the tactic again. Let us check whether the following argument is valid

$$A \vee B, \neg(A \wedge C), \neg(B \wedge \neg D) \therefore (\neg C \vee D)$$

At the first stage, we determine the truth value of the conclusion. Since this is a disjunction, it is true whenever either disjunct is true, so we can speed things along a bit. We can then ignore every line apart from the few lines where the conclusion is false.

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

We must now evaluate the premises. We use shortcuts where we can:

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

If we had used no shortcuts, we would have had to write 256 'T's or 'F's on this table. Using shortcuts, we only had to write 37. We have saved ourselves a *lot* of work.

I have been discussing shortcuts in testing for tautological validity. But exactly the same shortcuts can be used in testing for tautological entailment. By employing a similar notion of *bad* lines, you can save yourself a huge amount of work.

Practice exercises

A. Using shortcuts, determine whether each sentence is a tautology, a contradiction, or neither.

1. $\neg B \wedge B$
2. $\neg D \vee D$
3. $(A \wedge B) \vee (B \wedge A)$
4. $\neg[A \rightarrow (B \rightarrow A)]$
5. $A \leftrightarrow [A \rightarrow (B \wedge \neg B)]$
6. $\neg(A \wedge B) \leftrightarrow A$
7. $A \rightarrow (B \vee C)$
8. $(A \wedge \neg A) \rightarrow (B \vee C)$
9. $(B \wedge D) \leftrightarrow [A \leftrightarrow (A \vee C)]$

Partial truth tables 13

Sometimes, we do not need to know what happens on every line of a truth table. Sometimes, just a single line or two will do.

Tautology. In order to show that a sentence is a tautology, we need to show that it is true on every valuation. That is to say, we need to know that it comes out true on every line of the truth table. 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 false. Therefore, in order to show that some sentence is not a tautology, it is enough to provide a single valuation—a single line of the truth table—which makes the sentence false.

Suppose that we want to show that the sentence '$(U \wedge T) \rightarrow (S \wedge W)$' is *not* a tautology. We set up a PARTIAL TRUTH TABLE:

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

We have only left space for one line, rather than 16, since we are only looking for one line, on which the sentence is false. For just that reason, we have filled in 'F' for the entire sentence.

The main logical operator of the sentence is a conditional. In order for the conditional to be false, the antecedent must be true and the consequent must be false. 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

So we now have a partial truth table, which shows that '$(U \wedge T) \rightarrow (S \wedge W)$' is not a tautology. Put otherwise, we have shown that there is a valuation which makes '$(U \wedge T) \rightarrow (S \wedge W)$' false, namely, the valuation which makes 'S' false, 'T' true, 'U' true and 'W' false.

Contradiction. Showing that something is a contradiction requires a complete truth table: we need to show that there is no valuation which makes the sentence true; that is, we need to show that the sentence is false on every line of the truth table.

However, to show that something is *not* a contradiction, all we need to do is find a valuation which makes the sentence true, and a single line of a truth table will suffice. We can illustrate this with the same example.

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

To make the sentence true, it will suffice to ensure that the antecedent is false. Since the antecedent is a conjunction, we can just make one of them false. For no particular reason, we choose to make 'U' false; and then we can assign whatever truth value we like to the other atomic sentences.

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

Tautological equivalence. To show that two sentences are tautologically equivalent, we must show that the sentences have the same truth value on every valuation. So this requires a complete truth table.

To show that two sentences are *not* tautologically equivalent, we only need to show that there is a valuation on which they have different truth values. So this requires only a one-line partial truth table: make the table so that one sentence is true and the other false.

Consistency. To show that some sentences are jointly consistent, we must show that there is a valuation which makes all of the sentence true. So this requires only a partial truth table with a single line.

To show that some sentences are jointly inconsistent, we must show that there is no valuation which makes all of the sentence true. So this requires a complete truth table: You must show that on every row of the table at least one of the sentences is false.

Validity. To show that an argument is valid, we must show that there is no valuation which makes all of the premises true and the conclusion false. So this requires a complete truth table. (Likewise for entailment.)

To show that argument is *invalid*, we must show that there is a valuation which makes all of the premises true and the conclusion false. So this requires only a one-line partial truth table on which all of the premises are true and the conclusion is false. (Likewise for a failure of entailment.)

This table summarizes what is required:

	Yes	**No**
tautology?	complete truth table	one-line partial truth table
contradiction?	complete truth table	one-line partial 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
entailment?	complete truth table	one-line partial truth table

Practice exercises

A. Use complete or partial truth tables (as appropriate) to determine whether these pairs of sentences are tautologically equivalent:

1. A, $\neg A$
2. A, $A \vee A$
3. $A \rightarrow A$, $A \leftrightarrow A$
4. $A \vee \neg B$, $A \rightarrow B$
5. $A \wedge \neg A$, $\neg B \leftrightarrow B$
6. $\neg(A \wedge B)$, $\neg A \vee \neg B$
7. $\neg(A \rightarrow B)$, $\neg A \rightarrow \neg B$
8. $(A \rightarrow B)$, $(\neg B \rightarrow \neg A)$

B. Use complete or partial truth tables (as appropriate) to determine whether these sentences are jointly tautologically consistent, or jointly tautologically inconsistent:

1. $A \wedge B$, $C \rightarrow \neg B$, C
2. $A \rightarrow B$, $B \rightarrow C$, A, $\neg C$
3. $A \vee B$, $B \vee C$, $C \rightarrow \neg A$
4. A, B, C, $\neg D$, $\neg E$, F

C. Use complete or partial truth tables (as appropriate) to determine whether each argument is valid or invalid:

1. $A \vee \big[A \rightarrow (A \leftrightarrow A)\big] \therefore A$
2. $A \leftrightarrow \neg(B \leftrightarrow A) \therefore A$
3. $A \rightarrow B, B \therefore A$
4. $A \vee B, B \vee C, \neg B \therefore A \wedge C$
5. $A \leftrightarrow B, B \leftrightarrow C \therefore A \leftrightarrow C$

Chapter 4

Truth trees

Characteristic truth trees 14

As we saw in the last chapter, truth tables provide us with a mechanical procedure for determining whether a particular semantic concept (tautology, contradiction, tautological equivalence, consistency, or validity) is realized by any set of two or more sentences in TFL. We also saw that the method of truth tables has some drawbacks. First, complete truth tables become large and unwieldy as soon as there are more than three atomic sentences to evaluate. Second, as you've no doubt noticed, much of the 'calculation' involved in the construction of truth-tables must be completed 'in your head.' In other words, as you fill out a table for complex sentences you often must (repeatedly) apply the truth tables for the basic connectives to their constituents. Yet, you do this without writing anything down. It would be nice if all of these sub-procedures could be represented (perhaps graphically) in the end product. In this section, we will learn about a powerful and elegant method that does just this: truth trees.

Truth tables require us to assign all of the possible truth-values that the atomic constituents of complex sentences can take. However, as we witnessed in our motivation for turning to partial truth tables, this often provides more information than we need. When we employ this method, we aim to "build up" to the truth-values of the most complex sentences. Starting from the connective with the smallest scope, we work to the main connective and, in the final column, we see all the possible truth values for the compound statement.

Truth trees reverse this process and simplify it insofar as they only represent the assignments of True to sentences. On a truth tree, we "break down." Starting from the original statement, we ask at each step, "Under what conditions (or interpretations) would this statement be True?" Then, we write down those conditions in a systematic way. At each step, then, we only indicate when a statement would be 'True', NOT 'False'. When we complete a tree, we "read up" through the "branches" for the truth values that would make the statement true. So a big difference between tables and trees is that, while a table begins with showing all possible combinations of truth values, a tree shows those values that would make the statement true (if it is possible for it to be true).

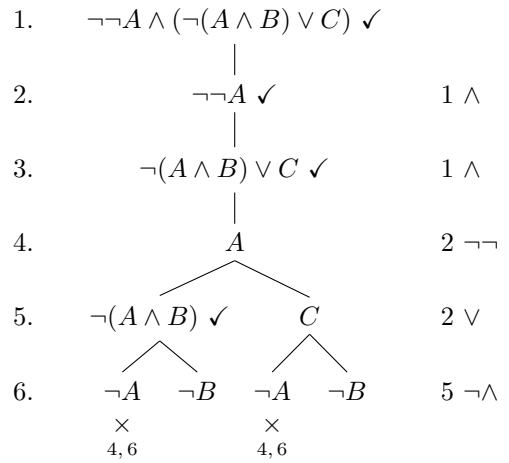
14.1 Anatomy of a Truth Tree

Truth trees are constructed using three columns. The left column lists numbers corresponding to the sentences in the tree. The central column contains the tree itself. The right column is for the justifications of sentences. The occurrence of a sentence in a tree is justified when it is the result of applying one of the

tree rules to a sentence already in the tree. The right-most column therefore contains a rule and the line number of the sentence to which it was applied.

Unlike conventional trees, truth trees grow downwards. The initial set of sentences at the top form the ROOT. We apply tree rules to these sentences to grow the tree. The tree rules represent the different conditions under which a type of sentence in TFL is true. By applying these rules, we are able to decompose complex sentences into simpler ones. There are 9 decomposition rules, each applying to a specific type of sentence. The rules either tell us to 'stack' new sentences, one on top of another, or to 'branch' them. Once a rule has been applied to a sentence the sentence is checked (✓) to indicate that we're done with it.

The following is an example of a completed truth tree with one initial sentence, namely, $\neg\neg A \wedge (\neg(A \wedge B) \vee C)$.



The characteristic feature of truth trees is that they have branches. A BRANCH is the set of sentences obtained by starting at the bottom of the tree and reading upward through the tree along a series of lines. For instance, the tree above has four branches, the right-most of which consists in the following: $\{\neg B, C, A, \neg(A \wedge B) \vee C, \neg\neg A, \neg\neg A \wedge (\neg(A \wedge B) \vee C)\}$. The initial sentence is true if every sentence in this set is true. So, a branch provides the same information as a row in a truth table where T appears below the main operator of the initial sentence(s).

The sentences at the end of each branch are called LEAVES. A COMPLETED BRANCH is a branch all of whose complex sentences have been checked and whose leaf is either an atomic sentence or the negation of an atomic sentence. Such sentences (atoms and their negations) are called LITERALS. A COMPLETED TREE is a tree whose branches are all completed.

Each branch of a completed tree is supposed to represent a truth value assignment—a row of a truth table. But what we want isn't always what we get. Recall that for each row of a truth table, all occurrences of the same sentence letter get the same truth value. In other words, you can't have a truth value assignment where a sentence is both true and false. On a truth tree we represent the falsity of a sentence by its negation. If a sentence, A, appears on a branch, then that branch assigns TRUE to A. If $\neg A$ appears on

a branch, the branch assigns FALSE to A. If both a sentence and its negation appear on a branch, then that branch has not succeeded in representing a truth value assignment; so we chop it off, and put an $\times$ under it, indicating that the branch is CLOSED. Under the $\times$ we write the line numbers of the contradictory sentences.

In the tree above, the first and third branches (left to right) are closed, since they contain an atom and its negation. But the second and fourth branches are not closed. A branch that is not closed is said to be OPEN. A COMPLETED OPEN BRANCH is a completed branch that does not have a $\times$ below its leaf. A tree is a COMPLETED OPEN TREE if and only if it contains at least one completed open branch. A CLOSED TREE is a completed tree all of whose branches are closed.

This terminology will become very important when we begin to use trees to determine the semantic properties—such as consistency, validity, and tautological equivalence—that apply to sets of sentences. Before we do that, we must learn how to grow a truth tree by applying the tree rules.

Truth Tree Rules 15

15.1 Conjunction.

Let's start with the tree rule for conjunction. If you're using a tree to analyze a set of sentences, you always begin the same way. Simply write the sentence(s) at the top of a piece of paper. This is the ROOT of the tree. We'll use $(A \wedge B) \wedge C$ as our example. Next, identify the main operator, which in this case is the conjunction. Now ask yourself, *under what condition(s) would this statement be true?* For a conjunction, we know there is only one condition under which it would be true: when both sides of the conjunction are true. You can confirm this by recalling the truth table for conjunction:

ϕ	ψ	$\phi \wedge \psi$
T	T	T
T	F	F
F	T	F
F	F	F

Since we only want to write down true sentences on the tree (unlike a table), beneath the initial sentence, write down the two conjuncts *stacked* on top of one another; this indicates that both conjuncts are true when the conjunction as a whole is true. In this way, the rule just represents the information given in the truth table for conjunction: a conjunction is true when both conjuncts are truth and false otherwise. After you've applied the rule, put a check beside the formula you just worked on:

1. $(A \wedge B) \wedge C$ ✓
2. $A \wedge B$
3. C

Again, the check marks indicate that you've completed answering the question, "Under what conditions would this statement be true?" You don't have to revisit this sentence.

Now look at your tree again and see whether you have any complex sentences remaining. If you do, then repeat the process:

1. $(A \wedge B) \wedge C$ ✓
2. $A \wedge B$ ✓
3. C
4. A
5. B

Now, are there any complex sentences left? No. This means your tree is completed. Now read back up the branch: what you've shown is that $(A \wedge B) \wedge C$ is true if and only if A, B, and C are ALL true.

The tree for $(A \wedge B) \wedge C$ represents exactly the same information as the corresponding truth table:

A	B	C	$(A \wedge B) \wedge C$
T	T	T	T T T **T** T
T	T	F	T T T **F** F
T	F	T	T F F **F** T
T	F	F	T F F **F** F
F	T	T	F F T **F** T
F	T	F	F F T **F** F
F	F	T	F F F **F** T
F	F	F	F F F **F** F

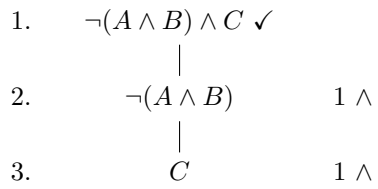
The table tells us that $(A \wedge B) \wedge C$ is only true when A, B, and C are all true—look at the first line! But that's exactly what the tree tells us since there is only one line below the root and it contains A, B, and C.

We can generalize from this example to formulate the following tree rule.

i	$\phi \wedge \psi$ ✓	
	|	
j	ϕ	$i\wedge$
	|	
k	ψ	$i\wedge$

We read this rule as saying that if an unchecked sentence of the form $\phi \wedge \psi$ occurs in a tree, then you should write ϕ stacked on top of ψ below each open uncompleted branch. Then check the original sentence. In this rule, 'i' simply stands for the number of the line with the conjunction on it. When we apply this rule, we write that number followed by '$\wedge$' to indicate that we've applied the tree rule for conjunction to that sentence. The $\wedge$ rule is an example of a *stacking* rule.

Negated conjunctions are handled differently. Consider the following variation on the above statement: $\neg(A \wedge B) \wedge C$. Once again, we begin by placing the sentence at the top and then breaking it down into its two conjuncts:



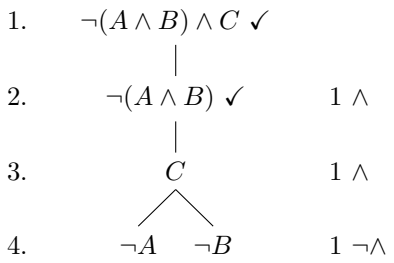
But now we must ask: under what conditions would $\neg(A \wedge B)$ be true? Since the negation is the main connective, in order for $\neg(A \wedge B)$ to be true, $A \wedge B$ would have to be false. So, what conditions would make $A \wedge B$ false? Again, our basic truth tables tell us that a conjunction is false if *either* conjunct is false or both are (rows 2, 3, and 4 in the table below).

ϕ	ψ	$\neg$	$(\phi \wedge \psi)$
T	T	**F**	T
T	F	**T**	F
F	T	**T**	F
F	F	**T**	F

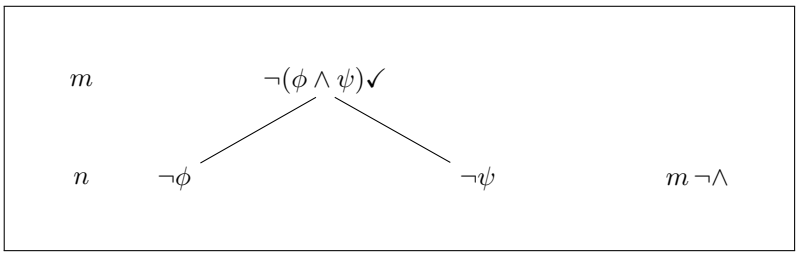
So, we need to have a way of representing the three different ways in which it could be false. We do this by *branching*.

Branching is when you draw one line out from the tree going to the left, and then another line out to the right. This indicates that you have at least two possible interpretations for what you're working on. Branching is necessary, but it obviously creates a larger, more complex tree than if you simply placed everything under the root of the tree. This is further complicated if you have any untouched complex sentences above the branch. When you break those down you have to make sure that you've placed the new sentences under each of the branches. For these reasons, when you're working on a tree, you usually put off branching as long as you can.

For our tree, we have to branch at this point; we have no choice. One branch covers the situation in which A is false, thereby making $A \wedge B$ false, and the other branch covers the situation in which B is false, again thereby making $A \wedge B$ false. The case when both A *and* B are false is captured indirectly by the fact that both branches are open, i.e. ready to receive more sentences below them. So even though there three ways of making the formula false, we only need two branches to represent this, since we can think of the two branches *together* as representing the third possibility. Trees never need more than two lines coming out from any sentence or 'node'. Each branch corresponds to a line on the truth table on which the formula you're decomposing is true.



The tree rule for negated conjunctions is as follows:



This is a *branching rule*. It tells us that if we encounter an unchecked sentence of the form $\neg(\phi \wedge \psi)$ in our tree, we should branch each open branch with two formulas: $\neg\phi$ and $\neg\psi$.

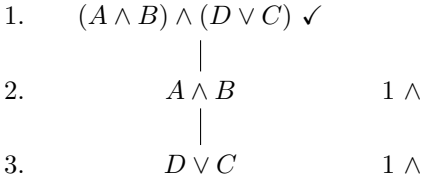
In the example above we've broken down all the complex sentences and have generated two different interpretations that would make the original sentence true. "Reading up" both branches we can see that the original statement is true when C is true and A is false *or* when C is true and B is false. Moreover, since either of these makes the statement true, you can recognize a third interpretation in which both A and B are false, and C is true and the formula would also come out as true.

So, when doing a truth tree, if you have a conjunction, put both the conjuncts underneath it. When you have a negated conjunction, draw two branches, and put one negated conjunct on each branch. Remember to put a check by the complex sentence once you have broken it down.

15.2 Disjunction

Since disjunctions are true if either of the two disjuncts is true, we'll need to branch in order to capture both scenarios. Consider the following conjunction that includes a disjunction as part of it: $(A \wedge B) \wedge (D \vee C)$.

We'll begin in the same manner as before.



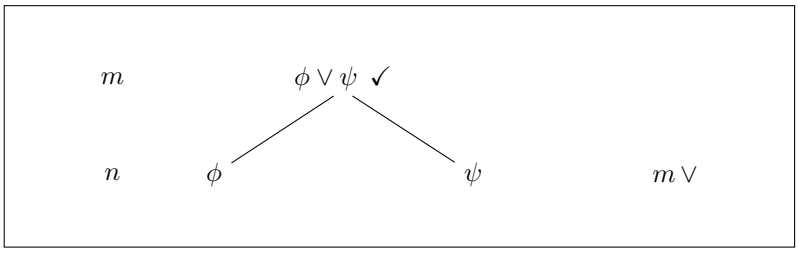
Now we have a choice of which sentence we want to work with first: the conjunction or the disjunction. As we pointed out a moment ago, you should put off branching as long as you can, so we begin with the conjunction.

1.	$(A \wedge B) \wedge (D \vee C)$ ✓	
2.	$A \wedge B$ ✓	1 $\wedge$
3.	$D \vee C$	1 $\wedge$
4.	A	2 $\wedge$
5.	B	2 $\wedge$

Now we turn to the disjunction. Ask yourself under what circumstances $D \vee C$ would be true. It would be true if either D or C were true. Since these are two distinct possibilities, we must proceed by branching.

1.	$(A \wedge B) \wedge (D \vee C)$ ✓	
2.	$A \wedge B$ ✓	1 $\wedge$
3.	$D \vee C$ ✓	1 $\wedge$
4.	A	2 $\wedge$
5.	B	2 $\wedge$
6.	$D \quad C$	3 $\vee$

Because we've broken down all the compound sentences we have completed our truth tree. We can easily see here that the original conjunction is true when D, B, and A are true *or* when C, B, and A are true. Here is the tree-rule for disjunctions:



As with the rules for conjunction, we must remember that this rule tells us to add the branching of ϕ and ψ to each open branch on the tree.

Naturally, constructing a tree for a negated disjunction is different. While disjunction is a branching rule, negated disjunction is a stacking rule. As always you have to begin with the main operator, which in this case is a conjunction. We proceed as before, breaking down the conjunctions first.

1.	$(A \vee B) \wedge \neg(D \vee C)$ ✓	
	|	
2.	$A \vee B$	$1 \wedge$
	|	
3.	$\neg(D \vee C)$	$1 \wedge$

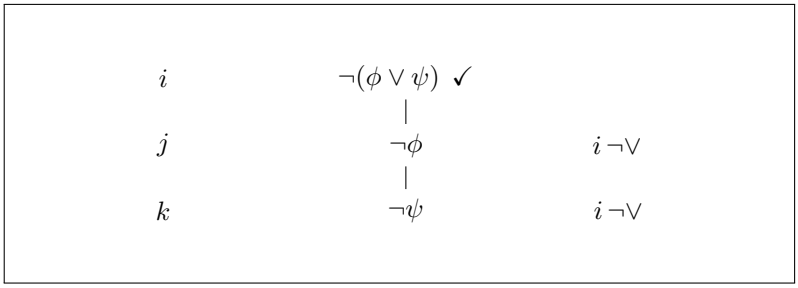
Now, we know that $A \vee B$ is a disjunction and that in order to break it down we will have to branch. But we also know that we should put off branching as long as possible. So, let's deal with $\neg(D \vee C)$ first. Once again, ask yourself, under what circumstances would $\neg(D \vee C)$ be true? It would be true if $D \vee C$ is false. So, under what conditions is $D \vee C$ false? We know that the only way that a disjunction comes out false is when *both* the disjuncts are false. So, you'll have to *stack* a negated D and a negated C directly under the main branch, and then check off the formula.

1.	$(A \vee B) \wedge \neg(D \vee C)$ ✓	
	|	
2.	$A \vee B$	$1 \wedge$
	|	
3.	$\neg(D \vee C)$ ✓	$1 \wedge$
	|	
4.	$\neg D$	$3 \neg\vee$
	|	
5.	$\neg C$	$3 \neg\vee$

We can now go back and apply the disjunction rule to $A \vee B$.

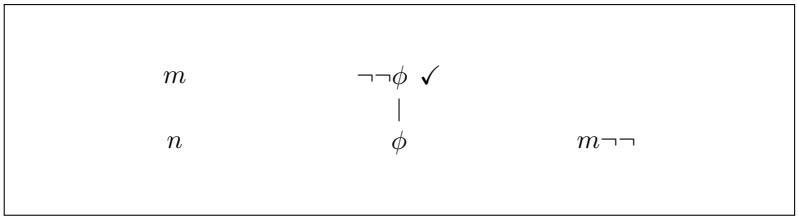
1.	$(A \vee B) \wedge \neg(D \vee C)$ ✓	
	|	
2.	$A \vee B$ ✓	$1 \wedge$
	|	
3.	$\neg(D \vee C)$ ✓	$1 \wedge$
	|	
4.	$\neg D$	$3 \neg\vee$
	|	
5.	$\neg C$	$3 \neg\vee$
	/\	
6.	A B	$2 \vee$

So, the rule for negated disjunctions tells you to negate both the disjuncts and stack them both on the relevant branches. We can now see that conditions under which the original statement is true are when A is true and C and D are false *or* when B is true, and C and D are false.

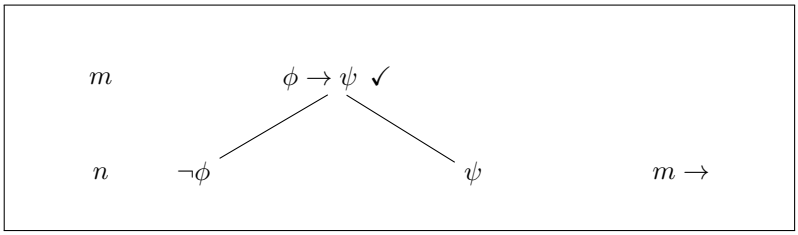


15.3 Double Negation and the Conditional.

Consider the following formula: $\neg\neg(A \vee B)$. What is its main operator? The $\vee$? The inner $\neg$? The outer $\neg$? Since the main operator is defined as the operator that has the widest scope, and the outer $\neg$ is the one operating over the rest of the formula, that makes it the main operator. The question then becomes, what do we do with a negated negation in a tree? We can't simply leave it this way. Recall that when we introduced trees we said that you were done breaking down a formula when you had reduced it to a literal, i.e. an atomic formula or its negation. Fortunately, we have a rule that lets us break down a double negation and it's a relatively simple one: Whenever you have a double negation, drop the negations, write down the remaining statement, and then check off the original statement.



Next, we turn to conditionals. Like disjunctions, conditionals also require branching, since a conditional is true when either the antecedent is false or the consequent is true. Conversely, a negated conditional does not branch, since the only way that a conditional comes out false is if the antecedent is true and the consequent is false. Here are the rules for conditionals:



i	$\neg(\phi \to \psi)$ ✓	
j	ϕ	$i\,\neg\,\to$
k	$\neg\psi$	$i\,\neg\,\to$

Here is a sentence that incorporates both:

1.	$(A \to B) \wedge \neg(A \to C)$ ✓	
2.	$A \to B$	$1\,\wedge$
3.	$\neg(A \to C)$	$1\,\wedge$

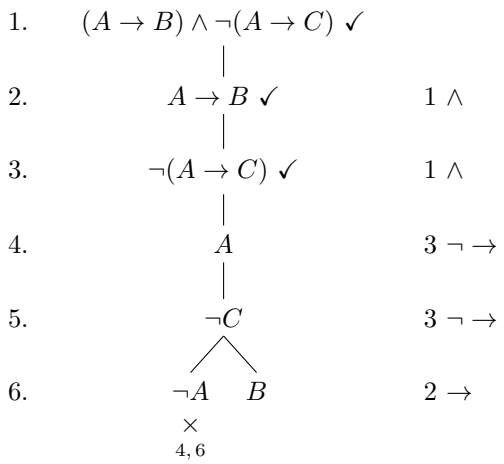
Since we have two formulas we need to work with, the first question is whether one of these can be done without branching. Since the second one is a negated conditional, and since they don't branch, we begin with that one.

1.	$(A \to B) \wedge \neg(A \to C)$ ✓	
2.	$A \to B$	$1\,\wedge$
3.	$\neg(A \to C)$ ✓	$1\,\wedge$
4.	A	$3\,\neg\,\to$
5.	$\neg C$	$3\,\neg\,\to$

Now we break down the last remaining complex sentence.

1.	$(A \to B) \wedge \neg(A \to C)$ ✓	
2.	$A \to B$ ✓	$1\,\wedge$
3.	$\neg(A \to C)$ ✓	$1\,\wedge$
4.	A	$3\,\neg\,\to$
5.	$\neg C$	$3\,\neg\,\to$
6.	$\neg A \quad B$	$2\,\to$

We're not done yet. Notice that A appears in the tree and $\neg A$ appears in the left leaf. This means that that branch is CLOSED. To represent this fact, we need to place an $\times$ under $\neg A$ and write down the line numbers of the contradictory literals.



There is only one completed open branch on this tree—namely, the one whose leaf is B. If we read up that branch and write down all of the literals in it, we get the following set: $\{B, \neg C, A\}$. From this, we know that the formula $(A \to B) \wedge \neg(A \to C)$ is true when A is true, B is true, and C is false.

Remember that each branch corresponds to a line on the truth table on which the formula you're decomposing is true. Here's the truth table for $(A \to B) \wedge \neg(A \to C)$.

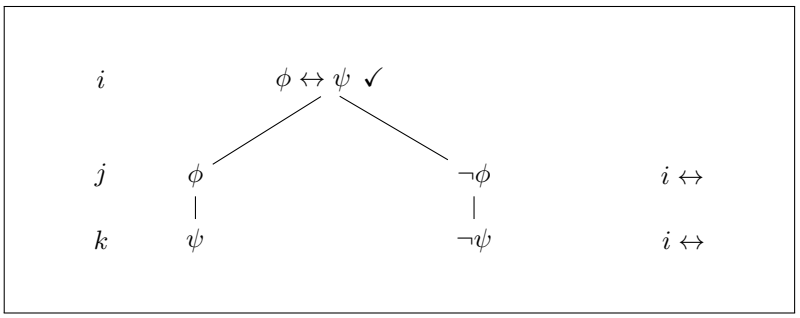
A	B	C	$(A \to B)$	$\wedge$	$\neg$	$(A \to C)$
T	T	T	T	**F**	F	T
T	T	F	T	**T**	T	F
T	F	T	F	**F**	F	T
T	F	F	F	**F**	T	F
F	T	T	T	**F**	F	T
F	T	F	T	**F**	F	T
F	F	T	T	**F**	F	T
F	F	F	T	**F**	F	T

We can see that the only time the formula is true is when A is true, B is true, and C is false (look at the second line on the table). But this is just what we learned from our tree! So we can always recover a truth table from a truth tree. If we were asked whether $(A \to B) \wedge \neg(A \to C)$ is a contradiction, we could quickly answer "No" by providing a one-line partial truth table that just contained the second line on the table above. This is just another way in which truth trees and truth tables provide the same information.

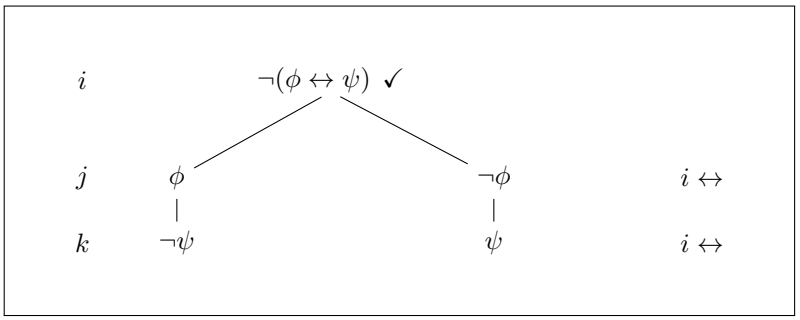
15.4 Biconditionals

Biconditionals, like disjunctions, have two different scenarios which make them come out true. Recall that a biconditional is true whenever the sentences on

either side of it are both true or both false. Therefore, you know that in a case where the biconditional is the main operator, you're going to have to branch right off the bat.



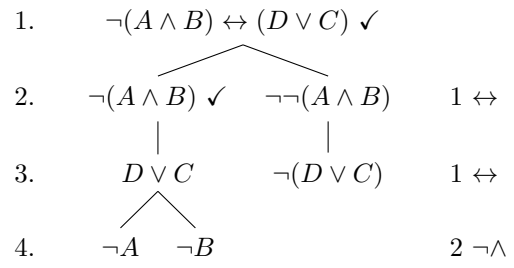
A biconditional is false whenever both sides have different truth values. Again, there are at least two possible situations to consider. So, the tree rule for negated biconditional will branch as well.



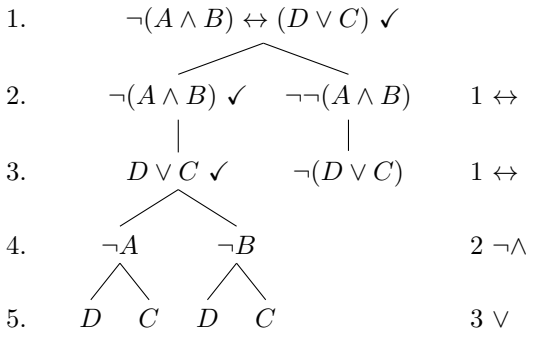
Let's work through an example. Construct a tree for the sentence $\neg(A \wedge B) \leftrightarrow (D \vee C)$. We start with the main operator, which is a biconditional.

1.	$\neg(A \wedge B) \leftrightarrow (D \vee C)$ ✓		
2.	$\neg(A \wedge B)$	$\neg\neg(A \wedge B)$	$1 \leftrightarrow$
3.	$D \vee C$	$\neg(D \vee C)$	$1 \leftrightarrow$

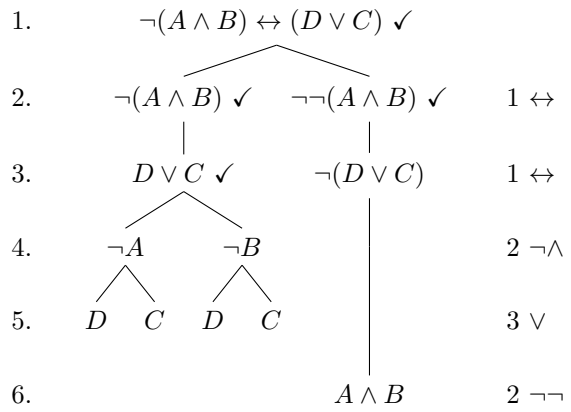
We now have two branches, each of which has complex sentences in it that need to be decomposed. Let's start with the left one. Since the first sentence is a negated conjunction and the second is a disjunction, we know that we'll have to branch no matter which we start with, so let's start with the top most sentence, i.e. $\neg(A \wedge B)$.



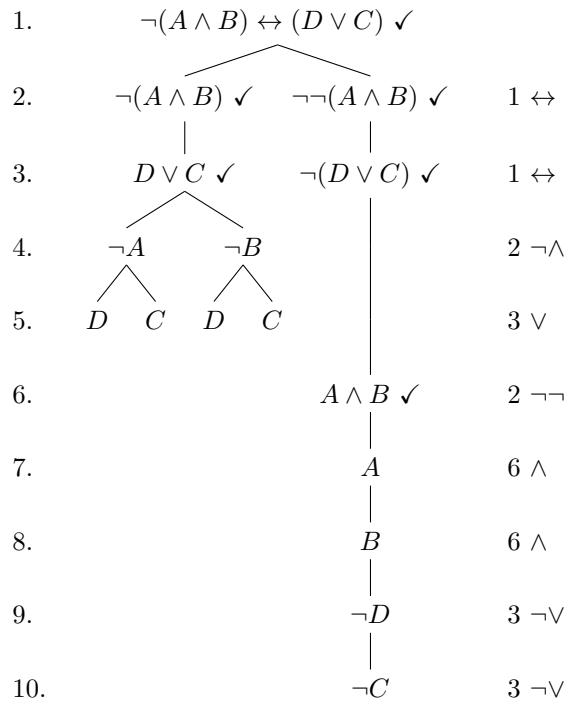
Now we'll break down $D \vee C$ on the left side branches.



Next, let's turn to $\neg\neg(A \wedge B)$ on the right side.



Finally, let's break down the last remaining complex sentences in the rightmost branch, i.e. $A \wedge B$ and $\neg(D \vee C)$. Neither requires branching, so it doesn't matter which one when decompose first.



None of the branches contain contradictory literals, so the tree is a completed open tree.

15.5 Truth Trees Strategies

Using the decomposition rules blindly will ultimately lead to a completed open or a closed tree. For example, for a tree involving $A \wedge B$ and $A \rightarrow B$, it is equally acceptable to begin by decomposing either of the two propositions. However, a strategic use of these rules will lead to the same result in a timelier manner.

There are four such strategic rules:

1. Use no more rules than needed.
2. Use rules that close branches.
3. Use stacking rules before branching rules.
4. Decompose more complex propositions before simpler propositions.

Practice exercises

A. Provide truth trees for each of the following:

1. $A \rightarrow A$
2. $C \rightarrow \neg C$
3. $(A \leftrightarrow B) \rightarrow \neg(A \leftrightarrow \neg B)$

4. $(A \to B) \vee (B \to A)$
5. $(A \wedge B) \to (B \vee A)$
6. $\neg(A \vee B) \leftrightarrow (\neg A \wedge \neg B)$
7. $\big[(A \wedge B) \wedge \neg(A \wedge B)\big] \wedge C$
8. $[(A \wedge B) \wedge C] \to B$
9. $\neg\big[(C \vee A) \vee B\big]$

Truth Trees and Semantic Properties 16

16.1 Consistency

As with truth tables, truth trees are a powerful device for determining whether a set of sentences exhibits a particular semantic property—i.e. consistency, inconsistency, tautological equivalence, tautological entailment, or validity—as well as for determining whether a single sentence is a tautology or a contradiction. Trees, however, are a much faster and more efficient method for answering these questions. Recall that in doing complete truth tables we were reconstructing truth values for a sentence or set of sentences on every possible valuation—and that in doing so a good deal of our work was wasted. In testing for consistency, for example, we were just looking for a row of the truth table in which all the sentences were true. If we found such a row all the other rows were irrelevant. But we had to do the complete truth table because if there was no row in which all the sentences were true we needed to determine that. Getting a 'yes' answer to the consistency question was easy: just find a row in which all the sentences get T. Getting a 'no' answer was tedious: we had to consider all the possibilities—and some of them were clearly irrelevant.

In contrast, truth trees systematically select all and only those truth value assignments or valuations in which each sentence individually is true and put them together, so that we can determine whether there's a valuations on which all of them are true together. To do this, we grow a tree with branches that represent valuations that make the sentence(s) we're testing true according to tree rules that represent the ways of making each of the sentences true.

Since a COMPLETED OPEN BRANCH tells us that there is a way to assign truth values that would make all of the sentences in the branch true, a completed open branch tells us that the stack of sentences (and all of those in the branch for that matter) are consistent.

$\phi_1, \phi_2, \ldots, \phi_n$ are CONSISTENT iff the truth tree whose ROOT is $\phi_1, \phi_2, \ldots, \phi_n$ has at least one COMPLETED OPEN BRANCH.

16.2 Inconsistency

Demonstrating inconsistency with trees is also quite easy. Recall that a CLOSED BRANCH represents a valuation on which the sentences in the root contradict one another—i.e. each closed branch represents one way in which all the sentences in the root are not true. When *all* the branches on a tree are closed,

the tree is closed, and hence there is no valuation that makes the sentences in the root true. So, we can define inconsistency according to trees as follows.

> $\phi_1, \phi_2, \ldots, \phi_n$ are INCONSISTENT iff the truth tree whose ROOT is $\phi_1, \phi_2, \ldots, \phi_n$ is CLOSED.

Obviously, we can apply this definition to single sentences to yield a tree-based definition for CONTRADICTION.

16.3 Tautology and Equivalence

We know that a sentence is a tautology just in case it is true on any possible valuation of its constituents. Moreover, a sentence is false whenever its negation is true. So, if ϕ is a tautology, then $\neg\phi$ is never true! This provides the basis for a straightforward tree-based definition of tautology.

> ϕ is a TAUTOLOGY iff the truth tree whose ROOT is $\neg\phi$ is CLOSED.

Now for two sentences to be tautologically equivalence they must always agree in their truth values. A biconditional statement is only true when the sentences on either side agree in truth value. So, two sentences, ϕ and ψ, are tautologically equivalent just in case, $\phi \leftrightarrow \psi$ is a tautology. Using our tree-based definition for TAUTOLOGY yields the following for equivalence:

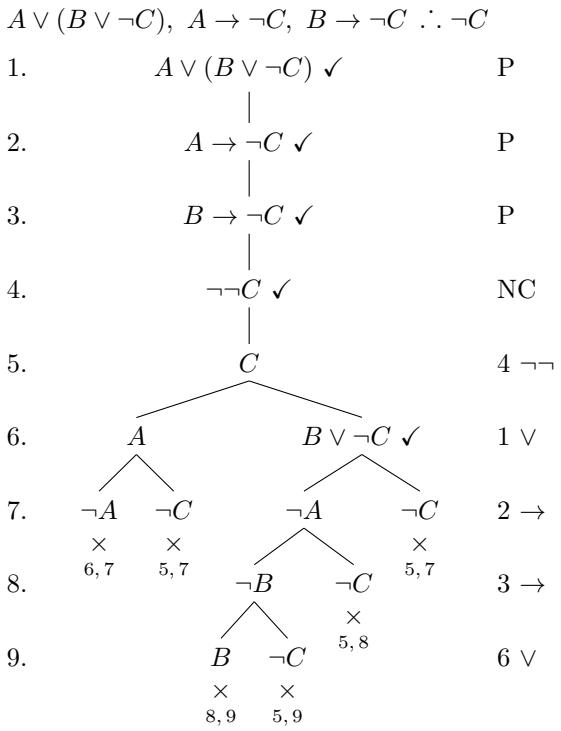
> ϕ and ψ are TAUTOLOGICALLY EQUIVALENT iff the truth tree whose ROOT is $\neg(\phi \leftrightarrow \psi)$ is CLOSED.

16.4 Validity and Entailment

The power and efficiency of trees really shines when it comes to validity and entailment. Recall that one set of sentences entails another sentence just in case the latter is never false when the former are (all) true.

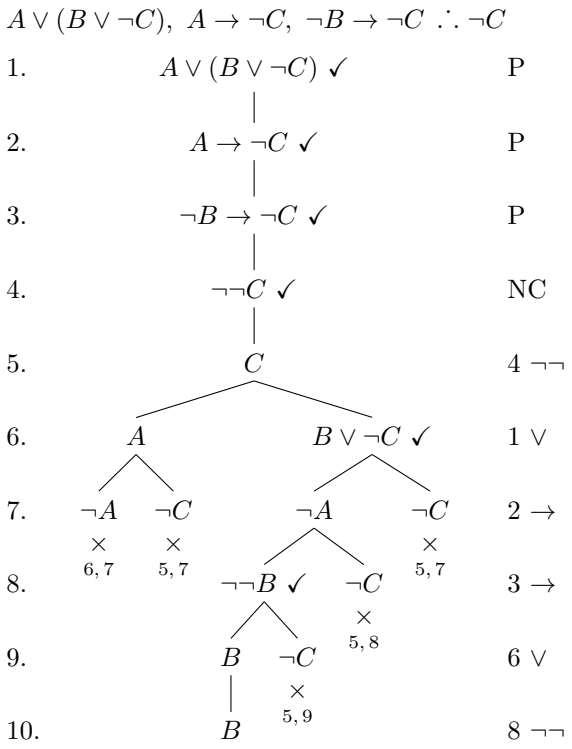
> The sentences $\phi_1, \phi_2, \ldots, \phi_n \vDash \psi$ if the truth tree whose ROOT is $\phi_1, \phi_2, \ldots, \phi_n, \neg\psi$ is CLOSED.

And, of course, an argument is valid when its premises entail its conclusion. Here's an example of how we use truth trees to test arguments for validity: $A \vee (B \vee \neg C)$, $A \rightarrow \neg C$, $B \rightarrow \neg C$ $\therefore \neg C$. In order to determine whether this argument is valid, we construct a tree with a root comprised of the premises (labeled 'P') and the negation of the conclusion (labeled 'NC').



Since the tree closes, we know that it is impossible for the premises to be true but the conclusion false. That means the argument is valid.

We saw in the section 15 that truth trees and truth tables represent the same information and that from a tree we can always recover a table. This connection between the two methods can be very helpful when determining the semantic properties of a set of sentences. For instance, suppose we wanted to know whether the following argument is valid (it's very similar to, but different from the one above; see if you can spot the difference!): $A \vee (B \vee \neg C),\ A \to \neg C,\ \neg B \to \neg C \ \therefore \neg C$



Unlike the previous tree, this one does not close. The third branch from the left, i.e. the one whose leaf is B does not have a $\times$ under it. As we know, an open branch represents a valuation on which the sentences in the root are all true. But since our root was composed of the premises and the *negation* of the conclusion, this open branch tells us that it is possible for the premises to be true and the conclusion to be *false*. But if *that* is possible, then the argument isn't valid. The open branch thus represents a COUNTEREXAMPLE to the argument—i.e. an instance in which the premises are true but the conclusion is false.

In Chapter 3 we learned that to show that an argument is invalid, we only need to provide a one-line partial truth table on which the premises are true but the conclusion is false. We can recover such a table by looking at our open branch and writing down each literal that appears on it, omitting repeated literals. If we do so, we get the following set $\{\neg A, B, C\}$. From this, we can conclude that the premises of our argument are true and its conclusion false when A is false, B is true, and C is true. In other words, we have found a counterexample.

To recover the table, we just write down those values for the atoms that correspond to the literals on our open branch, i.e. A is F, B is T, and C is T.

A	B	C	$A \vee (B \vee \neg C)$	$A \to \neg C$	$\neg B \to \neg C$	$\neg C$
F	T	T	T	T	T	F

We thus have our one-line partial truth table showing that the argument is invalid.

16.5 Semantic Properties and what to look for in a Tree

We've now seen how truth trees provide a powerful and efficient mechanism for determining semantic properties. In order to use this method to answer informal or semi-formal questions about the semantic properties of sentences, you'll need to know which tree to construct for each type of question and what to look for in the tree so as to answer the question correctly. If you've understood the section up to now, you won't learn anything new below. But it may help to reinforce some ideas.

Take, for instance, a question of the form "Are $\phi_1, \ldots, \phi_n$ inconsistent?". In formal terms, the question is whether $\phi_1, \ldots, \phi_n \vDash$ is true. To determine the answer, we simply write each sentence or formula on a line. That's the root of our tree. Then we construct the tree. If the tree closes—i.e. every branch has an '$\times$' under it—then we know that there is no valuation that makes all of $\phi_1, \ldots, \phi_n$ true. So, if the tree closes, $\phi_1, \ldots, \phi_n \vDash$ is true and thus $\phi_1, \ldots, \phi_n$ are inconsistent.

Consider another example. Suppose we want to know whether $\phi_1, \ldots, \phi_n$ entails ψ. In formal terms we're asking whether $\phi_1 \ldots \phi_n \vDash \psi$ is true. But recall that an entailment is true just in case there is no valuation that makes the left-hand side (i.e. $\phi_1 \ldots \phi_n$) true but the right-hand side (i.e. ψ) false. So, an entailment statement is true if there is no valuation that makes the left-hand side (i.e. $\phi_1 \ldots \phi_n$) true and the *negation of* the right-hand side (i.e. $\neg\psi$) also true. But this just means that an entailment statement is true if the left-hand side and the negation of the right-hand side are inconsistent. So, we can just as well ask whether $\phi_1 \ldots \phi_n, \neg\psi$ are inconsistent. If they are, then the entailment holds; if they aren't (i.e. they are consistent), then it is possible for the right-hand side to be true but the left-hand side to be false, so the entailment doesn't hold. This means that we can formalize the question in either of two ways: $\phi_1 \ldots \phi_n \vDash \psi$? or $\phi_1, \ldots, \phi_n, \neg\psi \vDash$? The latter formulation is a bit more helpful because we know exactly what tree to construct, namely a tree whose root is $\phi_1 / \ldots / \phi_n / \neg\psi$ where each '/' indicates that the next sentence goes on the next line on the tree. Now we construct the tree. If it closes, then we know that $\phi_1 \ldots \phi_n, \neg\psi$ are inconsistent and thus that the entailment statement holds, i.e. $\phi_1 \ldots \phi_n \vDash \psi$. If there is an open branch, however, then we know that $\phi_1 \ldots \phi_n, \neg\psi$ are consistent and thus that the entailment doesn't hold.

Below we provide a table that summarizes these moves, starting with an informal question, followed by its formalization, followed by the root of the tree we need to construct, and ending with how to answer the original question given what the tree looks like (i.e. closed or not).

Informal	Formal	Root	Closed Tree	Open Tree
Is ϕ a tautology?	$\vDash \phi$?	$\neg\phi$	yes	no
Is ϕ a contradiction?	$\phi \vDash$?	ϕ	yes	no
Are ϕ and ψ equivalent?	$\vDash \phi \leftrightarrow \psi$?	$\neg(\phi \leftrightarrow \psi)$	yes	no
Are $\phi_1, \ldots, \phi_n$ consistent?	$\phi_1, \ldots, \phi_n \nvDash$?	$\phi_1 / \ldots / \phi_n$	no	yes
Are $\phi_1, \ldots, \phi_n$ inconsistent?	$\phi_1, \ldots, \phi_n \vDash$?	$\phi_1 / \ldots / \phi_n$	yes	no
Does $\phi_1, \ldots, \phi_n$ entail ψ?	$\phi_1 \ldots \phi_n \vDash \psi$? or $\phi_1, \ldots, \phi_n, \neg\psi \vDash$?	$\phi_1 / \ldots / \phi_n / \neg\psi$	yes	no
Is $\phi_1, \ldots, \phi_n \therefore \psi$ valid?	$\phi_1, \ldots, \phi_n \therefore \psi$? or $\phi_1, \ldots, \phi_n, \neg\psi \vDash$?	$\phi_1 / \ldots / \phi_n / \neg\psi$	yes	no

Practice exercises

A. Revisit your answers to §14**A**. Determine which sentences were tautologies, which were contradictions, and which were neither tautologies nor contradictions.

B. Use truth trees to determine whether these sentences are jointly consistent, or jointly inconsistent:

1. $A \leftrightarrow (B \vee C)$, $C \rightarrow \neg A$, $A \rightarrow \neg B$
2. $A \vee (\neg B \rightarrow \neg C)$, $\neg\big[(C \leftrightarrow B) \vee (A \vee \neg C)\big]$
3. $A \leftrightarrow (B \wedge C)$, $(A \vee (\neg B \wedge \neg C))$, $\neg(A \leftrightarrow B)$
4. $A \rightarrow (B \wedge \neg C)$, $\neg C \rightarrow (B \wedge A)$, $\neg(A \leftrightarrow C)$
5. $(A \rightarrow B) \rightarrow (C \rightarrow D)$, $\neg\big[(C \rightarrow A) \wedge (D \rightarrow E)\big]$, $\neg E$

C. Use truth trees to decide whether the following pairs of statements are equivalent or not.

1. $A \rightarrow (B \rightarrow C)$, $(B \wedge A) \rightarrow C$
2. $A \rightarrow (B \rightarrow A)$, $B \vee \neg(B \rightarrow A)$
3. $\neg A \vee \neg(B \wedge C)$, $\neg(A \vee B) \wedge (C \vee A)$
4. $A \vee \neg(\neg B \vee \neg C)$, $(A \vee B) \wedge (A \vee C)$
5. $\neg(A \vee \neg A)$, $(A \vee B) \wedge (\neg B \wedge \neg A)$

D. Use truth trees to determine whether each argument is valid or invalid.

1. $A \vee B$, $B \vee C$, $\neg A \therefore B \wedge C$

2. $\neg A \vee \neg(\neg B \wedge C)$, $\neg(A \rightarrow D)$, $D \leftrightarrow B \therefore \neg C \vee \neg B$
3. $A \leftrightarrow (B \rightarrow D)$, $D \rightarrow (B \vee C) \therefore B \rightarrow (A \rightarrow D)$
4. $A \rightarrow \big[B \wedge (C \vee D)\big]$, $C \rightarrow \neg\neg D \therefore \neg D \rightarrow \neg C$
5. $(\neg B \vee \neg C) \rightarrow A$, $C \rightarrow \neg D \therefore A \rightarrow (\neg B \vee \neg D)$

Chapter 5

First-order logic

Building blocks of FOL 17

17.1 Splitting the Atom

Consider the following argument, which is obviously valid in English:

> Willard is a logician. All logicians wear funny hats. So Willard wears a funny hat.

To symbolise it in TFL, we might offer a symbolisation key:

L : Willard is a logician.
A : All logicians wear funny hats.
F : Willard wears a funny hat.

And the argument itself becomes:

$$L, A \therefore F$$

This is *invalid* in TFL. But the original English argument is clearly valid.

The problem is not that we have made a mistake while symbolising the argument. This is the best symbolisation we can give *in TFL*. The problem lies with TFL itself. 'All logicians wear funny hats' is about both logicians and hat-wearing. By not retaining this structure in our symbolisation, we lose the connection between Willard's being a logician and Willard's wearing a hat.

The basic units of TFL are atomic sentences, and TFL cannot decompose these. To symbolise arguments like the preceding one, we will have to develop a new logical language which will allow us to *split the atom*. We will call this language *first-order logic*, or *FOL*.

The details of FOL will be explained throughout this chapter, but here is the basic idea for splitting the atom.

First, we have *names*. In FOL, we indicate these with lowercase italic letters. For instance, we might let 'b' stand for Bertie, or let 'i' stand for Willard.

Second, we have predicates. English predicates are expressions like '_____ is a dog' or '_____ is a logician'. These are not complete sentences by themselves. In order to make a complete sentence, we need to fill in the gap. We need to say something like 'Bertie is a dog' or 'Willard is a logician'. In FOL, we indicate predicates with uppercase italic letters. For instance, we might let the FOL predicate 'D' symbolise the English predicate '_____ is a dog'. Then the expression 'Db' will be a sentence in FOL, which symbolises the English sentence 'Bertie is a dog'. Equally, we might let the FOL predicate 'L'

symbolise the English predicate '_____ is a logician'. Then the expression 'Li' will symbolise the English sentence 'Willard is a logician'.

Third, we have quantifiers. For instance, '$\exists$' will roughly convey 'There is at least one ...'. So we might symbolise the English sentence 'there is a dog' with the FOL sentence '$\exists x Dx$', which we would naturally read out-loud as 'there is at least one thing, x, such that x is a dog'.

That is the general idea. But FOL is significantly more subtle than TFL. So we will come at it slowly.

17.2 Names

In English, a *singular term* is a word or phrase that refers to a *specific* person, place, or thing. The word 'dog' is not a singular term, because there are a great many dogs. The phrase 'Bertie' is a singular term, because it refers to a specific terrier. Likewise, the phrase 'Philip's dog Bertie' is a singular term, because it refers to a specific little terrier.

Proper names are a particularly important kind of singular term. These are expressions that pick out individuals without describing them. The name 'Emerson' is a proper name, and the name alone does not tell you anything about Emerson. Of course, some names are traditionally given to boys and others are traditionally given to girls. If 'Hilary' is used as a singular term, you might guess that it refers to a woman. You might, though, be guessing wrongly. Indeed, the name does not necessarily mean that the person referred to is even a person: Hilary might be a giraffe, for all you could tell just from the name.

In FOL, our NAMES are lower-case letters 'a' through to 'r'. We can add subscripts if we want to use some letter more than once. So here are some singular terms in FOL:

$$a, b, c, \ldots, r, a_1, f_{32}, j_{390}, m_{12}$$

These should be thought of along the lines of proper names in English. But with one difference. 'Tim Button' is a proper name, but there are several people with this name. (Equally, there are at least two people with the name 'P.D. Magnus'.) We live with this kind of ambiguity in English, allowing context to individuate the fact that 'Tim Button' refers to the lecturer of this course, and not some other guy. In FOL, we do not tolerate any such ambiguity. Each name must pick out *exactly* one thing. (However, two different names may pick out the same thing.)

As with TFL, we can provide symbolisation keys. These indicate, temporarily, what a name shall pick out. So we might offer:

e : Elsa
g : Gregor
m : Marybeth

17.3 Predicates

The simplest predicates are properties of individuals. They are things you can say about an object. Here are some examples of English predicates:

_____ is a dog
_____ is a member of Monty Python
A piano fell on _____

In general, you can think about predicates as things which combine with singular terms to make sentences. Conversely, you can start with sentences and make predicates out of them by removing terms. Consider the sentence, 'Vinnie borrowed the family car from Nunzio.' By removing a singular term, we can obtain any of three different predicates:

_____ borrowed the family car from Nunzio
Vinnie borrowed _____ from Nunzio
Vinnie borrowed the family car from _____

FOL predicates are capital letters A through Z, with or without subscripts. We might write a symbolisation key for predicates thus:

Ax : _____$_x$ is angry
Hx : _____$_x$ is happy

(Why the subscripts on the gaps? We shall return to this in §19.)

If we combine our two symbolisation keys, we can start to symbolise some English sentences that use these names and predicates in combination. For example, consider the English sentences:

1. Elsa is angry.
2. Gregor and Marybeth are angry.
3. If Elsa is angry, then so are Gregor and Marybeth.

Sentence 1 is straightforward: we symbolise it by 'Ae'.

Sentence 2: this is a conjunction of two simpler sentences. The simple sentences can be symbolised just by 'Ag' and 'Am'. Then we help ourselves to our resources from TFL, and symbolise the entire sentence by '$Ag \wedge Am$'. This illustrates an important point: FOL has all of the truth-functional connectives of TFL.

Sentence 3: this is a conditional, whose antecedent is sentence 1 and whose consequent is sentence 2. So we can symbolise this with '$Ae \rightarrow (Ag \wedge Am)$'.

17.4 Quantifiers

We are now ready to introduce quantifiers. Consider these sentences:

4. Everyone is happy.
5. Someone is angry.

It might be tempting to symbolise sentence 4 as '$He \wedge Hg \wedge Hm$'. Yet this would only say that Elsa, Gregor, and Marybeth are happy. We want to say that *everyone* is happy, even those with no names. In order to do this, we introduce the '$\forall$' symbol. This is called the UNIVERSAL QUANTIFIER.

A quantifier must always be followed by a variable. In FOL, variables are italic lowercase letters 's' through 'z', with or without subscripts. So we might symbolise sentence 4 as '$\forall xHx$'. The variable 'x' is serving as a kind of placeholder. The expression '$\forall x$' intuitively means that you can pick anyone

and put them in as 'x'. The subsequent 'Hx' indicates, of that thing you picked out, that it is happy.

I should say that there is no special reason to use 'x' rather than some other variable. The sentences '$\forall xHx$', '$\forall yHy$', '$\forall zHz$', and '$\forall x_5Hx_5$' use different variables, but they will all be logically equivalent.

To symbolise sentence 5, we introduce another new symbol: the EXISTENTIAL QUANTIFIER, '$\exists$'. Like the universal quantifier, the existential quantifier requires a variable. Sentence 5 can be symbolised by '$\exists xAx$'. Whereas '$\forall xAx$' is read naturally as 'for all x, x is angry', '$\exists xAx$' is read naturally as 'there is something, x, such that x is angry'. Once again, the variable is a kind of placeholder; we could just as easily have symbolised sentence 5 with '$\exists zAz$', '$\exists w_{256}Aw_{256}$', or whatever.

Some more examples will help. Consider these further sentences:

6. No one is angry.
7. There is someone who is not happy.
8. Not everyone is happy.

Sentence 6 can be paraphrased as, 'It is not the case that someone is angry'. We can then symbolise it using negation and an existential quantifier: '$\neg\exists xAx$'. Yet sentence 6 could also be paraphrased as, 'Everyone is not angry'. With this in mind, it can be symbolised using negation and a universal quantifier: '$\forall x\neg Ax$'. Both of these are acceptable symbolisations. Indeed, it will transpire that, in general, $\forall x\neg\phi$ is logically equivalent to $\neg\exists x\phi$. (Notice that I have here returned to the practice of using 'ϕ' as a metavariable, from §7.) Symbolising a sentence one way, rather than the other, might seem more 'natural' in some contexts, but it is not much more than a matter of taste.

Sentence 7 is most naturally paraphrased as, 'There is some x, such that x is not happy'. This then becomes '$\exists x\neg Hx$'. Of course, we could equally have written '$\neg\forall xHx$', which we would naturally read as 'it is not the case that everyone is happy'. And that would be a perfectly adequate symbolisation of sentence 8.

17.5 Domains

Given the symbolisation key we have been using, '$\forall xHx$' symbolises 'Everyone is happy'. Who is included in this *everyone*? When we use sentences like this in English, we usually do not mean everyone now alive on the Earth. We certainly do not mean everyone who was ever alive or who will ever live. We usually mean something more modest: everyone now in the building, everyone enrolled in the ballet class, or whatever.

In order to eliminate this ambiguity, we will need to specify a DOMAIN, which we symbolize using $\mathcal{D}$. The domain is the set of things that we are talking about. So if we want to talk about people in Chicago, we define the domain to be people in Chicago. We write this at the beginning of the symbolisation key, like this:

$$\mathcal{D} = \{x \mid \text{people in Chicago}\}$$

The curly braces, '{}', tell us that the domain is a set and the '$x \mid$' tells us that the set is composed of all those things, x, that are ... well, whatever follows

the '|'. In this case, it tells us that the set consists of all those things that are people in Chicago.

The quantifiers *range over* the domain. Given this domain, '$\forall x$' is to be read roughly as 'Every person in Chicago is such that...' and '$\exists x$' is to be read roughly as 'Some person in Chicago is such that...'.

In FOL, the domain must always include at least one thing. Moreover, in English we can infer 'something is angry' from 'Gregor is angry'. In FOL, then, we shall want to be able to infer '$\exists x Ax$' from 'Ag'. So we shall insist that each name must pick out exactly one thing in the domain. If we want to name people in places beside Chicago, then we need to include those people in the domain.

> A domain, $\mathcal{D}$, must have *at least* one member. A name must pick out *exactly* one member of the domain. But a member of the domain may be picked out by one name, many names, or none at all.

Sentences with one quantifier 18

We now have all of the pieces of FOL. Symbolising more complicated sentences will only be a matter of knowing the right way to combine predicates, names, quantifiers, and connectives. There is a knack to this, and there is no substitute for practice.

18.1 Dealing with syncategorematic adjectives

When we encounter a sentence like

1. Herbie is a white car

We can paraphrase this as 'Herbie is white and Herbie is a car'. We can then use a symbolisation key like:

Wx : _____$_x$ is white
Cx : _____$_x$ is a car
h : Herbie

This allows us to symbolise sentence 1 as '$Wh \wedge Ch$'. But now consider:

2. Damon Stoudamire is a short basketball player.
3. Damon Stoudamire is a man.
4. Damon Stoudamire is a short man.

Following the case of Herbie, we might try to use a symbolisation key like:

Sx : _____$_x$ is short
Bx : _____$_x$ is a basketball player
Mx : _____$_x$ is a man
d : Damon Stoudamire

Then we would symbolise sentence 2 with '$Sd \wedge Bd$', sentence 3 with 'Md' and sentence 4 with '$Sd \wedge Md$'. But that would be a terrible mistake! For this now suggests that sentences 2 and 3 together *entail* sentence 4. But they do not. Standing at 5'10", Damon Stoudamire is one of the shortest professional basketball players of all time, but he is nevertheless an averagely-tall man. The point is that sentence 2 says that Damon is short *qua* basketball player, even though he is of average height *qua* man. So you will need to symbolise '_____ is a short basketball player' and '_____ is a short man' using completely different predicates.

Similar examples abound. All basketball players are people, but some good basketball players are bad people. I might be an incompetent player of chess,

but a competent individual. And so it goes. The moral is: when you see two adjectives in a row, you need to ask yourself carefully whether they can be treated as a conjunction or not.

18.2 Common quantifier phrases

Consider these sentences:

5. Every coin in my pocket is a quarter.
6. Some coin on the table is a dime.
7. Not all the coins on the table are dimes.
8. None of the coins in my pocket are dimes.

In providing a symbolisation key, we need to specify a domain. Since we are talking about coins in my pocket and on the table, the domain must at least contain all of those coins. Since we are not talking about anything besides coins, we let the domain be all coins. Since we are not talking about any specific coins, we do not need to deal with any names. So here is our key:

$\mathcal{D} =$ $\{x \mid \text{coins}\}$
Px : $\_\_\_\_\_x$ is in my pocket
Tx : $\_\_\_\_\_x$ is on the table
Qx : $\_\_\_\_\_x$ is a quarter
Dx : $\_\_\_\_\_x$ is a dime

Sentence 5 is most naturally symbolised using a universal quantifier. The universal quantifier says something about everything in the domain, not just about the coins in my pocket. Sentence 5 can be paraphrased as 'for any coin, *if* that coin is in my pocket *then* it is a quarter'. So we can symbolise it as '$\forall x(Px \to Qx)$'.

Since sentence 5 is about coins that are both in my pocket *and* that are quarters, it might be tempting to translate it using a conjunction. However, the sentence '$\forall x(Px \wedge Qx)$' would symbolise the sentence 'every coin is both a quarter and in my pocket'. This obviously means something very different than sentence 5. And so we see:

> A sentence can be symbolised as $\forall x(\mathcal{F}x \to \mathcal{G}x)$ if it can be paraphrased in English as 'every F is G'.

Sentence 6 is most naturally symbolised using an existential quantifier. It can be paraphrased as 'there is some coin which is both on the table and which is a dime'. So we can symbolise it as '$\exists x(Tx \wedge Dx)$'.

Notice that we needed to use a conditional with the universal quantifier, but we used a conjunction with the existential quantifier. Suppose we had instead written '$\exists x(Tx \to Dx)$'. That would mean that there is some object in the domain of which '$(Tx \to Dx)$' is true. Recall that, in TFL, $\phi \to \psi$ is tautologically equivalent to $\neg\phi \vee \psi$. This equivalence will also hold in FOL. So '$\exists x(Tx \to Dx)$' is true if there is some object in the domain, such that '$(\neg Tx \vee Dx)$' is true of that object. That is, '$\exists x(Tx \to Dx)$' is true if some coin is *either* not on the table *or* is a dime. Of course there is a coin that is not on the table: there are coins lots of other places. So it is *very easy* for

'$\exists x(Tx \to Dx)$' to be true. A conditional will usually be the natural connective to use with a universal quantifier, but a conditional within the scope of an existential quantifier tends to say something very weak indeed. As a general rule of thumb, do not put conditionals in the scope of existential quantifiers unless you are sure that you need one.

> A sentence can be symbolised as $\exists x(\mathcal{F}x \wedge \mathcal{G}x)$ if it can be paraphrased in English as 'some F is G'.

Sentence 7 can be paraphrased as, 'It is not the case that every coin on the table is a dime'. So we can symbolise it by '$\neg\forall x(Tx \to Dx)$'. You might look at sentence 7 and paraphrase it instead as, 'Some coin on the table is not a dime'. You would then symbolise it by '$\exists x(Tx \wedge \neg Dx)$'. Although it is probably not immediately obvious yet, these two sentences are logically equivalent. (This is due to the logical equivalence between $\neg\forall x\phi$ and $\exists x\neg\phi$, mentioned in §17, along with the equivalence between $\neg(\phi \to \psi)$ and $\phi \wedge \neg\psi$.)

Sentence 8 can be paraphrased as, 'It is not the case that there is some dime in my pocket'. This can be symbolised by '$\neg\exists x(Px \wedge Dx)$'. It might also be paraphrased as, 'Everything in my pocket is a non-dime', and then could be symbolised by '$\forall x(Px \to \neg Dx)$'. Again the two symbolisations are logically equivalent. Both are correct symbolisations of sentence 8.

There are four sentences in FOL that corresponding to four important forms of sentences in English. It's always a good idea to begin a symbolization task by determining whether the sentence you're dealing with fits one of these forms.

"All As are Bs"	$\rightsquigarrow$	$\forall x(Ax \to Bx)$
"No As are Bs"	$\rightsquigarrow$	$\forall x(Ax \to \neg Bx)$ or $\neg\exists x(Ax \wedge Bx)$
"Some As are Bs"	$\rightsquigarrow$	$\exists x(Ax \wedge Bx)$
"Some As are not Bs"	$\rightsquigarrow$	$\exists x(Ax \wedge \neg Bx)$

18.3 Empty predicates

In §17, I emphasised that a name must pick out exactly one object in the domain. However, a predicate need not apply to anything in the domain. A predicate that applies to nothing in the domain is called an EMPTY predicate. This is worth exploring.

Suppose we want to symbolise these two sentences:

9. Every monkey knows sign language
10. Some monkey knows sign language

It is possible to write the symbolisation key for these sentences in this way:

$\mathcal{D}$ = $\{x \mid \text{animals}\}$
Mx : $\_\_\_\_\_x$ is a monkey.
Sx : $\_\_\_\_\_x$ knows sign language.

Sentence 9 can now be symbolised by '$\forall x(Mx \to Sx)$'. Sentence 10 can be symbolised as '$\exists x(Mx \wedge Sx)$'.

It is tempting to say that sentence 9 *entails* sentence 10. That is, we might think that it is impossible for it to be the case that every monkey knows sign language, without it's also being the case that some monkey knows sign language. But this would be a mistake. It is possible for the sentence '$\forall x(Mx \to Sx)$' to be true even though the sentence '$\exists x(Mx \wedge Sx)$' is false.

How can this be? The answer comes from considering whether these sentences would be true or false *if there were no monkeys*. If there were no monkeys at all (in the domain), then '$\forall x(Mx \to Sx)$' would be *vacuously* true: take any monkey you like—it knows sign language! But if there were no monkeys at all (in the domain), then '$\exists x(Mx \wedge Sx)$' would be false.

Another example will help to bring this home. Suppose we extend the above symbolisation key, by adding:

Rx : $\_\_\_\_\_x$ is a refrigerator

Now consider the sentence '$\forall x(Rx \to Mx)$'. This symbolises 'every refrigerator is a monkey'. And this sentence is true, given our symbolisation key. This is counterintuitive, since we do not want to say that there are a whole bunch of refrigerator monkeys. It is important to remember, though, that '$\forall x(Rx \to Mx)$' is true iff any member of the domain that is a refrigerator is a monkey. Since the domain is *animals*, there are no refrigerators in the domain. Again, then, the sentence is *vacuously* true.

If you were actually dealing with the sentence 'All refrigerators are monkeys', then you would most likely want to include kitchen appliances in the domain. Then the predicate 'R' would not be empty and the sentence '$\forall x(Rx \to Mx)$' would be false.

> When $\mathcal{F}$ is an empty predicate, a sentence $\forall x(\mathcal{F}x \to \ldots)$ will be vacuously true.

As we will see in the next chapter, the class of objects of which a predicate is true is called the predicate's EXTENSION. Empty predicates are therefore said to have NULL EXTENSIONS.

18.4 Picking a domain

The appropriate symbolisation of an English language sentence in FOL will depend on the symbolisation key. Choosing a key can be difficult. Suppose we want to symbolise the English sentence:

11. Every rose has a thorn.

We might offer this symbolisation key:

Rx : $\_\_\_\_\_x$ is a rose
Tx : $\_\_\_\_\_x$ has a thorn

It is tempting to say that sentence 11 should be symbolised as '$\forall x(Rx \to Tx)$'. But we have not yet chosen a domain. If the domain contains all roses, this would be a good symbolisation. Yet if the domain is merely *things on my kitchen table*, then '$\forall x(Rx \to Tx)$' would only come close to covering the fact that every rose *on my kitchen table* has a thorn. If there are no roses on my

kitchen table, the sentence would be trivially true. This is not what we want. To symbolise sentence 11 adequately, we need to include all the roses in the domain. But now we have two options.

First, we can restrict the domain to include all roses but *only* roses. Then sentence 11 can, if we like, be symbolised with '$\forall xTx$'. This is true iff everything in the domain has a thorn; since the domain is just the roses, this is true iff every rose has a thorn. By restricting the domain, we have been able to symbolise our English sentence with a very short sentence of FOL. So this approach can save us trouble, if every sentence that we want to deal with is about roses.

Second, we can let the domain contain things besides roses: rhododendrons; rats; rifles; whatever. And we will certainly need to include a more expansive domain if we simultaneously want to symbolise sentences like:

12. Every cowboy sings a sad, sad song.

Our domain must now include both all the roses (so that we can symbolise sentence 11) and all the cowboys (so that we can symbolise sentence 12). So we might offer the following symbolisation key:

$\mathcal{D} =$ $\{x \mid$ people and plants$\}$
Cx : $\_\_\_\_\_x$ is a cowboy
Sx : $\_\_\_\_\_x$ sings a sad, sad song
Rx : $\_\_\_\_\_x$ is a rose
Tx : $\_\_\_\_\_x$ has a thorn

Now we will have to symbolise sentence 11 with '$\forall x(Rx \to Tx)$', since '$\forall xTx$' would symbolise the sentence 'every person or plant has a thorn'. Similarly, we will have to symbolise sentence 12 with '$\forall x(Cx \to Sx)$'.

In general, the universal quantifier can be used to symbolise the English expression 'everyone' if the domain only contains people. If there are people and other things in the domain, then 'everyone' must be treated as 'every person'.

18.5 The utility of paraphrase

When symbolising English sentences in FOL, it is important to understand the structure of the sentences you want to symbolise. What matters is the final symbolisation in FOL, and sometimes you will be able to move from an English language sentence directly to a sentence of FOL. Other times, it helps to paraphrase the sentence one or more times. Each successive paraphrase should move from the original sentence closer to something that you can finally symbolise directly in FOL.

For the next several examples, we will use this symbolisation key:

$\mathcal{D} =$ $\{x \mid$ people$\}$
Bx : $\_\_\_\_\_x$ is a bassist.
Rx : $\_\_\_\_\_x$ is a rock star.
k : Kim Deal

Now consider these sentences:

13. If Kim Deal is a bassist, then she is a rock star.
14. If a person is a bassist, then she is a rock star.

The same words appear as the consequent in sentences 13 and 14 ('... she is a rock star'), but they mean very different things. To make this clear, it often helps to paraphrase the original sentences, removing pronouns.

Sentence 13 can be paraphrased as, 'If Kim Deal is a bassist, then *Kim Deal* is a rockstar'. This can obviously be symbolised as '$Bk \to Rk$'.

Sentence 14 must be paraphrased differently: 'If a person is a bassist, then *that person* is a rock star'. This sentence is not about any particular person, so we need a variable. As a halfway house, we can paraphrase this as, 'For any person x, if x is a bassist, then x is a rockstar'. Now this can be symbolised as '$\forall x(Bx \to Rx)$'. This is the same sentence we would have used to symbolise 'Everyone who is a bassist is a rock star'. And on reflection, that is surely true iff sentence 14 is true, as we would hope.

Consider these further sentences:

15. If anyone is a bassist, then Kim Deal is a rock star.
16. If anyone is a bassist, then she is a rock star.

The same words appear as the antecedent in sentences 15 and 16 ('If anyone is a bassist...'). But it can be tricky to work out how to symbolise these two uses. Again, paraphrase will come to our aid.

Sentence 15 can be paraphrased, 'If there is at least one bassist, then Kim Deal is a rock star'. It is now clear that this is a conditional whose antecedent is a quantified expression; so we can symbolise the entire sentence with a conditional as the main logical operator: '$\exists x Bx \to Rk$'.

Sentence 16 can be paraphrased, 'For all people x, if x is a bassist, then x is a rock star'. Or, in more natural English, it can be paraphrased by 'All bassists are rock stars'. It is best symbolised as '$\forall x(Bx \to Rx)$', just like sentence 14.

The moral is that the English words 'any' and 'anyone' should typically be symbolised using quantifiers. And if you are having a hard time determining whether to use an existential or a universal quantifier, try paraphrasing the sentence with an English sentence that uses words *besides* 'any' or 'anyone'.

18.6 Quantifiers and scope

Continuing the example, suppose I want to symbolise these sentences:

17. If everyone is a bassist, then Tim is a bassist
18. Everyone is such that, if she is a bassist, then Tim is a bassist.

To symbolise these sentences, I shall have to add a new name to the symbolisation key, namely:

b : Tim

Sentence 17 is a conditional, whose antecedent is 'everyone is a bassist'. So we will symbolise it with '$\forall x Bx \to Bb$'. This sentence is *necessarily* true: if *everyone* is indeed a bassist, then take any one you like—for example Tim—and he will be a bassist.

Sentence 18, by contrast, might best be paraphrased by 'every person x is such that, if x is a bassist, then Tim is a bassist'. This is symbolised by '$\forall x(Bx \rightarrow Bb)$'. And this sentence is false. Kim Deal is a bassist. So 'Bk' is true. But Tim is not a bassist, so 'Bb' is false. Accordingly, '$Bk \rightarrow Bb$' will be false. So '$\forall x(Bx \rightarrow Bb)$' will be false as well.

In short, '$\forall xBx \rightarrow Bb$' and '$\forall x(Bx \rightarrow Bb)$' are very different sentences. We can explain the difference in terms of the *scope* of the quantifier. The scope of quantification is very much like the scope of negation, which we considered when discussing TFL, and it will help to explain it in this way.

In the sentence '$\neg Bk \rightarrow Bb$', the scope of '$\neg$' is just the antecedent of the conditional. We are saying something like: if 'Bk' is false, then 'Bb' is true. Similarly, in the sentence '$\forall xBx \rightarrow Bb$', the scope of '$\forall x$' is just the antecedent of the conditional. We are saying something like: if 'Bx' is true of *everything*, then 'Bb' is also true.

In the sentence '$\neg(Bk \rightarrow Bb)$', the scope of '$\neg$' is the entire sentence. We are saying something like: '$(Bk \rightarrow Bb)$' is false. Similarly, in the sentence '$\forall x(Bx \rightarrow Bb)$', the scope of '$\forall x$' is the entire sentence. We are saying something like: '$(Bx \rightarrow Bb)$' is true of *everything*.

The moral of the story is simple. When you are using conditionals, be very careful to make sure that you have sorted out the scope correctly.

Practice exercises

A. Here are the syllogistic figures identified by Aristotle and his successors, along with their medieval names:

- **Barbara.** All G are F. All H are G. So: All H are F
- **Celarent.** No G are F. All H are G. So: No H are F
- **Ferio.** No G are F. Some H is G. So: Some H is not F
- **Darii.** All G are F. Some H is G. So: Some H is F.
- **Camestres.** All F are G. No H are G. So: No H are F.
- **Cesare.** No F are G. All H are G. So: No H are F.
- **Baroko.** All F are G. Some H is not G. So: Some H is not F.
- **Festino.** No F are G. Some H are G. So: Some H is not F.
- **Datisi.** All G are F. Some G is H. So: Some H is F.
- **Disamis.** Some G is F. All G are H. So: Some H is F.
- **Ferison.** No G are F. Some G is H. So: Some H is not F.
- **Bokardo.** Some G is not F. All G are H. So: Some H is not F.
- **Camenes.** All F are G. No G are H So: No H is F.
- **Dimaris.** Some F is G. All G are H. So: Some H is F.
- **Fresison.** No F are G. Some G is H. So: Some H is not F.

Symbolise each argument in FOL.

B. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid \text{people}\}$
Kx : $\_\_\_\_\_x$ knows the combination to the safe
Sx : $\_\_\_\_\_x$ is a spy
Vx : $\_\_\_\_\_x$ is a vegetarian

h : Hofthor
i : Ingmar

symbolise the following sentences in FOL:

1. Neither Hofthor nor Ingmar is a vegetarian.
2. No spy knows the combination to the safe.
3. No one knows the combination to the safe unless Ingmar does.
4. Hofthor is a spy, but no vegetarian is a spy.

C. Using this symbolisation key:

$\mathcal{D} =$ $\{x \mid \text{animals}\}$
Ax : $\_\_\_\_\_x$ is an alligator.
Mx : $\_\_\_\_\_x$ is a monkey.
Rx : $\_\_\_\_\_x$ is a reptile.
Zx : $\_\_\_\_\_x$ lives at the zoo.
a : Amos
b : Bouncer
c : Cleo

symbolise each of the following sentences in FOL:

1. Amos, Bouncer, and Cleo all live at the zoo.
2. Bouncer is a reptile, but not an alligator.
3. Some reptile lives at the zoo.
4. Every alligator is a reptile.
5. Any animal that lives at the zoo is either a monkey or an alligator.
6. There are reptiles which are not alligators.
7. If any animal is an reptile, then Amos is.
8. If any animal is an alligator, then it is a reptile.

D. For each argument, write a symbolisation key and symbolise the argument in FOL.

1. Willard is a logician. All logicians wear funny hats. So Willard wears a funny hat
2. Nothing on my desk escapes my attention. There is a computer on my desk. As such, there is a computer that does not escape my attention.
3. All my dreams are black and white. Old TV shows are in black and white. Therefore, some of my dreams are old TV shows.
4. Neither Holmes nor Watson has been to Australia. A person could see a kangaroo only if they had been to Australia or to a zoo. Although Watson has not seen a kangaroo, Holmes has. Therefore, Holmes has been to a zoo.
5. No one expects the Spanish Inquisition. No one knows the troubles I've seen. Therefore, anyone who expects the Spanish Inquisition knows the troubles I've seen.
6. All babies are illogical. Nobody who is illogical can manage a crocodile. Berthold is a baby. Therefore, Berthold is unable to manage a crocodile.

Multiple generality 19

So far, we have only considered sentences that require one-place predicates and one quantifier. The full power of FOL really comes out when we start to use many-place predicates and multiple quantifiers. For this insight, we largely have Gottlob Frege (1879) to thank, but also Peirce.

19.1 Many-placed predicates

All of the predicates that we have considered so far concern properties that objects might have. The predicates have one gap in them, and to make a sentence, we simply need to slot in one name. They are ONE-PLACE predicates.

But other predicates concern the *relation* between two things. Here are some examples of relational predicates in English:

_____ loves _____
_____ is to the left of _____
_____ is in debt to _____

These are TWO-PLACE predicates. They need to be filled in with two terms in order to make a sentence. Conversely, if we start with an English sentence containing many singular terms, we can remove two singular terms, to obtain different two-place predicates. Consider the sentence 'Vinnie borrowed the family car from Nunzio'. By deleting two singular terms, we can obtain any of three different two-place predicates

Vinnie borrowed _____ from _____
_____ borrowed the family car from _____
_____ borrowed _____ from Nunzio

And by removing all three singular terms, we obtain a THREE-PLACE predicate:

_____ borrowed _____ from _____

Indeed, there is no in principle upper limit on the number of places that our predicates may contain.

Now there is a little foible with the above. I have used the same symbol, '_____', to indicate a gap formed by deleting a term from a sentence. However (as Frege emphasised), these are *different* gaps. To obtain a sentence, we can fill them in with the same term, but we can equally fill them in with different terms, and in various different orders. The following are all perfectly good sentences, and they all mean very different things:

Karl loves Karl
Karl loves Imre
Imre loves Karl
Imre loves Imre

The point is that we need to keep track of the gaps in predicates, so that we can keep track of how we are filling them in.

To keep track of the gaps, I shall label them. The labelling conventions I shall adopt are best explained by example. Suppose I want to symbolise the following sentences:

1. Karl loves Imre.
2. Imre loves himself.
3. Karl loves Imre, but not vice versa.
4. Karl is loved by Imre.

I will start with the following representation key:

$\mathcal{D} =$ $\{x \mid \text{people}\}$
$i:$ Imre
$k:$ Karl
$Lxy:$ _____$_x$ loves _____$_y$

Sentence 1 will now be symbolised by 'Lki'.

Sentence 2 can be paraphrased as 'Imre loves Imre'. It can now be symbolised by 'Lii'.

Sentence 3 is a conjunction. We might paraphrase it as 'Karl loves Imre, and Imre does not love Karl'. It can now be symbolised by '$Lki \wedge \neg Lik$'.

Sentence 4 might be paraphrased by 'Imre loves Karl'. It can then be symbolised by 'Lik'. Of course, this slurs over the difference in tone between the active and passive voice; such nuances are lost in FOL.

This last example, though, highlights something important. Suppose we add to our symbolisation key the following:

$Mxy:$ _____$_y$ loves _____$_x$

Here, we have used the same English word ('loves') as we used in our symbolisation key for 'Lxy'. However, we have swapped the order of the *gaps* around (just look closely at those little subscripts!) So 'Mki' and 'Lik' now *both* symbolise 'Imre loves Karl'. 'Mik' and 'Lki' now *both* symbolise 'Karl loves Imre'. Since love can be unrequited, these are very different claims.

The moral is simple. When we are dealing with predicates with more than one place, we need to pay careful attention to the order of the places.

19.2 The order of quantifiers

Consider the sentence 'everyone loves someone'. This is potentially ambiguous. It might mean either of the following:

5. For every person x, there is some person that x loves
6. There is some particular person whom every person loves

Sentence 5 can be symbolised by '$\forall x \exists y Lxy$', and would be true of a love-triangle. For example, suppose that our domain of discourse is restricted to Imre, Juan and Karl. Suppose also that Karl loves Imre but not Juan, that Imre loves Juan but not Karl, and that Juan loves Karl but not Imre. Then sentence 5 is true.

Sentence 6 is symbolised by '$\exists y \forall x Lxy$'. Sentence 6 is *not* true in the situation just described. Again, suppose that our domain of discourse is restricted to Imre, Juan and Karl. Then this requires that all of Juan, Imre and Karl converge on (at least) one object of love.

The point of the example is to illustrate that the order of the quantifiers matters a great deal. Indeed, to switch them around is called a *quantifier shift fallacy*. Here is an example, which comes up in various forms throughout the philosophical literature:

	For every person, there is some truth they cannot know.	$(\forall\exists)$
So:	There is some truth, that no person can know.	$(\exists\forall)$

This argument form is obviously invalid. It's just as bad as:[1]

	Every dog has its day.	$(\forall\exists)$
So:	There is a day for all the dogs.	$(\exists\forall)$

The moral is: take great care with the order of quantification.

19.3 Stepping-stones to symbolisation

Once we have the possibility of multiple quantifiers and many-place predicates, representation in FOL can quickly start to become a bit tricky. When you are trying to symbolise a complex sentence, I recommend laying down several stepping stones. As usual, this idea is best illustrated by example. Consider this representation key:

$\mathcal{D} =$ $\{x \mid$ people and dogs$\}$
Dx : $\_\_\_\_\_x$ is a dog
Fxy : $\_\_\_\_\_x$ is a friend of $\_\_\_\_\_y$
Oxy : $\_\_\_\_\_x$ owns $\_\_\_\_\_y$
g : Geraldo

And now let's try to symbolise these sentences:

7. Geraldo is a dog owner.
8. Someone is a dog owner.
9. All of Geraldo's friends are dog owners.
10. Every dog owner is the friend of a dog owner.
11. Every dog owner's friend owns a dog of a friend.

Sentence 7 can be paraphrased as, 'There is a dog that Geraldo owns'. This can be symbolised by '$\exists x(Dx \wedge Ogx)$'.

Sentence 8 can be paraphrased as, 'There is some y such that y is a dog owner'. Dealing with part of this, we might write '$\exists y(y$ is a dog owner)'. Now

[1] Thanks to Rob Trueman for the example.

the fragment we have left as 'y is a dog owner' is much like sentence 7, except that it is not specifically about Geraldo. So we can symbolise sentence 8 by:

$$\exists y \exists x (Dx \wedge Oyx)$$

I need to pause to clarify something here. In working out how to symbolise the last sentence, we wrote down '$\exists y$(y is a dog owner)'. To be very clear: this is *neither* an FOL sentence *nor* an English sentence: it uses bits of FOL ('$\exists$', 'y') and bits of English ('dog owner'). It is really *just a stepping-stone* on the way to symbolising the entire English sentence with a FOL sentence. You should regard it as a bit of rough-working-out, on a par with the doodles that you might absent-mindedly draw in the margin of this book, whilst you are concentrating fiercely on some problem.

Sentence 9 can be paraphrased as, 'Everyone who is a friend of Geraldo is a dog owner'. Using our stepping-stone tactic, we might write

$$\forall x \big[Fxg \rightarrow x \text{ is a dog owner} \big]$$

Now the fragment that we have left to deal with, 'x is a dog owner', is structurally just like sentence 7. But it would be a mistake for us simply to write

$$\forall x \big[Fxg \rightarrow \exists x (Dx \wedge Oxx) \big]$$

for we would here have a *clash of variables*. The scope of the universal quantifier, '$\forall x$', is the entire conditional, so the 'x' in 'Dx' should be governed by that. But 'Dx' also falls under the scope of the existential quantifier '$\exists x$', so the 'x' in 'Dx' should be governed by that. And now confusion reigns: which 'x' are we talking about? Suddenly the sentence would be ambiguous (if it is even meaningful at all), and logicians hate ambiguity. The broad moral is that a single variable cannot serve two masters simultaneously.

To continue our symbolisation, then, we must choose some different variable for our existential quantifier. What we want is something like:

$$\forall x \big[Fxg \rightarrow \exists z (Dz \wedge Oxz) \big]$$

and this adequately symbolises sentence 9.

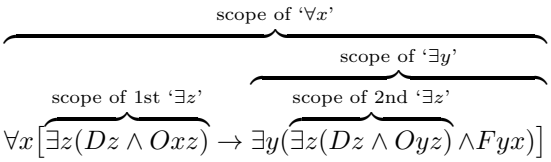
Sentence 10 can be paraphrased as 'For any x that is a dog owner, there is a dog owner who is a friend of x'. Using our stepping-stone tactic, this becomes

$$\forall x \big[x \text{ is a dog owner} \rightarrow \exists y (y \text{ is a dog owner} \wedge Fyx) \big]$$

Completing the symbolisation, we end up with

$$\forall x \big[\exists z (Dz \wedge Oxz) \rightarrow \exists y \big(\exists z (Dz \wedge Oyz) \wedge Fyx \big) \big]$$

Note that we have used the same letter, 'z', in both the antecedent and the consequent of the conditional, but that these are governed by two different quantifiers. This is ok: there is no clash here, because is clear which quantifier the letter falls under. We might graphically represent the scope of the quantifiers thus:



This shows that no variable is being forced to serve two masters simultaneously.

Sentence 11 is the trickiest yet. First we paraphrase it as 'For any x that is a friend of a dog owner, x owns a dog which is also owned by a friend of x'. Using our stepping-stone tactic, this becomes:

$$\forall x\big[x \text{ is a friend of a dog owner} \rightarrow x \text{ owns a dog which is owned by a friend of } x\big]$$

Breaking this down a bit more:

$$\forall x\big[\exists y(Fxy \wedge y \text{ is a dog owner}) \rightarrow \exists y(Dy \wedge Oxy \wedge y \text{ is owned by a friend of } x)\big]$$

And a bit more:

$$\forall x\big[\exists y(Fxy \wedge \exists z(Dz \wedge Oyz)) \rightarrow \exists y(Dy \wedge Oxy \wedge \exists z(Fzx \wedge Ozy))\big]$$

And we are done!

Practice exercises

A. Using this symbolisation key:

$\mathcal{D} =$ $\{x \mid \text{animals}\}$
Ax : $\_\_\_\_\_x$ is an alligator
Mx : $\_\_\_\_\_x$ is a monkey
Rx : $\_\_\_\_\_x$ is a reptile
Zx : $\_\_\_\_\_x$ lives at the zoo
Lxy : $\_\_\_\_\_x$ loves $\_\_\_\_\_y$
a : Amos
b : Bouncer
c : Cleo

symbolise each of the following sentences in FOL:

1. If Cleo loves Bouncer, then Bouncer is a monkey.
2. If both Bouncer and Cleo are alligators, then Amos loves them both.
3. Cleo loves a reptile.
4. Bouncer loves all the monkeys that live at the zoo.
5. All the monkeys that Amos loves love him back.
6. Every monkey that Cleo loves is also loved by Amos.
7. There is a monkey that loves Bouncer, but sadly Bouncer does not reciprocate this love.

B. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid \text{animals}\}$
Dx : $\_\_\_\_\_x$ is a dog
Sx : $\_\_\_\_\_x$ likes samurai movies
Lxy : $\_\_\_\_\_x$ is larger than $\_\_\_\_\_y$
b : Bertie

e : Emerson
f : Fergis

symbolise the following sentences in FOL:

1. Bertie is a dog who likes samurai movies.
2. Bertie, Emerson, and Fergis are all dogs.
3. Emerson is larger than Bertie, and Fergis is larger than Emerson.
4. All dogs like samurai movies.
5. Only dogs like samurai movies.
6. There is a dog that is larger than Emerson.
7. If there is a dog larger than Fergis, then there is a dog larger than Emerson.
8. No animal that likes samurai movies is larger than Emerson.
9. No dog is larger than Fergis.
10. Any animal that dislikes samurai movies is larger than Bertie.
11. There is an animal that is between Bertie and Emerson in size.
12. There is no dog that is between Bertie and Emerson in size.
13. No dog is larger than itself.
14. Every dog is larger than some dog.
15. There is an animal that is smaller than every dog.
16. If there is an animal that is larger than any dog, then that animal does not like samurai movies.

C. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid$ people and dishes at a potluck$\}$
Rx : $\_\_\_\_\_x$ has run out.
Tx : $\_\_\_\_\_x$ is on the table.
Fx : $\_\_\_\_\_x$ is food.
Px : $\_\_\_\_\_x$ is a person.
Lxy : $\_\_\_\_\_x$ likes $\_\_\_\_\_y$.
e : Eli
f : Francesca
g : the guacamole

symbolise the following English sentences in FOL:

1. All the food is on the table.
2. If the guacamole has not run out, then it is on the table.
3. Everyone likes the guacamole.
4. If anyone likes the guacamole, then Eli does.
5. Francesca only likes the dishes that have run out.
6. Francesca likes no one, and no one likes Francesca.
7. Eli likes anyone who likes the guacamole.
8. Eli likes anyone who likes the people that he likes.
9. If there is a person on the table already, then all of the food must have run out.

D. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid$ people$\}$
Dx : $\_\_\_\_\_x$ dances ballet.

Fx : $\_\_\_\_\_x$ is female.
Mx : $\_\_\_\_\_x$ is male.
Cxy : $\_\_\_\_\_x$ is a child of $\_\_\_\_\_y$.
Sxy : $\_\_\_\_\_x$ is a sibling of $\_\_\_\_\_y$.
e : Elmer
j : Jane
p : Patrick

symbolise the following arguments in FOL:

1. All of Patrick's children are ballet dancers.
2. Jane is Patrick's daughter.
3. Patrick has a daughter.
4. Jane is an only child.
5. All of Patrick's sons dance ballet.
6. Patrick has no sons.
7. Jane is Elmer's niece.
8. Patrick is Elmer's brother.
9. Patrick's brothers have no children.
10. Jane is an aunt.
11. Everyone who dances ballet has a brother who also dances ballet.
12. Every woman who dances ballet is the child of someone who dances ballet.

Identity 20

Consider this sentence:

1. Pavel owes money to everyone

Let the domain be people; this will allow us to translate 'everyone' as a universal quantifier. Offering the symbolisation key:

Oxy : $\_\_\_\_\_x$ owes money to $\_\_\_\_\_y$
p : Pavel

we can symbolise sentence 1 by '$\forall xOpx$'. But this has a (perhaps) odd consequence. It requires that Pavel owes money to every member of the domain (whatever the domain may be). The domain certainly includes Pavel. So this entails that Pavel owes money to himself.

Perhaps we meant to say:

2. Pavel owes money to everyone *else*
3. Pavel owes money to everyone *other than* Pavel
4. Pavel owes money to everyone *except* Pavel himself

But we do not know how to deal with the italicised words yet. The solution is to add another symbol to FOL.

20.1 Adding identity

The symbol '=' is a two-place predicate. Since it is to have a special meaning, we shall write it a bit differently: we put it between two terms, rather than out front. And it *does* have a very particular meaning. We *always* adopt the following symbolisation key:

$x = y$: $\_\_\_\_\_x$ is identical to $\_\_\_\_\_y$

This does not mean *merely* that the objects in question are indistinguishable, or that all of the same things are true of them. Rather, it means that the objects in question are *the very same* object.

Now suppose we want to symbolise this sentence:

5. Pavel is Mister Checkov.

Let us add to our symbolisation key:

c : Mister Checkov

Now sentence 5 can be symbolised as '$p = c$'. This means that the names 'p' and 'c' both name the same thing.

We can also now deal with sentences 2–4. All of these sentences can be paraphrased as 'Everyone who is not Pavel is owed money by Pavel'. Paraphrasing some more, we get: 'For all x, if x is not Pavel, then x is owed money by Pavel'. Now that we are armed with our new identity symbol, we can symbolise this as '$\forall x(\neg x = p \rightarrow Opx)$'.

This last sentence contains the formula '$\neg x = p$'. And that might look a bit strange, because the symbol that comes immediately after the '$\neg$' is a variable, rather than a predicate. But this is no problem. We are simply negating the entire formula, '$x = p$'.

In addition to sentences that use the word 'else', 'other than' and 'except', identity will be helpful when symbolising some sentences that contain the words 'besides' and 'only.' Consider these examples:

6. No one besides Pavel owes money to Hikaru.
7. Only Pavel owes Hikaru money.

Letting 'h' name Hikaru, sentence 6 can be paraphrased as, 'No one who is not Pavel owes money to Hikaru'. This can be symbolised by '$\neg\exists x(\neg x = p \wedge Oxh)$'. Equally, sentence 6 can be paraphrased as 'for all x, if x owes money to Hikaru, then x is Pavel'. Then it can be symbolised as '$\forall x(Oxh \rightarrow x = p)$'.

Sentence 7 can be treated similarly. But there is one subtlety here. Do either sentence 6 or 7 entail that Pavel himself owes money to Hikaru?

20.2 There are at least...

We can also use identity to say how many things there are of a particular kind. For example, consider these sentences:

8. There is at least one apple
9. There are at least two apples
10. There are at least three apples

We shall use the symbolisation key:

Ax : $\_\_\_\_\_x$ is an apple

Sentence 8 does not require identity. It can be adequately symbolised by '$\exists x Ax$': There is some apple; perhaps many, but at least one.

It might be tempting to also translate sentence 9 without identity. Yet consider the sentence '$\exists x \exists y(Ax \wedge Ay)$'. Roughly, this says that there is some apple x in the domain and some apple y in the domain. Since nothing precludes these from being one and the same apple, this would be true even if there were only one apple. In order to make sure that we are dealing with *different* apples, we need an identity predicate. Sentence 9 needs to say that the two apples that exist are not identical, so it can be symbolised by '$\exists x \exists y(Ax \wedge Ay \wedge \neg x = y)$'.

Sentence 10 requires talking about three different apples. Now we need three existential quantifiers, and we need to make sure that each will pick out something different: '$\exists x \exists y \exists z(Ax \wedge Ay \wedge Az \wedge \neg x = y \wedge \neg y = z \wedge \neg x = z)$'.

20.3 There are at most...

Now consider these sentences:

11. There is at most one apple
12. There are at most two apples

Sentence 11 can be paraphrased as, 'It is not the case that there are at least *two* apples'. This is just the negation of sentence 9:

$$\neg\exists x\exists y(Ax \wedge Ay \wedge \neg x = y)$$

But sentence 11 can also be approached in another way. It means that if you pick out an object and it's an apple, and then you pick out an object and it's also an apple, you must have picked out the same object both times. With this in mind, it can be symbolised by

$$\forall x\forall y\big[(Ax \wedge Ay) \rightarrow x = y\big]$$

The two sentences will turn out to be logically equivalent.

In a similar way, sentence 12 can be approached in two equivalent ways. It can be paraphrased as, 'It is not the case that there are *three* or more distinct apples', so we can offer:

$$\neg\exists x\exists y\exists z(Ax \wedge Ay \wedge Az \wedge \neg x = y \wedge \neg y = z \wedge \neg x = z)$$

Or, we can read it as saying that if you pick out an apple, and an apple, and an apple, then you will have picked out (at least) one of these objects more than once. Thus:

$$\forall x\forall y\forall z\big[(Ax \wedge Ay \wedge Az) \rightarrow (x = y \vee x = z \vee y = z)\big]$$

20.4 There are exactly...

We can now consider precise statements, like:

13. There is exactly one apple.
14. There are exactly two apples.

Sentence 13 can be paraphrased as, 'There is *at least* one apple and there is *at most* one apple'. This is just the conjunction of sentence 8 and sentence 11. So we can offer:

$$\exists x Ax \wedge \forall x\forall y\big[(Ax \wedge Ay) \rightarrow x = y\big]$$

But it is perhaps more straightforward to paraphrase sentence 13 as, 'There is a thing x which is an apple, and everything which is an apple is just x itself'. Thought of in this way, we offer:

$$\exists x\big[Ax \wedge \forall y(Ay \rightarrow x = y)\big]$$

Similarly, sentence 14 may be paraphrased as, 'There are *at least* two apples, and there are *at most* two apples'. Thus we could offer

$$\exists x\exists y(Ax \wedge Ay \wedge \neg x = y) \wedge \forall x\forall y\forall z\big[(Ax \wedge Ay \wedge Az) \rightarrow (x = y \vee x = z \vee y = z)\big]$$

More efficiently, though, we can paraphrase it as 'There are at least two different apples, and every apple is one of those two apples'. Then we offer:

$$\exists x \exists y \big[Ax \wedge Ay \wedge \neg x = y \wedge \forall z (Az \rightarrow (x = z \vee y = z)) \big]$$

Finally, consider these sentence:

15. There are exactly two things
16. There are exactly two objects

It might be tempting to add a predicate to our symbolisation key, to symbolise the English predicate '_____ is a thing' or '_____ is an object'. But this is unnecessary. Words like 'thing' and 'object' do not sort wheat from chaff: they apply trivially to everything, which is to say, they apply trivially to every thing. So we can symbolise either sentence with either of the following:

$$\exists x \exists y \neg x = y \wedge \neg \exists x \exists y \exists z (\neg x = y \wedge \neg y = z \wedge \neg x = z)$$
$$\exists x \exists y \big[\neg x = y \wedge \forall z (x = z \vee y = z) \big]$$

Practice exercises

A. Explain why:

- '$\exists x \forall y (Ay \leftrightarrow x = y)$' is a good symbolisation of 'there is exactly one apple'.
- '$\exists x \exists y \big[\neg x = y \wedge \forall z (Az \leftrightarrow (x = z \vee y = z)) \big]$' is a good symbolisation of 'there are exactly two apples'.

Definite descriptions 21

Consider sentences like:

1. Nick is the traitor.
2. The traitor went to Cambridge.
3. The traitor is the deputy

These are definite descriptions: they are meant to pick out a *unique* object. They should be contrasted with *indefinite* descriptions, such as 'Nick is *a* traitor'. They should equally be contrasted with *generics*, such as '*The* whale is a mammal' (it's inappropriate to ask *which* whale). The question we face is: how should we deal with definite descriptions in FOL?

21.1 Treating definite descriptions as terms

One option would be to introduce new names whenever we come across a definite description. This is probably not a great idea. We know that *the* traitor—whoever it is—is indeed *a* traitor. We want to preserve that information in our symbolisation.

A second option would be to use a *new* definite description operator, such as '$\imath$'. The idea would be to symbolise 'the F' as '$\imath xFx$'; or to symbolise 'the G' as '$\imath xGx$', etc. Expression of the form $\imath\chi\phi\chi$ would then behave like names. If we followed this path, then using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid \text{people}\}$
Tx : $\_\_\_\_\_x$ is a traitor
Dx : $\_\_\_\_\_x$ is a deputy
Cx : $\_\_\_\_\_x$ went to Cambridge
n : Nick

We could symbolise sentence 1 with '$\imath xTx = n$', sentence 2 with '$C\imath xTx$', and sentence 3 with '$\imath xTx = \imath xDx$'.

However, it would be nice if we didn't have to add a new symbol to FOL. And indeed, we might be able to make do without one.

21.2 Russell's analysis

Bertrand Russell offered an analysis of definite descriptions. Very briefly put, he observed that, when we say 'the F' in the context of a definite description,

our aim is to pick out the *one and only* thing that is F (in the appropriate context). Thus Russell analysed the notion of a definite description as follows:[1]

> the F is G **iff** there is at least one F, *and*
> there is at most one F, *and*
> every F is G

Note a very important feature of this analysis: *'the' does not appear on the right-side of the equivalence.* Russell is aiming to provide an understanding of definite descriptions, in terms that do not presuppose them.

Now, one might worry that I can say 'the table is brown' without implying that there is one and only one table in the universe. But this is not (yet) a fantastic counterexample to Russell's analysis. The domain of discourse is likely to be restricted by context (e.g. to objects in my line of sight).

If we accept Russell's analysis of definite descriptions, then we can symbolise sentences of the form 'the F is G' using our strategy for numerical quantification in FOL. After all, we can deal with the three conjuncts on the right-hand side of Russell's analysis as follows:

$$\exists x Fx \wedge \forall x \forall y((Fx \wedge Fy) \rightarrow x = y) \wedge \forall x(Fx \rightarrow Gx)$$

In fact, we could express the same point rather more crisply, by recognising that the first two conjuncts just amount to the claim that there is *exactly* one F, and that the last conjunct tells us that that object is F. So, equivalently, we could offer:

$$\exists x\big[Fx \wedge \forall y(Fy \rightarrow x = y) \wedge Gx\big]$$

Using these sorts of techniques, we can now symbolise sentences 1–3 without using any new-fangled fancy operator, such as 'ι'.

Sentence 1 is exactly like the examples we have just considered. So we would symbolise it by '$\exists x(Tx \wedge \forall y(Ty \rightarrow x = y) \wedge x = n)$'.

Sentence 2 poses no problems either: '$\exists x(Tx \wedge \forall y(Ty \rightarrow x = y) \wedge Cx)$'.

Sentence 3 is a little trickier, because it links two definite descriptions. But, deploying Russell's analysis, it can be paraphrased by 'there is exactly one traitor, x, and there is exactly one deputy, y, and x = y'. So we can symbolise it by:

$$\exists x \exists y\big(\big[Tx \wedge \forall z(Tz \rightarrow x = z)\big] \wedge \big[Dy \wedge \forall z(Dz \rightarrow y = z)\big] \wedge x = y\big)$$

Note that I have made sure that the formula '$x = y$' falls within the scope of both quantifiers!

21.3 Empty definite descriptions

One of the nice features of Russell's analysis is that it allows us to handle *empty* definite descriptions neatly.

France has no king at present. Now, if we were to introduce a name, 'k', to name the present King of France, then everything would go wrong: remember

[1] Bertrand Russell, 'On Denoting', 1905, *Mind 14*, pp. 479–93; also Russell, *Introduction to Mathematical Philosophy*, 1919, London: Allen and Unwin, ch. 16.

from §17 that a name must always pick out some object in the domain, and whatever we choose as our domain, it will contain no present kings of France.

Russell's analysis neatly avoids this problem. Russell tells us to treat definite descriptions using predicates and quantifiers, instead of names. Since predicates can be empty (see §18), this means that no difficulty now arises when the definite description is empty.

Indeed, Russell's analysis helpfully highlights two ways to go wrong in a claim involving a definite description. To adapt an example from Stephen Neale (1990),[2] suppose I, Tim Button, claim:

4. I am dating the present king of France.

Using the following symbolisation key:

b : Tim
Kx : $\_\_\_\_\__x$ is a present king of France
Dxy : $\_\_\_\_\__x$ is dating $\_\_\_\_\__y$

Sentence 4 would be symbolised by '$\exists x(\forall y(Ky \leftrightarrow x = y) \wedge Dbx)$'. Now, this can be false in (at least) two ways, corresponding to these two different sentences:

5. There is no one who is both the present King of France and such that he and Tim are dating.
6. There is a unique present King of France, but Tim is not dating him.

Sentence 5 might be paraphrased by 'It is not the case that: the present King of France and Tim are dating'. It will then be symbolised by '$\neg\exists x\big[Kx \wedge \forall y(Ky \rightarrow x = y) \wedge Dbx\big]$'. We might call this *outer* negation, since the negation governs the entire sentence. Note that it will be true if there is no present King of France.

Sentence 6 can be symbolised by '$\exists x(Kx \wedge \forall y(Ky \rightarrow x = y) \wedge \neg Dbx)$. We might call this *inner* negation, since the negation occurs within the scope of the definite description. Note that its truth requires that there is a present King of France, albeit one who is not dating Tim.

21.4 The adequacy of Russell's analysis

How good is Russell's analysis of definite descriptions? This question has generated a substantial philosophical literature, but I shall content myself with two observations.

One worry focusses on Russell's treatment of empty definite descriptions. If there are no Fs, then on Russell's analysis, both 'the F is G' is and 'the F is non-G' are false. P.F. Strawson suggested that such sentences should not be regarded as false, exactly.[3] Rather, they involve presupposition failure, and need to be regarded as *neither* true *nor* false.

If we agree with Strawson here, we shall need to revise our logic. For, in our logic, there are only two truth values (True and False), and every sentence is assigned exactly one of these truth values.

[2] Neale, *Descriptions*, 1990, Cambridge: MIT Press.
[3] P.F. Strawson, 'On Referring', 1950, *Mind 59*, pp. 320–34.

But there is room to disagree with Strawson. Strawson is appealing to some linguistic intuitions, but it is not clear that they are very robust. For example: isn't it just *false*, not 'gappy', that Tim is dating the present King of France?[4]

Keith Donnellan raised a second sort of worry, which (very roughly) can be brought out by thinking about a case of mistaken identity.[5] Two men stand in the corner: a very tall man drinking what looks like a gin martini; and a very short man drinking what looks like a pint of water. Seeing them, Malika says:

7. The gin-drinker is very tall!

Russell's analysis will have us render Malika's sentence as:

7′. There is exactly one gin-drinker [in the corner], and whomever is a gin-drinker [in the corner] is very tall.

But now suppose that the very tall man is actually drinking *water* from a martini glass; whereas the very short man is drinking a pint of (neat) gin. By Russell's analysis, Malika has said something false. But don't we want to say that Malika has said something *true*?

Again, one might wonder how clear our intuitions are on this case. We can all agree that Malika intended to pick out a particular man, and say something true of him (that he was tall). On Russell's analysis, she actually picked out a different man (the short one), and consequently said something false of him. But maybe advocates of Russell's analysis only need to explain *why* Malika's intentions were frustrated, and so why she said something false. This is easy enough to do: Malika said something false because she had false beliefs about the men's drinks; if Malika's beliefs about the drinks had been true, then she would have said something true.[6]

To say much more here would lead us into deep philosophical waters. That would be no bad thing, but for now it would distract us from the immediate purpose of learning formal logic. So, for now, we shall stick with Russell's analysis of definite descriptions, when it comes to putting things into FOL. It is certainly the best that we can offer, without significantly revising our logic. And it is quite defensible as an analysis.

Practice exercises

A. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid \text{people}\}$
Kx : $\_\_\_\_\_x$ knows the combination to the safe.
Sx : $\_\_\_\_\_x$ is a spy.
Vx : $\_\_\_\_\_x$ is a vegetarian.
Txy : $\_\_\_\_\_x$ trusts $\_\_\_\_\_y$.
h : Hofthor

[4] This is Neale's (1990) line.

[5] Keith Donnellan, 'Reference and Definite Descriptions', 1966, *Philosophical Review 77*, pp. 281–304.

[6] Interested parties should read Saul Kripke, 'Speaker Reference and Semantic Reference', 1977, in French et al (eds.), *Contemporary Perspectives in the Philosophy of Language*, Minneapolis: University of Minnesota Press, pp. 6-27.

i : Ingmar

symbolise the following sentences in FOL:

1. Hofthor trusts a vegetarian.
2. Everyone who trusts Ingmar trusts a vegetarian.
3. Everyone who trusts Ingmar trusts someone who trusts a vegetarian.
4. Only Ingmar knows the combination to the safe.
5. Ingmar trusts Hofthor, but no one else.
6. The person who knows the combination to the safe is a vegetarian.
7. The person who knows the combination to the safe is not a spy.

B. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid$ card in a standard deck$\}$
Bx : $\_\_\_\_\_x$ is black.
Cx : $\_\_\_\_\_x$ is a club.
Dx : $\_\_\_\_\_x$ is a deuce.
Jx : $\_\_\_\_\_x$ is a jack.
Mx : $\_\_\_\_\_x$ is a man with an axe.
Ox : $\_\_\_\_\_x$ is one-eyed.
Wx : $\_\_\_\_\_x$ is wild.

symbolise each sentence in FOL:

1. All clubs are black cards.
2. There are no wild cards.
3. There are at least two clubs.
4. There is more than one one-eyed jack.
5. There are at most two one-eyed jacks.
6. There are two black jacks.
7. There are four deuces.
8. The deuce of clubs is a black card.
9. One-eyed jacks and the man with the axe are wild.
10. If the deuce of clubs is wild, then there is exactly one wild card.
11. The man with the axe is not a jack.
12. The deuce of clubs is not the man with the axe.

C. Using the following symbolisation key:

$\mathcal{D} =$ $\{x \mid$ animals in the world$\}$
Bx : $\_\_\_\_\_x$ is in Farmer Brown's field.
Hx : $\_\_\_\_\_x$ is a horse.
Px : $\_\_\_\_\_x$ is a Pegasus.
Wx : $\_\_\_\_\_x$ has wings.

symbolise the following sentences in FOL:

1. There are at least three horses in the world.
2. There are at least three animals in the world.
3. There is more than one horse in Farmer Brown's field.
4. There are three horses in Farmer Brown's field.

5. There is a single winged creature in Farmer Brown's field; any other creatures in the field must be wingless.
6. The Pegasus is a winged horse.
7. The animal in Farmer Brown's field is not a horse.
8. The horse in Farmer Brown's field does not have wings.

D. In this section, I symbolised 'Nick is the traitor' by '$\exists x(Tx \wedge \forall y(Ty \rightarrow x = y) \wedge x = n)$'. Two equally good symbolisations would be:

- $Tn \wedge \forall y(Ty \rightarrow n = y)$
- $\forall y(Ty \leftrightarrow y = n)$

Explain why these would be equally good symbolisations.

Sentences of FOL 22

We know how to represent English sentences in FOL. The time has finally come to define the notion of a *sentence* of FOL.

22.1 Expressions

There are six kinds of symbols in FOL:

Predicates	$A, B, C, \ldots, Z$
with subscripts, as needed	$A_1, B_1, Z_1, A_2, A_{25}, J_{375}, \ldots$
Constants	$a, b, c, \ldots, r$
with subscripts, as needed	$a_1, b_{224}, h_7, m_{32}, \ldots$
Variables	s, t, u, v, w, x, y, z
with subscripts, as needed	$x_1, y_1, z_1, x_2, \ldots$
Connectives	$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
Brackets	(,)
Quantifiers	$\forall, \exists$

We define an EXPRESSION OF FOL as any string of symbols of FOL. Take any of the symbols of FOL and write them down, in any order, and you have an expression.

22.2 Extended Metalanguage

Here are the symbols that we will use in our metalanguage to talk about expressions in FOL.

Meta-variables for terms	$\mathbf{t}$, with subscripts as needed.
Meta-variables for constants	$\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, with subscripts as needed.
Meta-variables for variables	$\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$, , with subscripts as needed.
Meta-variables for many-place predicates	$\mathbf{\mathcal{R}}$, with subscripts as needed.
Meta-variables for formulas with at least one free occurrence of $\mathbf{x}$	$\phi(\mathbf{x})$
Meta-variables for formulas in which $\mathbf{x}$ is bound	$\forall\mathbf{x}\phi(\mathbf{x})$, $\exists\mathbf{x}\phi(\mathbf{x})$

22.3 Terms and formulas

In §6, we went straight from the statement of the vocabulary of TFL to the definition of a sentence of TFL. In FOL, we shall have to go via an intermediary stage: via the notion of a *formula*. The intuitive idea is that a formula is any sentence, or anything which can be turned into a sentence by adding quantifiers out front. But this will take some unpacking.

We start by defining the notion of a term.

> A TERM is any name or any variable.

So, here are some terms:

$$a, b, x, x_1x_2, y, y_{254}, z$$

We next need to define atomic formulas.

> 1. If $\mathcal{R}$ is an n-place predicate and $t_1, t_2, \ldots, t_n$ are terms, then $\mathcal{R}t_1t_2\ldots t_n$ is an atomic formula.
> 2. If t_1 and t_2 are terms, then $t_1 = t_2$ is an atomic formula.
> 3. Nothing else is an atomic formula.

The swashfonts (i.e. $\mathcal{R}, \mathcal{x}, \mathcal{t}$) are part of our metalanguage, and their use follows the conventions laid down in §7. So, '$\mathcal{R}$' is not itself a predicate of FOL. Rather, it is a symbol of our metalanguage (augmented English) that we use to talk about any predicate of FOL. Similarly, 't_1' is not a term of FOL, but a symbol of the metalanguage that we can use to talk about any term of FOL. So, where 'F' is a one-place predicate, 'G' is a three-place predicate, and 'S' is a six-place predicate, here are some atomic formulas:

$$x = a$$
$$a = b$$
$$Fx$$
$$Fa$$
$$Gxay$$
$$Gaaa$$

$$Sx_1x_2abyx_1$$
$$Sby_{254}zaaz$$

Notice that Fx is an atomic formula. But because x occurs free in it, i.e. it isn't bound by a quantifier, Fx is *not* a sentence in FOL.

Once we know what atomic formulas are, we can offer recursion clauses to define arbitrary formulas. The first few clauses are exactly the same as for TFL.

1. Every atomic formula is a formula.
2. If ϕ is a formula, then $\neg\phi$ is a formula.
3. If ϕ and ψ are formulas, then $(\phi \wedge \psi)$ is a formula.
4. If ϕ and ψ are formulas, then $(\phi \vee \psi)$ is a formula.
5. If ϕ and ψ are formulas, then $(\phi \rightarrow \psi)$ is a formula.
6. If ϕ and ψ are formulas, then $(\phi \leftrightarrow \psi)$ is a formula.
7. If ϕ is a formula, χ is a variable, ϕ contains at least one occurrence of χ, and ϕ contains neither $\forall\chi$ nor $\exists\chi$, then $\forall\chi\phi$ is a formula.
8. If ϕ is a formula, χ is a variable, ϕ contains at least one occurrence of χ, and ϕ contains neither $\forall\chi$ nor $\exists\chi$, then $\exists\chi\phi$ is a formula.
9. Nothing else is a formula.

So, assuming again that 'F' is a one-place predicate, 'G' is a three-place predicate and 'H' is a six place-predicate, here are some formulas:

$$Fx$$
$$Gayz$$
$$Syzyayx$$
$$(Gayz \rightarrow Syzyayx)$$
$$\forall z(Gayz \rightarrow Syzyayx)$$
$$Fx \leftrightarrow \forall z(Gayz \rightarrow Syzyayx)$$
$$\exists y(Fx \leftrightarrow \forall z(Gayz \rightarrow Syzyayx))$$
$$\forall x\exists y(Fx \leftrightarrow \forall z(Gayz \rightarrow Syzyayx))$$

But this is *not* a formula:

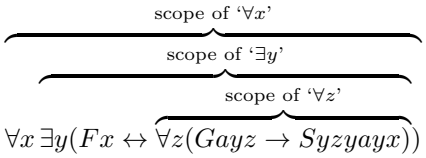
$$\forall x\exists xGxxx$$

Certainly '$Gxxx$' is a formula. And '$\exists xGxxx$' is therefore also a formula. But we cannot form a new formula by putting '$\forall x$' at the front. This violates the constraints on clause 7 of our recursive definition: '$\exists xGxxx$' contains at least one occurrence of 'x', but it already contains '$\exists x$'.

These constraints have the effect of ensuring that variables only serve one master at any one time (see §19). And in fact, we can now give a formal definition of scope, which incorporates the definition of the scope of a quantifier. Here we follow the case of TFL, though we note that a logical operator can be either a connective or a quantifier:

> The MAIN LOGICAL OPERATOR in a formula is the operator that was introduced last, when that formula was constructed using the recursion rules.
>
> The SCOPE of a logical operator in a formula is the subformula for which that operator is the main logical operator.

So we can graphically illustrate the scope of the quantifiers in the preceding example thus:



22.4 Sentences

Recall that we are largely concerned in logic with assertoric sentences: sentences that can be either true or false. Many formulas are not sentences. Consider the following symbolisation key:

$\mathcal{D}$ = {x | people}
Lxy : _____$_x$ loves _____$_y$
b : Boris

Consider the atomic formula 'Lzz'. All atomic formula are formulas, so 'Lzz' is a formula. But can it be true or false? You might think that it will be true just in case the person named by 'z' loves herself, in the same way that 'Lbb' is true just in case Boris (the person named by 'b') loves himself. *But 'z' is a variable, and does not name anyone or any thing.*

Of course, if we put an existential quantifier out front, obtaining '$\exists zLzz$', then this would be true iff someone loves herself. Equally, if we wrote '$\forall zLzz$', this would be true iff everyone loves herself. The point is that we need a quantifier to tell us how to deal with a variable.

Let's make this idea precise.

> A BOUND VARIABLE is an occurrence of a variable χ that is within the scope of either $\forall\chi$ or $\exists\chi$.
>
> A FREE VARIABLE is any variable that is not bound.

For example, consider the formula

$$\forall x(Ex \vee Dy) \rightarrow \exists z(Ex \rightarrow Lzx)$$

The scope of the universal quantifier '$\forall x$' is '$\forall x(Ex \vee Dy)$', so the first 'x' is bound by the universal quantifier. However, the second and third occurrence of 'x' are free. Equally, the 'y' is free. The scope of the existential quantifier '$\exists z$' is '$(Ex \rightarrow Lzx)$', so 'z' is bound.

Finally we can say the following.

> A SENTENCE of FOL is any formula of FOL that contains no free variables.

22.5 Bracketing conventions

We will adopt the same notational conventions governing brackets that we did for TFL (see §6 and §10.3.)

First, we may omit the outermost brackets of a formula.

Second, we may use square brackets, '[' and ']', in place of brackets to increase the readability of formulas.

Third, we may omit brackets between each pair of conjuncts when writing long series of conjunctions.

Fourth, we may omit brackets between each pair of disjuncts when writing long series of disjunctions.

Practice exercises

A. Identify which variables are bound and which are free.

1. $\exists x Lxy \wedge \forall y Lyx$
2. $\forall x Ax \wedge Bx$
3. $\forall x(Ax \wedge Bx) \wedge \forall y(Cx \wedge Dy)$
4. $\forall x \exists y[Rxy \rightarrow (Jz \wedge Kx)] \vee Ryx$
5. $\forall x_1(Mx_2 \leftrightarrow Lx_2x_1) \wedge \exists x_2 Lx_3x_2$

Chapter 6

Models

Extensionality 23

Recall that TFL is a truth-functional language. Its connectives are all truth-functional, and *all* that we can do with TFL is key sentences to particular truth values. We can do this *directly*. For example, we might stipulate that the TFL sentence 'P' is to be true. Alternatively, we can do this *indirectly*, offering a symbolisation key, e.g.:

P Big Ben is in London

But recall from §9 that this should be taken to mean:

- The TFL sentence 'P' is to take the same truth value as the English sentence 'Big Ben is in London' (whatever that truth value may be)

The point that I emphasised is that TFL cannot handle differences in meaning that go beyond mere differences in truth value.

23.1 Symbolising versus translating

FOL has some similar limitations. It gets beyond mere truth values, since it enables us to split up sentences into terms, predicates and quantifier expressions. This enables us to consider what is *true of* some particular object, or of some or all objects. But we can do no more than that.

When we provide a symbolisation key for some FOL predicates, such as:

Cx : $\_\_\_\_\_x$ lectures logic at Agnes Scott College in 2018

we do not carry the *meaning* of the English predicate across into our FOL predicate. We are simply stipulating something like the following:

- 'Cx' and '$\_\_\_\_\_x$ lectures logic at Agnes Scott College in 2018' are to be *true of* exactly the same things.

So, in particular:

- 'Cx' is to be true of all and only those things which lectures logic at Agnes Scott College in 2018 (whatever those things might be).

This is an indirect stipulation. Alternatively we can stipulate predicate extensions directly. We can stipulate that 'Cx' is to be true of Jared Millson, and Jared Millson alone. As it happens, this direct stipulation would have the same effect as the indirect stipulation. But note that the English predicates '_____ is Jared Millson' and '_____ lectures logic at Agnes Scott College in 2018' have very different meanings!

The point is that FOL does not give us any resources for dealing with nuances of meaning. When we interpret FOL, all we are considering is what the predicates are true of. This is normally summed up by saying that FOL is an EXTENSIONAL LANGUAGE.

For this reason, I say only that FOL sentences *symbolise* English sentences. It is doubtful that we are *translating* English into FOL, for translations should preserve meanings.

23.2 A word on extensions

We can stipulate directly what predicates are to be true of. So it is worth noting that our stipulations can be as arbitrary as we like. For example, we could stipulate that 'Hx' should be true of, and only of, the following objects:

Barack Obama
the number π
every top-F key on every piano ever made

Now, the objects that we have listed have nothing particularly in common. But this doesn't matter. Logic doesn't care about what strikes us mere humans as 'natural' or 'similar'. Armed with this interpretation of 'Hx', suppose I now add to my symbolisation key:

d : Barack Obama
n : Michelle Obama
p : the number π

Then 'Hd' and 'Hp' will both be true, on this interpretation, but 'Hn' will be false, since Michelle Obama was not among the stipulated objects.

(This process of explicit stipulation is sometimes described as stipulating the *extension* of a predicate.)

23.3 Many-place predicates

All of this is quite easy to understand when it comes to one-place predicates. But it gets much messier when we consider two-place predicates. Consider a symbolisation key like:

Lxy : _____$_x$ loves _____$_y$

Given what I said above, this symbolisation key should be read as saying:

- 'Lxy' and '_____$_x$ loves _____$_y$' are to be true of exactly the same things

So, in particular:

- 'Lxy' is to be true of x and y (in that order) iff x loves y.

It is important that we insist upon the order here, since love—famously—is not always reciprocated. (Note that 'x' and 'y' here are symbols of augmented English, and that they are being *used*. By contrast, 'x' and 'y' are symbols of FOL, and they are being *mentioned*.)

That is an indirect stipulation. What about a direct stipulation? This is slightly harder. If we *simply* list objects that fall under 'Lxy', we will not know whether they are the lover or the beloved (or both). We have to find a way to include the order in our explicit stipulation.

To do this, we can specify that two-place predicates are true of *pairs* of objects, where the order of the pair is important. Thus we might stipulate that 'Bxy' is to be true of, and only of, the following pairs of objects:

⟨Lenin, Marx⟩
⟨Heidegger, Sartre⟩
⟨Sartre, Heidegger⟩

Here the angle-brackets keep us informed concerning order. Suppose I now add the following stipulations:

l : Lenin
m : Marx
h : Heidegger
s : Sartre

Then 'Blm' will be true, since ⟨Lenin, Marx⟩ was in my explicit list. But 'Bml' will be false, since ⟨Marx, Lenin⟩ was not in my list. However, both 'Bhs' and 'Bsh' will be true, since both ⟨Heidegger, Sartre⟩ and ⟨Sartre, Heidegger⟩ are in my explicit list

To make these ideas more precise, we would need to develop some *set theory*. This will give you some apparatus for dealing with extensions and with ordered pairs (and ordered triples, etc.) However, set theory is not covered in this book. So I shall leave these ideas at an imprecise level. I hope that the general idea is clear enough.

23.4 Semantics for identity

Identity is a special predicate of FOL. We write it a bit differently than other two-place predicates: '$x = y$' instead of 'Ixy' (for example). More important, though, its interpretation is fixed, once and for all.

If two names refer to the same object, then swapping one name for another will not change the truth value of any sentence. So, in particular, if 'a' and 'b' name the same object, then all of the following will be true:

$$Aa \leftrightarrow Ab$$
$$Ba \leftrightarrow Bb$$
$$Raa \leftrightarrow Rbb$$
$$Raa \leftrightarrow Rab$$
$$Rca \leftrightarrow Rcb$$
$$\forall x Rxa \leftrightarrow \forall x Rxb$$

Some philosophers have believed the reverse of this claim. That is, they have believed that when exactly the same sentences (not containing '=') are true of two objects, then they are really just one and the same object after all. This is a highly controversial philosophical claim (sometimes called the *identity of*

indiscernibles) and our logic will not subscribe to it. So we allow that exactly the same things might be true of two *distinct* objects.

To bring this out, consider the following model:

$\mathcal{D}$= {P.D. Magnus, Jared Millson}
a: P.D. Magnus
b: Jared Millson
- For every primitive predicate we care to consider, that predicate is true of *nothing*.

Suppose 'A' is a one-place predicate; then 'Aa' is false and 'Ab' is false, so '$Aa \leftrightarrow Ab$' is true. Similarly, if 'R' is a two-place predicate, then 'Raa' is false and 'Rab' is false, so that '$Raa \leftrightarrow Rab$' is true. And so it goes: every atomic sentence not involving '=' is false, so every biconditional linking such sentences is true. For all that, Jared Millson and P.D. Magnus are two distinct people, not one and the same!

23.5 Models

We defined a *valuation* in TFL as any assignment of truth and falsity to atomic sentences. The counterpart notion in FOL is that of a MODEL. A model consists two things:

- the specification of a domain, $\mathcal{D}$, and
- the specification of an *interpretation function*, I, which assigns exactly one object within the domain to each name (i.e. its reference) and which, for each predicate that we care to consider—other than '='—specifies what things (in what order) the predicate is to be true of (i.e. its EXTENSION).

The symbolisation keys that I considered in chapter 5 consequently give us one very convenient way to present an model. We shall continue to use them throughout this chapter. Still, some special notation will make it easier to represent interpretations. *Moving forward, we will only retain the symbolization of predicates, e.g. 'Ax : $\_\_\_\_\_x$ is such and such' in our representation of models. All other components, such as name references and extensions will be represented using our special notion of interpretation functions.*

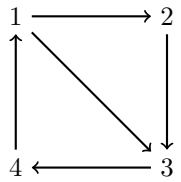
We abbreviate the interpretation of names as follows:

- $I(a)$ = Susan, says that the name 'a' is interpreted as the object Susan in the domain $\mathcal{D}$.
- $I(a)$ = a, says that the name 'a' is interpreted as the object a in the domain $\mathcal{D}$.
- $I(Rebecca)$ = Rebecca, says that the name '*Rebecca*' is interpreted as the object Rebecca in the domain $\mathcal{D}$.

The interpretation function also assigns extensions to predicates. That is, it assigns a set of objects of which the predicate is true. Suppose our domain consists of just three objects, $\mathcal{D}$={Rebecca, Jim, Susan}, and that Rebecca is taller than Susan, and Jim is taller than Rebecca. In this model, the interpretation function would assign the following extension to the two-place predicate Txy: $\_\_\_\_\_x$ is taller than $\_\_\_\_\_y$.

- I(T)= {⟨Rebecca, Susan⟩, ⟨Jim, Rebecca ⟩, ⟨Jim, Susan⟩}.

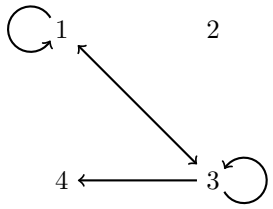
Let's see how this works with a different example. Suppose we want to consider just a single two-place predicate, 'Rxy'. Then we can represent it just by drawing an arrow between two objects, and stipulate that 'Rxy' is to hold of x and y just in case there is an arrow running from x to y in our diagram. As an example, we might offer:



This would be suitable to characterize the following model:

$\mathcal{D}$ = {1, 2, 3, 4}
$I(1)$ =1
$I(2)$ =2
$I(3)$ =3
$I(4)$ =4
$I(R)$ ={⟨1, 2⟩, ⟨2, 3⟩, ⟨3, 4⟩, ⟨4, 1⟩, ⟨1, 3⟩}

Equally we might use:



to characterize the following model:

$\mathcal{D}$ = {1, 2, 3, 4}
$I(1)$ =1
$I(2)$ =2
$I(3)$ =3
$I(4)$ =4
$I(R)$ ={⟨1, 3⟩, ⟨3, 1⟩, ⟨3, 4⟩, ⟨1, 1⟩, ⟨3, 3⟩}

If we wanted, we could make our diagrams more complex. For example, we could add names as labels for particular objects. Equally, to symbolise the extension of a one-place predicate, we might simply draw a ring around some particular objects and stipulate that the thus encircled objects (and only them) are to fall under the predicate 'Hx', say.

Truth in FOL 24

We know what models are. Since, among other things, they tell us which predicates are true of which objects, they will provide us with an account of the truth of atomic sentences. But we must also present a detailed account of what it is for an arbitrary FOL sentence to be true or false in a model.

We know from §22 that there are three kinds of sentence in FOL:

- atomic sentences
- sentences whose main logical operator is a sentential connective
- sentences whose main logical operator is a quantifier

We need to explain truth for all three kinds of sentence.

I shall offer a completely general explanation in this section. However, to try to keep the explanation comprehensible, I shall at several points use the following model:

$\mathcal{D} =$ $\{x \mid$ people born before 2000CE$\}$
$I(a) =$ Aristotle
$I(b) =$ Bush
$Wx :$ $\_\_\_\_\_x$ is wise
$Rxy :$ $\_\_\_\_\_x$ was born before $\_\_\_\_\_y$

This will be my *go-to example* in what follows.

24.1 Atomic sentences

The truth of atomic sentences should be fairly straightforward. The sentence 'Wa' should be true just in case 'Wx' is true of 'a'. Given our go-to model, this is true iff Aristotle is wise. Aristotle is wise. So the sentence is true. Equally, 'Wb' is false on our go-to model.

Likewise, in this model, 'Rab' is true iff the object named by 'a' was born before the object named by 'b'. Well, Aristotle was born before Bush. So 'Rab' is true. Equally, 'Raa' is false: Aristotle was not born before Aristotle. So $I(R) = \{\langle$a,b$\rangle\}$.

Dealing with atomic sentences, then, is very intuitive. When $\mathcal{R}$ is an n-place predicate and $\mathfrak{a}_1, \mathfrak{a}_2, \ldots, \mathfrak{a}_n$ are names,

> $\mathcal{R}\mathfrak{a}_1\mathfrak{a}_2\ldots\mathfrak{a}_n$ is true in a model **iff**
> $\mathcal{R}$ is true of the objects named by $\mathfrak{a}_1, \mathfrak{a}_2, \ldots, \mathfrak{a}_n$ in that model (considered in that order), i.e. $I(\mathcal{R}) = \{\langle \mathfrak{a}_1, \mathfrak{a}_2, \ldots, \mathfrak{a}_n\rangle\}$.

Recall, though, that there is a second kind of atomic sentence—two names connected by an identity sign constitute an atomic sentence. This kind of atomic sentence is also easy to handle. Where $\mathfrak{a}$ and $\mathfrak{b}$ are any names,

> $\mathfrak{a} = \mathfrak{b}$ is true in a model **iff**
> $\mathfrak{a}$ and $\mathfrak{b}$ name the very same object in that model,
> i.e. $I(\mathfrak{a}) = \mathfrak{c}$ and $I(\mathfrak{b}) = \mathfrak{c}$.

So in our go-to model, '$a = b$' is false, since Aristotle is distinct from Bush.

24.2 Sentential connectives

We saw in §22 that FOL sentences can be built up from simpler ones using the truth-functional connectives that were familiar from TFL. The rules governing these truth-functional connectives are *exactly* the same as they were when we considered TFL. Here they are:

> $\phi \wedge \psi$ is true in a model **iff**
> both ϕ is true and ψ is true in that model
>
> $\phi \vee \psi$ is true in a model **iff**
> either ϕ is true or ψ is true in that model
>
> $\neg\phi$ is true in a model **iff**
> ϕ is false in that model
>
> $\phi \rightarrow \psi$ is true in a model **iff**
> either ϕ is false or ψ is true in that model
>
> $\phi \leftrightarrow \psi$ is true in a model **iff**
> ϕ has the same truth value as ψ in that model

This presents the very same information as the characteristic truth tables for the connectives; it just does it in a slightly different way. Some examples will probably help to illustrate the idea. On our go-to model:

- '$a = a \wedge Wa$' is true
- '$Rab \wedge Wb$' is false because, although 'Rab' is true, 'Wb' is false
- '$a = b \vee Wa$' is true
- '$\neg a = b$' is true
- '$Wa \wedge \neg(a = b \wedge Rab)$' is true, because '$Wa$' is true and '$a = b$' is false

Make sure you understand these examples.

Recall that in 18.3, we discussed the fact that sometimes a predicate is not true of *any* objects in a domain. When a predicate is EMPTY, we say that it has a NULL EXTENSION. When we apply an interpretation function to an empty predicate we get an empty set, which we write as '$\{\,\}$'. So, if no objects in the domain $\mathcal{D}$ are F, then $\mathcal{I}(F) = \{\,\}$.

24.3 When the main logical operator is a quantifier

The exciting innovation in FOL, though, is the use of *quantifiers*. And in fact, expressing the truth conditions for quantified sentences is a bit more fiddly than one might expect.

Here is a naïve first thought. We want to say that '$\forall x Fx$' is true iff 'Fx' is true of everything in the domain. This should not be too problematic: our model will specify directly what 'Fx' is true of.

Unfortunately, this naïve first thought is not general enough. For example, we want to be able to say that '$\forall x \exists y Lxy$' is true just in case '$\exists y Lxy$' is true of everything in the domain. And this is problematic, since our model does not directly specify what '$\exists y Lxy$' is to be true of. Instead, whether or not this is true of something should follow just from the model of 'Lxy', the domain, and the meanings of the quantifiers.

So here is a naïve second thought. We might try to say that '$\forall x \exists y Lxy$' is to be true in a model iff $\exists y L\mathbf{a}y$ is true for *every* name $\mathbf{a}$ that we have included in our model. And similarly, we might try to say that $\exists y L\mathbf{a}y$ is true just in case $L\mathbf{a}\mathbf{b}$ is true for *some* name $\mathbf{b}$ that we have included in our model.

Unfortunately, this is not right either. To see this, observe that in our go-to model, we have only given models for *two* names, 'a' and 'b'. But the domain—all people born before the year 2000CE—contains many more than two people. I have no intention of trying to name *all* of them!

So here is a third thought. (And this thought is not naïve, but correct.) Although it is not the case that we have named *everyone*, each person *could* have been given a name. So we should focus on this possibility of extending a model, by adding a new name. I shall offer a few examples of how this might work, centring on our go-to model, and I shall then present the formal definition.

In our go-to model, '$\exists x Rbx$' should be true. After all, in the domain, there is certainly someone who was born after Bush. Lady Gaga is one of those people. Indeed, if we were to extend our go-to model—temporarily, mind—by adding the name 'c' to refer to Lady Gaga, then 'Rbc' would be true on this extended model. And this, surely, should suffice to make '$\exists x Rbx$' true on the original go-to model.

In our go-to model, '$\exists x(Wx \wedge Rxa)$' should also be true. After all, in the domain, there is certainly someone who was both wise and born before Aristotle. Socrates is one such person. Indeed, if we were to extend our go-to model by letting a new name, 'c', denote Socrates, then '$Wc \wedge Rca$' would be true on this extended model. Again, this should surely suffice to make '$\exists x(Wx \wedge Rxa)$' true on the original go-to model.

In our go-to model, '$\forall x \exists y Rxy$' should be false. After all, consider the last person born in the year 1999. I don't know who that was, but if we were to extend our go-to model by letting a new name, 'd', denote that person, then we would not be able to find anyone else in the domain to denote with some further new name, perhaps 'e', in such a way that 'Rde' would be true. Indeed, no matter *whom* we named with 'e', 'Rde' would be false. And this observation is surely sufficient to make '$\exists y Rdy$' *false* in our extended model. And this is surely sufficient to make '$\forall x \exists y Rxy$' false on the original go-to model.

If you have understood these three examples, then that's what matters. Strictly speaking, though, we still need to give a precise definition of the truth

conditions for quantified sentences. The result, sadly, is a bit ugly, and requires a few new definitions. Brace yourself!

Suppose that ϕ is a formula containing at least one instance of the variable χ, and that χ is free in ϕ. We will write this thus:

$$\phi(\chi)$$

Suppose also that $\mathfrak{c}$ is a name. Then we shall write:

$$\phi(\mathfrak{c}/\chi)$$

for the formula obtained by replacing every occurrence of χ in ϕ with $\mathfrak{c}$. The resulting formula is called a SUBSTITUTION INSTANCE of $\forall\chi\phi$ and $\exists\chi\phi$. $\mathfrak{c}$ is called the INSTANTIATING NAME. So:

$$\exists x(Rex \leftrightarrow Fx)$$

is a substitution instance of

$$\forall y\exists x(Ryx \leftrightarrow Fx)$$

with the instantiating name 'e'.

Armed with this notation, the rough idea is as follows. The sentence $\forall\chi\phi(\chi)$ will be true iff $\phi(\mathfrak{c}/\chi)$ is true no matter what object (in the domain) we name with $\mathfrak{c}$. Similarly, the sentence $\exists\chi\phi$ will be true iff there is *some* way to assign the name $\mathfrak{c}$ to an object that makes $\phi(\mathfrak{c}/\chi)$ true. More precisely, we stipulate:

> $\forall\chi\phi(\chi)$ is true in a model **iff**
> $\phi(\mathfrak{c}/\chi)$ is true in *every* model that extends the original model by assigning an object to any name $\mathfrak{c}$ (without changing the model in any other way).
>
> $\exists\chi\phi(\chi)$ is true in a model **iff**
> $\phi(\mathfrak{c}/\chi)$ is true in *some* model that extends the original model by assigning an object to some name $\mathfrak{c}$ (without changing the model in any other way).

To be clear: all this is doing is formalising (very pedantically) the intuitive idea expressed on the previous page. The result is a bit ugly, and the final definition might look a bit opaque. Hopefully, though, the *spirit* of the idea is clear.

Practice exercises

A. Consider the following model:

$$\begin{aligned} \mathcal{D} &= \{\text{Corwin, Benedict}\} \\ I(c) &= \text{Corwin} \\ I(A) &= \{\text{Corwin, Benedict}\} \\ I(B) &= \{\text{Benedict}\} \\ I(N) &= \{\} \end{aligned}$$

Determine whether each of the following sentences is true or false in that model:

1. Bc
2. $Ac \leftrightarrow \neg Nc$
3. $Nc \rightarrow (Ac \vee Bc)$
4. $\forall x Ax$
5. $\forall x \neg Bx$
6. $\exists x(Ax \wedge Bx)$
7. $\exists x(Ax \rightarrow Nx)$
8. $\forall x(Nx \vee \neg Nx)$
9. $\exists x Bx \rightarrow \forall x Ax$

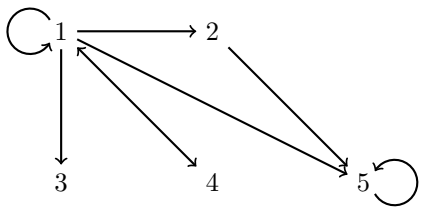
B. Consider the following model:

$\mathcal{D} =$ {Lemmy, Courtney and Eddy}
$I(c) =$ Courtney
$I(e) =$ Eddy
$I(G) =$ {Lemmy, Courtney, Eddy}
$I(H) =$ {Courtney}
$I(M) =$ {Lemmy, Eddy}

Determine whether each of the following sentences is true or false in that model:

1. Hc
2. He
3. $Mc \vee Me$
4. $Gc \vee \neg Gc$
5. $Mc \rightarrow Gc$
6. $\exists x Hx$
7. $\forall x Hx$
8. $\exists x \neg Mx$
9. $\exists x(Hx \wedge Gx)$
10. $\exists x(Mx \wedge Gx)$
11. $\forall x(Hx \vee Mx)$
12. $\exists x Hx \wedge \exists x Mx$
13. $\forall x(Hx \leftrightarrow \neg Mx)$
14. $\exists x Gx \wedge \exists x \neg Gx$
15. $\forall x \exists y(Gx \wedge Hy)$

C. Following the diagram conventions introduced at the end of §23, consider the following model:



Determine whether each of the following sentences is true or false in that model:

1. $\exists x Rxx$
2. $\forall x Rxx$
3. $\exists x \forall y Rxy$

4. $\exists x \forall y Ryx$
5. $\forall x \forall y \forall z((Rxy \wedge Ryz) \rightarrow Rxz)$
6. $\forall x \forall y \forall z((Rxy \wedge Rxz) \rightarrow Ryz)$
7. $\exists x \forall y \neg Rxy$
8. $\forall x(\exists y Rxy \rightarrow \exists y Ryx)$
9. $\exists x \exists y(\neg x = y \wedge Rxy \wedge Ryx)$
10. $\exists x \forall y(Rxy \leftrightarrow x = y)$
11. $\exists x \forall y(Ryx \leftrightarrow x = y)$
12. $\exists x \exists y(\neg x = y \wedge Rxy \wedge \forall z(Rzx \leftrightarrow y = z))$

Semantic concepts 25

Offering a precise definition of truth in FOL was more than a little complicated. But now that we are done, we can define various central logical notions. These will look very similar to the definitions we offered for TFL. However, remember that they concern *models*, rather than valuations. A quick terminological point is needed before we proceed; when a formula is true in some particular model, we say that the model SATISFIES the formula. When no model satisfies a formula, we say that the formula is UNSATISFIABLE.

We will use the symbol '$\vDash$' for FOL much as we did for TFL. So:

$$\phi_1, \phi_2, \ldots, \phi_n \vDash \psi$$

means that there is no model in which all of $\phi_1, \phi_2, \ldots, \phi_n$ are true and in which ψ is false, i.e. every model that satisfies the former also satisfies the latter. Derivatively,

$$\vDash \phi$$

means that ϕ is true in every model.

An FOL sentence ϕ is a LOGICAL TRUTH iff ϕ is true in every model; i.e., $\vDash \phi$.

ϕ is a CONTRADICTION iff ϕ is false in every model or, more simply, ϕ is UNSATISFIABLE; i.e., $\phi \vDash$.

$\phi_1, \phi_2, \ldots \phi_n \therefore \psi$ is VALID IN FOL iff there is no model in which all of the premises are true and the conclusion is false; i.e., $\phi_1, \phi_2, \ldots \phi_n \nvDash \psi$. It is INVALID IN FOL otherwise.

Two FOL sentences ϕ and ψ are LOGICALLY EQUIVALENT iff they are true in exactly the same models as each other; i.e., both $\phi \vDash \psi$ and $\psi \vDash \phi$.

The FOL sentences $\phi_1, \phi_2, \ldots, \phi_n$ are JOINTLY CONSISTENT iff there is some model in which all of the sentences are true. They are JOINTLY INCONSISTENT iff there is no such model.

Using Models 26

26.1 Logical truths and contradictions

Suppose we want to show that '$\exists x Axx \rightarrow Bd$' is *not* a logical truth. This requires showing that the sentence is not true in every model; i.e., that it is false in some model. If we can provide just one model in which the sentence is false, then we will have shown that the sentence is not a logical truth.

In order for '$\exists x Axx \rightarrow Bd$' to be false, the antecedent ('$\exists x Axx$') must be true, and the consequent ('Bd') must be false. To construct such a model, we start by specifying a domain. Keeping the domain small makes it easier to specify what the predicates will be true of, so we shall start with a domain that has just one member. For concreteness, let's say it is the city of Paris.

$\mathcal{D} =$ {Paris}

The name 'd' must name something in the domain, so we have no option but:

$I(d) =$ Paris

Recall that we want '$\exists x Axx$' to be true, so we want all members of the domain to be paired with themselves in the extension of 'A'. We can just offer:

Axy : $\_\_\_\_\_x$ is identical with $\_\_\_\_\_y$

Now 'Add' is true, so it is surely true that '$\exists x Axx$'. Next, we want 'Bd' to be false, so the referent of 'd' must not be in the extension of 'B'. We might simply offer:

Bx : $\_\_\_\_\_x$ is in Germany

Now we have a model where '$\exists x Axx$' is true, but where 'Bd' is false. So there is a model where '$\exists x Axx \rightarrow Bd$' is false. So '$\exists x Axx \rightarrow Bd$' is not a logical truth.

We can just as easily show that '$\exists x Axx \rightarrow Bd$' is not a contradiction. We need only specify a model in which '$\exists x Axx \rightarrow Bd$' is true; i.e., a model in which either '$\exists x Axx$' is false or 'Bd' is true. Here is one:

$\mathcal{D} =$ {Paris}
$I(d) =$ Paris
Axy : $\_\_\_\_\_x$ is identical with $\_\_\_\_\_y$
Bx : $\_\_\_\_\_x$ is in France
$I(A) =$ {⟨Paris, Paris⟩}
$I(B) =$ {Paris}

This shows that there is a model where '$\exists x Axx \rightarrow Bd$' is true. So '$\exists x Axx \rightarrow Bd$' is not a contradiction.

26.2 Logical equivalence

Suppose we want to show that '$\forall x Sx$' and '$\exists x Sx$' are not logically equivalent. We need to construct a model in which the two sentences have different truth values; we want one of them to be true and the other to be false. We start by specifying a domain. Again, we make the domain small so that we can specify extensions easily. In this case, we shall need at least two objects. (If we chose a domain with only one member, the two sentences would end up with the same truth value. In order to see why, try constructing some partial models with one-member domains.) For concreteness, let's take:

$\mathcal{D} =$ {Ornette Coleman, Miles Davis}

We can make '$\exists x Sx$' true by including something in the extension of 'S', and we can make '$\forall x Sx$' false by leaving something out of the extension of 'S'. For concreteness we shall offer:

Sx : $\_\_\_\_\__x$ plays saxophone

Now '$\exists x Sx$' is true, because 'Sx' is true of Ornette Coleman. Slightly more precisely, extend our model by allowing 'c' to name Ornette Coleman. 'Sc' is true in this extended model, so '$\exists x Sx$' was true in the original model. Similarly, '$\forall x Sx$' is false, because 'Sx' is false of Miles Davis. Slightly more precisely, extend our model by allowing 'd' to name Miles Davis and 'Sd' is false in this extended model, so '$\forall x Sx$' was false in the original model. We now have a model in which to the claim that '$\forall x Sx$' and '$\exists x Sx$' are logically equivalent is false. When you provide a model to refute a claim—to logical truth, say, or to entailment—this is called providing a *countermodel*. We represent this countermodel as follows:

Countermodel to $\models \forall x Sx \leftrightarrow \exists x Sx$

$\mathcal{D} =$ {Ornette Coleman, Miles Davis}
$I(c) =$ Ornette Coleman
$I(d) =$ Miles Davis
Sx : $\_\_\_\_\__x$ plays saxophone
$I(S) =$ {Ornette Coleman}.

To show that ϕ is not a logical truth, it suffices to find a model where ϕ is false.
To show that ϕ is not a contradiction, it suffices to find a model where ϕ is true.
To show that ϕ and ψ are not logically equivalent, it suffices to find a model where one is true and the other is false.

26.3 Validity, entailment and consistency

To test for validity, entailment, or consistency, we typically need to produce models that determine the truth value of several sentences simultaneously.

Consider the following argument in FOL:

$$\exists x(Gx \to Ga) \therefore \exists x Gx \to Ga$$

To show that this is invalid, we must make the premise true and the conclusion false. The conclusion is a conditional, so to make it false, the antecedent must be true and the consequent must be false. Clearly, our domain must contain two objects. Let's try:

$\mathcal{D} =$ {Karl Marx, Ludwig von Mises}
$I(a) =$ Karl Marx
$Gx :$ $\_\_\_\_\_{}_x$ hated communism
$I(G) =$ {Ludwig von Mises}

Given that Marx wrote *The Communist Manifesto*, 'Ga' is plainly false in this model. But von Mises famously hated communism. So '$\exists x Gx$' is true in this model. Hence '$\exists x Gx \to Ga$' is false, as required.

But does this model make the premise true? Yes it does! Note that '$Ga \to Ga$' is true. (Indeed, it is a logical truth.) But then certainly '$\exists x(Gx \to Ga)$' is true. So the premise is true, and the conclusion is false, in this model. The argument is therefore invalid. This demonstrates that '$\exists x(Gx \to Ga)$' does *not* entail '$\exists x Gx \to Ga$' and that the sentences '$\exists x(Gx \to Ga)$' and '$\neg(\exists x Gx \to Ga)$' are jointly consistent. The take away is, once again, that the *scope* of quantifiers matters, even when a non-quantified expression, e.g. 'Ga,' is within that scope.

Let's consider a second example. Consider:

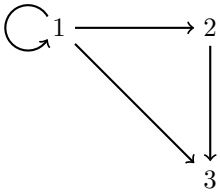
$$\forall x \exists y Lxy \therefore \exists y \forall x Lxy$$

Again, I want to show that this is invalid. To do this, we must make the premises true and the conclusion false. Here is a suggestion:

$\mathcal{D} =$ {x | UK citizens currently in a civil partnership with another UK citizen}
Lxy $\_\_\_\_\_{}_x$ is in a civil partnership with $\_\_\_\_\_{}_y$

The premise is clearly true on this model. Anyone in the domain is a UK citizen in a civil partnership with some other UK citizen. That other citizen will also, then, be in the domain. So for everyone in the domain, there will be someone (else) in the domain with whom they are in a civil partnership. Hence '$\forall x \exists y Lxy$' is true. But the conclusion is clearly false, for that would require that there is some single person who is in a civil partnership with everyone in the domain, and there is no such person. So the argument is invalid. We observe immediately that the sentences '$\forall x \exists y Lxy$' and '$\neg \exists y \forall x Lxy$' are jointly consistent and that '$\forall x \exists y Lxy$' does not entail '$\exists y \forall x Lxy$'.

For a third example, let's mix things up a bit. In §23, we saw that we can present some models using diagrams. For example:



Using the conventions employed in §23, the domain of this model is the first three positive whole numbers, and 'Rxy' is true of x and y just in case there is an arrow from x to y in our diagram. Here are some sentences that the model makes true:

- '$\forall x \exists y Ryx$'
- '$\exists x \forall y Rxy$' witness 1
- '$\exists x \forall y (Ryx \leftrightarrow x = y)$' witness 1
- '$\exists x \exists y \exists z (\neg y = z \wedge Rxy \wedge Rzx)$' witness 2
- '$\exists x \forall y \neg Rxy$' witness 3
- '$\exists x (\exists y Ryx \wedge \neg \exists y Rxy)$' witness 3

This immediately shows that all of the preceding six sentences are jointly consistent. We can use this observation to generate *invalid* arguments, e.g.:

$$\forall x \exists y Ryx, \exists x \forall y Rxy \therefore \forall x \exists y Rxy$$
$$\exists x \forall y Rxy, \exists x \forall y \neg Rxy \therefore \neg \exists x \exists y \exists z (\neg y = z \wedge Rxy \wedge Rzx)$$

and many more besides.

To show that $\phi_1, \phi_2, \ldots, \phi_n \therefore \psi$ is invalid, it suffices to find a model where all of $\phi_1, \phi_2, \ldots, \phi_n$ are true and where ψ is false.
That same model will show that $\phi_1, \phi_2, \ldots, \phi_n$ do not entail ψ.
That same model will show that $\phi_1, \phi_2, \ldots, \phi_n, \neg\psi$ are jointly consistent.

Reasoning about all models 27

27.1 Logical truths and contradictions

We can show that a sentence is *not* a logical truth just by providing one carefully specified model: a model in which the sentence is false. To show that something is a logical truth, on the other hand, it would not be enough to construct ten, one hundred, or even a thousand models in which the sentence is true. A sentence is only a logical truth if it is true in *every* model, and there are infinitely many models. We need to reason about all of them, and we cannot do this by dealing with them one by one!

Sometimes, we can reason about all models fairly easily. For example, we can offer a relatively simple argument that '$Raa \leftrightarrow Raa$' is a logical truth:

> Any relevant model will give 'Raa' a truth value. If 'Raa' is true in a model, then '$Raa \leftrightarrow Raa$' is true in that model. If 'Raa' is false in a model, then '$Raa \leftrightarrow Raa$' is true in that model. These are the only alternatives. So '$Raa \leftrightarrow Raa$' is true in every model. Therefore, it is a logical truth.

This argument is valid, of course, and its conclusion is true. However, it is not an argument in FOL. Rather, it is an argument in English *about* FOL: it is an argument in the metalanguage.

Note another feature of the argument. Since the sentence in question contained no quantifiers, we did not need to think about how to interpret 'a' and 'R'; the point was just that, however we interpreted them, 'Raa' would have some truth value or other. (We could ultimately have given the same argument concerning TFL sentences.)

Here is another bit of reasoning. Consider the sentence '$\forall x(Rxx \leftrightarrow Rxx)$'. Again, it should obviously be a logical truth. But to say precisely why is quite a challenge. We cannot say that '$Rxx \leftrightarrow Rxx$' is true in every model, since '$Rxx \leftrightarrow Rxx$' is not even a *sentence* of FOL (remember that 'x' is a variable, not a name). So we have to be a bit cleverer.

> Consider some arbitrary model. Consider some arbitrary member of the model's domain, which, for convenience, we shall call *obbie*, and suppose we extend our original model by adding a new name, 'c', to name *obbie*. Then either 'Rcc' will be true or it will be false. If 'Rcc' is true, then '$Rcc \leftrightarrow Rcc$' is true. If 'Rcc' is false, then '$Rcc \leftrightarrow Rcc$' will be true. So either way, '$Rcc \leftrightarrow Rcc$' is true. Since there was nothing special about *obbie*—we might have chosen any object—we see that no matter how we extend our original model

> by allowing 'c' to name some new object, '$Rcc \leftrightarrow Rcc$' will be true in the new model. So '$\forall x(Rxx \leftrightarrow Rxx)$' was true in the original model. But we chose our model arbitrarily. So '$\forall x(Rxx \leftrightarrow Rxx)$' is true in every model. It is therefore a logical truth.

This is quite longwinded, but, as things stand, there is no alternative. In order to show that a sentence is a logical truth, we must reason about *all* models.

27.2 Other cases

Similar points hold of other cases too. Thus, we must reason about all models if we want to show:

- that a sentence is a contradiction; for this requires that it is false in *every* model.
- that two sentences are logically equivalent; for this requires that they have the same truth value in *every* model.
- that some sentences are jointly inconsistent; for this requires that there is no model in which all of those sentences are true together; i.e. that, in *every* model, at least one of those sentences is false.
- that an argument is valid; for this requires that the conclusion is true in *every* model where the premises are true.
- that some sentences entail another sentence.

The problem is that, with the tools available to you so far, reasoning about all models is a serious challenge! Let's take just one more example. Here is an argument which is obviously valid:

$$\forall x(Hx \wedge Jx) \therefore \forall x Hx$$

After all, if everything is both H and J, then everything is H. But we can only show that the argument is valid by considering what must be true in every model in which the premise is true. And to show this, we would have to reason as follows:

> Consider an arbitrary model in which the premise '$\forall x(Hx \wedge Jx)$' is true. It follows that, however we expand the model with a new name, for example 'c', '$Hc \wedge Jc$' will be true in this new model. 'Hc' will, then, also be true in this new model. But since this held for *any* way of expanding the model, it must be that '$\forall x Hx$' is true in the old model. And we assumed nothing about the model except that it was one in which '$\forall x(Hx \wedge Jx)$' is true. So any model in which '$\forall x(Hx \wedge Jx)$' is true is one in which '$\forall x Hx$' is true. The argument is valid!

Even for a simple argument like this one, the reasoning is somewhat complicated. For longer arguments, the reasoning can be extremely torturous.

The following table summarises whether a single (counter-)model suffices, or whether we must reason about all models.

	Yes	**No**
logical truth?	all models	one countermodel
contradiction?	all models	one countermodel
equivalent?	all models	one countermodel
consistent?	one model	consider all models
valid?	all models	one countermodel
entailment?	all models	one countermodel

This might usefully be compared with the table at the end of §13. The key difference resides in the fact that TFL concerns truth tables, whereas FOL concerns models. This difference is deeply important, since each truth-table only ever has finitely many lines, so that a complete truth table is a relatively tractable object. By contrast, there are infinitely many models for any given sentence(s), so that reasoning about all models can be a deeply tricky business. In the next section, we'll learn a much easier way to perform this reasoning, with the help of an old friend: Truth Trees!

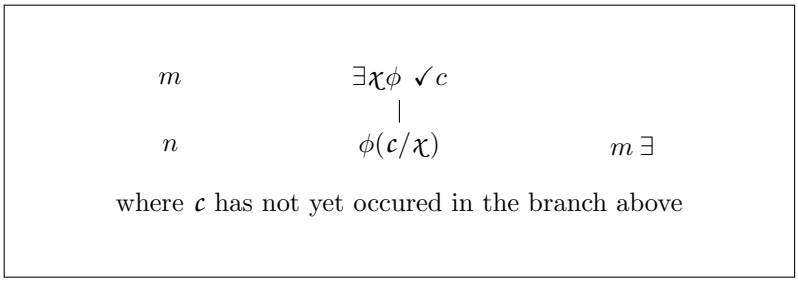
Using Truth Trees to Reason about Models 28

While the informal approach to thinking about models can be helpful, it quickly becomes rather cumbersome when the number of quantified formulas, variables, and names increases. In TFL, we were (implicitly) able to reason about models using truth tables. However, truth tables cannot be used when we are dealing with sentences in FOL (what would we write in the column below $\exists x Fx$?) Luckily we *can* use truth trees.

28.1 Truth Tree Rules for FOL

There is a simple extension of trees that can cope with quantifiers. In addition to our nine rules for logical connectives, we need to add extra rules for the quantifiers.

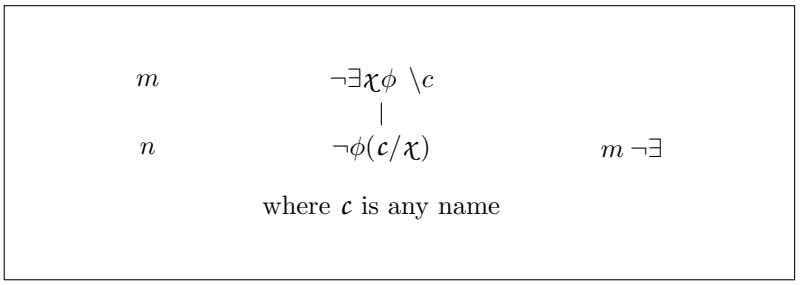
To decompose a formula of the form $\exists \boldsymbol{\chi} \phi$ we extend any open branch in which the formula occurs with an instance of ϕ using a new name that has not occurred in the branch before. When we have done so, we check the original formula and write the name that we have used next to it.



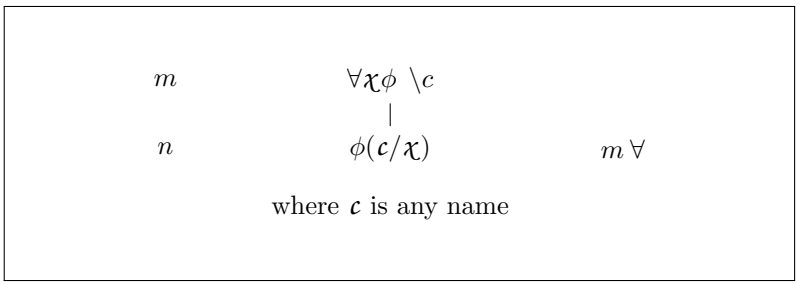
The rationale of this rule is simple. If $\exists x Fx$ is true then some instance of F must be true. We have no idea which object does the job, and we cannot presume we've seen a name for this object already. So, we use a new name, and we are safe. If there is a model satisfying $\exists x Fx$, then this model can also satisfy Fa, because we can choose the interpretation of the new name a to be an object picked out by F.

Given a formula of the form $\neg \exists \boldsymbol{\chi} \phi$ you can extend any open branch in which the formula occurs by *any* instance of ϕ, for any name you wish. The rationale here is similar. If $\neg \exists \boldsymbol{\chi} \phi$ is true then nothing is truly described by ϕ, and hence everything is $\neg \phi$. I can't assume I have all of the names at my disposal when I get to work with the formula so whenever a new name is added

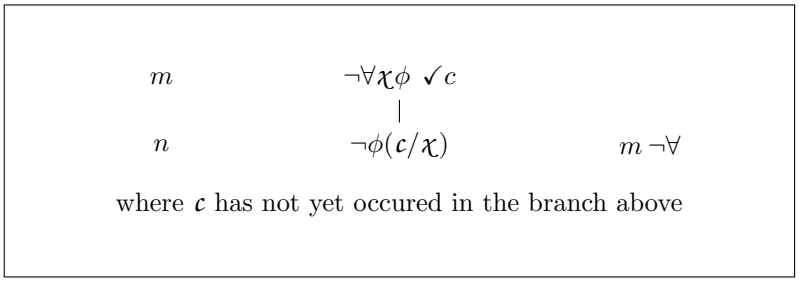
to the branch, I can (and as we shall see, *should*) add a new instance of $\neg\phi$ to that branch with that name. Thus, we can apply the tree rule for negative existential formulas as many times as there are names in the language, and since we can always introduce new names, there is no limit to the number of instantiations we can produce. For this reason, we do not check these formulas when we apply the rule; rather, we simply indicate what name we've used to instantiate the formula.



The universal rule works on the same principle. Given a formula of the form $\forall\chi\phi$ you can extend any open branch in which the formula occurs by any instance of ϕ, for any name we wish.



The negated universal rule is just like the existential rule. To decompose a formula of the form $\neg\forall\chi\phi$ we extend any open branch in which the formula occurs with an instance of $\neg\phi$ using a new name that has not yet occurred in the branch.



As you can see, the rules come in two different kinds. We will call the existential quantifier rule and the negated universal quantifier rule *particular rules*. You use them once only. They introduce a new name to the tree, for a particular object. In the same vein, we will call the universal quantifier rule and the negated existential rule *general rules*. These rules can be used repeatedly. You are allowed to form instances of $\forall\chi\phi$ and $\neg\exists\chi\phi$ as often as

you like. They apply generally to all objects in the domain. We will treat general rules and particular rules differently when creating trees, much in the same way as treating branching rules differently from stacking rules.

Since the general rules do not require us to check off formulas, and since they can be applied repeatedly (indeed, *infinitely*), we must refine our definition of COMPLETED OPEN BRANCH for trees constructed in FOL.

> A branch is a COMPLETED OPEN BRANCH if and only if (1) it is not closed, (2) all complex formulas that can be fully decomposed are decomposed, and (3) for all universally quantified formulas, $\forall(x)\phi$, and negated existentially quantified formulas, $\neg\exists(x)\phi$, occurring in the branch, there is a substitution instance $\phi(\boldsymbol{c}/\boldsymbol{\chi})$, $\neg\phi(\boldsymbol{c}/\boldsymbol{\chi})$ respectively, for *each name* in that branch.

From this definition of a completed branch, we know that in applying the general rules we have to search the branch on which the quantified sentence appears for names. Then, at the bottom of every open branch on which the sentence appears, we must instantiate the original quantified sentence with each name that occurs along that branch. You might think that when we have thus accounted for all the names on the branch we are done with the original quantified sentence. But notice that new names can arise after first working on a universally quantified or negated existentially quantified sentence. In such a case, we must come back and work on the quantified sentence again. Because we must recognize the possibility of having to return to these kinds of quantified sentences, we never check them. Instead, we list the names which we have thus far used in the sentence, so that we know that we do not need to initiate it with those names again. We'll look at an example of how this works in §28.3.

Let's use the rules to test an argument. As with Truth Trees for TFL, a branch is closed when it contains contradictory literals. We just need to remember that in FOL, atomic formulas are represented by predicates attached to names, e.g. Fa. We will test the argument from $\forall x(Fx \to Gx)$ and $\exists x\neg Gx$ to $\exists x\neg Fx$. As before, you start the tree with the premises and the negated conclusion. Here is the top of the tree as we start:

$\forall x(Fx \to Gx)$, $\exists x\neg Gx \therefore \exists x\neg Fx$

1.	$\forall x(Fx \to Gx)$	P
	$\vert$	
2.	$\exists x\neg Gx$	P
	$\vert$	
3.	$\neg\exists x\neg Fx$	NC

These three formulas are all complex, so we have a choice of rules to apply. General rules apply to the first formula and the last formula and a particular rule applies to the middle formula. We will apply the particular rule first, as it will give us a name to use to substitute into the other universal rules we will use later. So, we decompose the existential quantifier, using a name new to the branch. The name a hasn't been used yet, so we make the substitution and extend the branch as follows:

$\forall x(Fx \to Gx)$, $\exists x \neg Gx \therefore \exists x \neg Fx$

1.	$\forall x(Fx \to Gx)$	P
	$\vert$	
2.	$\exists x \neg Gx \checkmark a$	P
	$\vert$	
3.	$\neg \exists x \neg Fx$	NC
	$\vert$	
4.	$\neg Ga$	2 $\exists$

Once we extended the tree with $\neg Ga$, we checked off the original formula to indicate that we have exhausted its content. We have placed an a after the check to indicate that we have decomposed the formula by substituting the name a. This reminds us that we cannot use a to decompose any other particular rules, as a is now in the branch.

On the other hand, we can substitute a in general rules, as these apply to every object in the domain. In fact, that is what we will do now. We have a at our disposal, so we will substitute it into the universal quantifier at the top of the tree:

$\forall x(Fx \to Gx)$, $\exists x \neg Gx \therefore \exists x \neg Fx$

1.	$\forall x(Fx \to Gx) \backslash a$	P
	$\vert$	
2.	$\exists x \neg Gx \checkmark a$	P
	$\vert$	
3.	$\neg \exists x \neg Fx$	NC
	$\vert$	
4.	$\neg Ga$	2 $\exists$
	$\vert$	
5.	$Fa \to Ga$	1 $\forall$

We indicate that we have substituted a into the formula at the top of the tree by writing '$\backslash a$' after the formula. This reminds us that we no longer have to substitute a into this formula. For our next move, let's substitute a into the negated existential formula, $\neg \exists x \neg Fx$.

$\forall x(Fx \to Gx)$, $\exists x \neg Gx \therefore \exists x \neg Fx$

1.	$\forall x(Fx \to Gx)$ $\backslash a$	P
2.	$\exists x \neg Gx$ ✓a	P
3.	$\neg \exists x \neg Fx$ $\backslash a$	NC
4.	$\neg Ga$	2 $\exists$
5.	$Fa \to Ga$	1 $\forall$
6.	$\neg \neg Fa$	3 $\neg\exists$

We get the formula $\neg\neg Fa$, and we write '$\backslash a$' on the third line to indicate that we have substituted the a on this line. Now, we could process $\neg\neg Fa$, but that would be a waste of time, as we can get a closure immediately by decomposing the conditional $Fa \to Ga$.

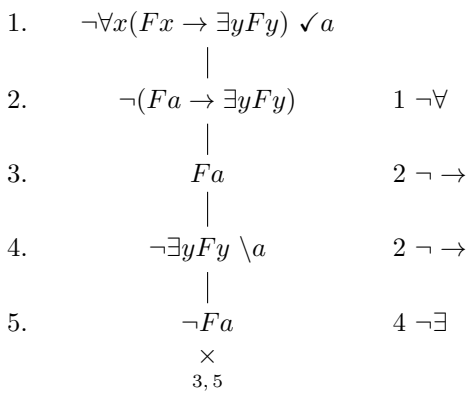
$\forall x(Fx \to Gx)$, $\exists x \neg Gx \therefore \exists x \neg Fx$

1.	$\forall x(Fx \to Gx)$ $\backslash a$	P
2.	$\exists x \neg Gx$ ✓a	P
3.	$\neg \exists x \neg Fx$ $\backslash a$	NC
4.	$\neg Ga$	2 $\exists$
5.	$Fa \to Ga$ ✓	1 $\forall$
6.	$\neg \neg Fa$	3 $\neg\exists$
7.	$\neg Fa$ $\quad$ Ga	5 $\to$
	× $\quad$ ×	
	6, 7 $\quad$ 4, 7	

So, the tree closes, and the argument form is valid. The reasoning is perfectly general, as it applies in any model. The tree tells us that if $\exists x \neg Gx$ is true, there must be some object with the property $\neg G$. Call the object a. Now, since $Fx \to Gx$ holds of every object, it holds of a, so we have $Fa \to Ga$. And we want to try to make $\neg\exists x \neg Fx$ true, so we must make $\neg\neg Fa$ true too. These three requirements are inconsistent, so there is no way to make the three formulas true, in any model, on a domain of any size. The reasoning works no matter how large the domain.

This tree is much simpler than the informal method used in the previous chapters. It shows that the argument has no countermodel.

Let's use the rules to show that $\forall x(Fx \rightarrow \exists yFy)$ is a logical truth. That means it is true in every model, no matter how large or small. To show that this must be the case, we try to find a model in which it is false. The tree is simple:



The first step was to apply the particular rule to the negated universal quantifier. This gave us a new name a, and we learned that $\neg(Fa \rightarrow \exists yFy)$ must be true. This is a negated conditional, so, to make that true, Fa and $\neg\exists yFy$ must be true. But this negated existential is inconsistent with what we already know! We have Fa, but substituting a into $\neg\exists yFy$ gives us $\neg Fa$, a contradiction. Therefore this formula is true in every model. So we know that it is a logical truth. This is fortunate, because the formula ought to be a logical truth. It says that for anything you care to choose, if it's got property F, then something has property F. That sounds like the kind of thing that ought to be true in every model.

28.2 FOL Tree Strategies

1. Use no more rules than needed.
2. Apply particular rules before applying general rules. That is, decompose negated universally quantified expressions and existentially quantified expressions before decomposing universally quantified and negated existentially quantified expressions.
3. Use rules that close branches.
4. Use stacking rules before branching rules.
5. If a universally quantified or negated existentially quantified sentence is on an open branch, add to that branch an instantiation for every name that occurs on the branch. If no name occurs on the branch, pick a name arbitrarily and instantiate it in the branch. If new names occur later in that branch, make sure to instantiate the quantified sentences again in the branch using the new names.
6. Decompose more complex sentences before simpler sentences.

28.3 What Completed Open Branches Mean

Let's look at a tree that doesn't close. We'll construct a tree for $\exists xFx \wedge \exists xGx \vDash \exists x(Fx \wedge Gx)$. Following the strategic rules above, we start by decomposing the existentially quantified formula on line 1. To do so, we must first break down the conjunction. Once we've done that, we apply the $\exists$ rule to each conjunct. Notice, however, that whatever name we use for the first substitution instance will be unavailable for the second. So, we will have to substitute in different names for each.

$\exists xFx \wedge \exists xGx \vDash \exists x(Fx \wedge Gx)$

1.	$\exists xFx \wedge \exists xGx$ ✓	
2.	$\neg\exists x(Fx \wedge Gx)$	
3.	$\exists xFx$ ✓a	1 $\wedge$
4.	$\exists xGx$ ✓b	1 $\wedge$
5.	Fa	3 $\exists$
6.	Gb	4 $\exists$

Now we must deal with the negated existentially quantified sentence. Our fifth strategic rule tells us to apply $\neg\exists$ so as to instantiate the sentence with each name that occurs in the branch. So far, the names that occur in the branch are a and b.

$\exists xFx \wedge \exists xGx \vDash \exists x(Fx \wedge Gx)$

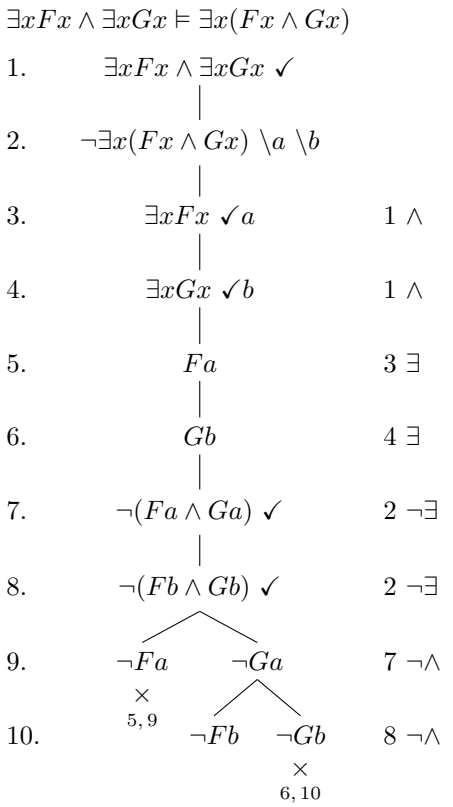
1.	$\exists xFx \wedge \exists xGx$	
2.	$\neg\exists x(Fx \wedge Gx)$ \$a \$b	
3.	$\exists xFx$ ✓a	1 $\wedge$
4.	$\exists xGx$ ✓b	1 $\wedge$
5.	Fa	3 $\exists$
6.	Gb	4 $\exists$
7.	$\neg(Fa \wedge Ga)$	2 $\neg\exists$
8.	$\neg(Fb \wedge Gb)$	2 $\neg\exists$

Next, we decompose the first substitution instance.

$\exists xFx \wedge \exists xGx \vDash \exists x(Fx \wedge Gx)$

1.	$\exists xFx \wedge \exists xGx$ ✓	
2.	$\neg\exists x(Fx \wedge Gx)$ $\backslash a$ $\backslash b$	
3.	$\exists xFx$ ✓a	1 $\wedge$
4.	$\exists xGx$ ✓b	1 $\wedge$
5.	Fa	3 $\exists$
6.	Gb	4 $\exists$
7.	$\neg(Fa \wedge Ga)$ ✓	2 $\neg\exists$
8.	$\neg(Fb \wedge Gb)$	2 $\neg\exists$
9.	$\neg Fa$ $\times$ 5, 9 $\quad \neg Ga$	7 $\neg\wedge$

We can see that the left-hand branch contains a contradiction between Fa on line 5 and $\neg Fa$ on line 9, so we close it. Finally, we decompose the remaining substitution instance (line 8) of the negated existentially quantified sentence (line 2).



This tree does not close; the middle branch is open. Furthermore, the middle branch is COMPLETED. What does this mean? One way of thinking about it is this: a completed open branch in a tree is a world unto itself. For every existentially quantified formula, $\exists\chi\phi$, in the branch, there is a (named) object a occurring in the branch such that $\phi(a)$ is in the branch. Similarly, for every universally quantified formula $\forall\chi\phi$ in the branch, and for every name a in the branch, the formula $\phi(a)$ is also in the branch. This branch describes a coherent, consistent and complete possibility, requiring nothing else to make everything in that branch true.

In short, a closed tree tells us that there is no model where the formulas in the root are valuated as true. A completed open tree tells us that there is at least one model of the formulas in the root where each formula is true.

From a completed open branch, we recover a model in the following way:

1. Identify a completed open branch.
2. From the leaf of the completed open branch move upward writing down all of the literals in that branch.
3. Provide an interpretation for each name in every literal by assigning it one and only one object in the domain.
4. For every name interpreted in terms of an object in the domain, explicitly indicate that those objects are a part of the domain.

5. For each atom, provide an interpretation for each many-place predicate consistent with the truth of the literals in the branch.

In our example tree, the middle branch is COMPLETED and OPEN, so, we will read off the literals in that branch to create a model. The branch contains Fa and $\neg Fb$, $\neg Ga$ and Gb. The names are $I(a) = 1$ and $I(b) = 2$, so we will make our domain: $\mathcal{D} = \{1, 2\}$. The interpretation for F and G is read off the branch by looking at the literals, so $I(\mathrm{F}) = \{1\}$ and $I(\mathrm{G}) = \{2\}$. We now have our model.

This model suffices to make the left-side of the entailment true (there is an F and there is a G) but to make the right-side false (there is no object which is both an F and a G). So, the COMPLETED OPEN BRANCH provides a *countermodel*, to the entailment statement: $\exists xFx \wedge \exists xGx \vDash \exists x(Fx \wedge Gx)$.

Countermodel to $\exists xFx \wedge \exists xGx \vDash \exists x(Fx \wedge Gx)$

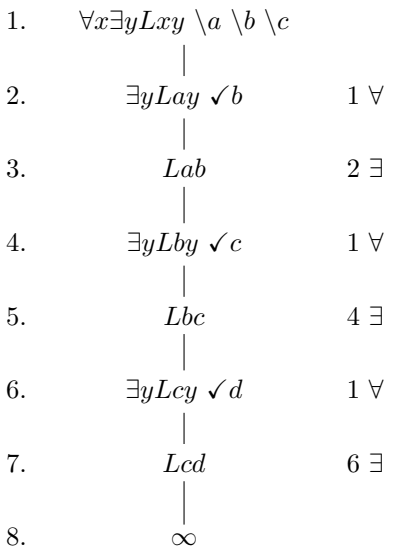
$$\begin{aligned} \mathcal{D} &= \{1, 2\} \\ I(a) &= 1 \\ I(b) &= 2 \\ I(F) &= \{1\} \\ I(G) &= \{2\} \end{aligned}$$

28.4 Infinite Trees

So far, truth trees have provided a mechanical means for testing arguments for validity and sentences for consistency, logical truth, logical contradiction, or logical equivalence. But if logic were a purely mechanical procedure it could not have enough interest to absorb the attention of hundreds of logicians, mathematicians, and philosophers. However, in one way, the truth tree method is not purely mechanical.

Sometimes your tree will never stop. The rules just make you keep going and going and going. What does this mean? Let's look at an example.

Is the formula $\forall x \exists y Lxy$ a contradiction? We will check this by creating a tree for it.



The first interesting thing about this tree is that we had to invent a name to substitute into the universal quantifier at the top of the tree to get the tree going. This is permissible, as we know that any domain must have at least one object. So, we can give that object a name (here it is a) and continue from there. This happens in general: if you have a general formula in a branch, the branch is not completed until you have substituted at least one name into the formula.

The next striking thing about this tree is the fact that it goes on forever. Any branch (and this tree will only ever have one branch) must be infinite. The infinite branch given by completing the tree is reasonably simple to understand. It helps if we use different names instead of $a, b, c, \ldots$, as we will use infinitely many names. If we use different names, $a_1, a_2, a_3, \ldots$, you can see that the branch will contain each of these formulas: $La_1a_2, La_2a_3, La_3a_4, \ldots, La_na_{n+1}$. This means that there are infinitely many models that satisfy the original formula. For instance, we might have one model in which all of the formulas $La_1a_2, La_2a_3, La_3a_4, \ldots, La_na_{n+1}$ are true, and in which La_1a_1 is true, since that is consistent with the information represented in the branch. Conversely, a model in which $\neg La_1a_1$ is true, would also satisfy the formula. The formula would even be satisfied in a model in which there is just one object, so long as $La_1a_2, La_2a_3, La_3a_4, \ldots, La_na_{n+1}$ is true, that is, so long as all the names in the model refer to the same object in the domain. Likewise, a model with just one name, a would also satisfy the formula so long as Laa is true.

In the infinite tree we have seen, it is very easy to tell that the tree will go on forever. And it is easy to figure out what infinite interpretation the infinite tree will provide. But in more complicated problems it will not be so easy. The rub is that there can be no mechanical way which will apply to all cases to tell us, in some limited number of steps, whether or not the tree will eventually close. There will always be cases in which, even after thousands of pages, we will still not know whether the tree will close in just a few more steps or whether it will go on forever.

One can show that this problem, or some analogue of it, will come up no matter how we write the rules of logic. Indeed, this is one of the many exciting things about logic. The rules can be mechanically applied. But logic will always leave room for insight and ingenuity. For in difficult problems the mechanical rules may never tell you whether the tree will eventually close. In these cases you can find out only by being clever. Unfortunately, we must stop at this point. But I hope that you have been able to get a glimpse of one of the ways in which logic can be exciting. If you continue your study of logic, you will come to understand why and how the problem of infinite trees is really a very general fact about all formulations of predicate logic, and you will understand the essential limitations of predicate logic in a much more thorough way.

Practice exercises

A. Provide a countermodel that shows that each of the following is not a logical truth:

1. $Da \wedge Db$
2. $\exists x Txh$
3. $Pm \wedge \neg \forall x Px$
4. $\forall z Jz \leftrightarrow \exists y Jy$
5. $\forall x(Wxmn \vee \exists y Lxy)$
6. $\exists x(Gx \rightarrow \forall y My)$
7. $\exists x(x = h \wedge x = i)$

B. Provide a countermodel that shows that the following pairs of sentences are not logically equivalent.

1. Ja, Ka
2. $\exists x Jx$, Jm
3. $\forall x Rxx$, $\exists x Rxx$
4. $\exists x Px \rightarrow Qc$, $\exists x(Px \rightarrow Qc)$
5. $\forall x(Px \rightarrow \neg Qx)$, $\exists x(Px \wedge \neg Qx)$
6. $\exists x(Px \wedge Qx)$, $\exists x(Px \rightarrow Qx)$
7. $\forall x(Px \rightarrow Qx)$, $\forall x(Px \wedge Qx)$
8. $\forall x \exists y Rxy$, $\exists x \forall y Rxy$
9. $\forall x \exists y Rxy$, $\forall x \exists y Ryx$

C. Using a truth tree, extract a model that shows that the following sentences are jointly consistent:

1. $Ma, \neg Na, Pa, \neg Qa$
2. $Lee, Leg, \neg Lge, \neg Lgg$
3. $\neg(Ma \wedge \exists x Ax), Ma \vee Fa, \forall x(Fx \rightarrow Ax)$
4. $Ma \vee Mb, Ma \rightarrow \forall x \neg Mx$
5. $\forall y Gy, \forall x(Gx \rightarrow Hx), \exists y \neg Iy$
6. $\exists x(Bx \vee Ax), \forall x \neg Cx, \forall x\big[(Ax \wedge Bx) \rightarrow Cx\big]$
7. $\exists x Xx, \exists x Yx, \forall x(Xx \leftrightarrow \neg Yx)$
8. $\forall x(Px \vee Qx), \exists x \neg(Qx \wedge Px)$
9. $\exists z(Nz \wedge Ozz), \forall x \forall y(Oxy \rightarrow Oyx)$
10. $\neg \exists x \forall y Rxy, \forall x \exists y Rxy$

D. Using a truth tree, extract a model that shows that the following arguments are invalid:

1. $\forall x(Ax \rightarrow Bx) \therefore \exists xBx$
2. $\forall x(Rx \rightarrow Dx), \forall x(Rx \rightarrow Fx) \therefore \exists x(Dx \wedge Fx)$
3. $\exists x(Px \rightarrow Qx) \therefore \exists xPx$
4. $Na \wedge Nb \wedge Nc \therefore \forall xNx$
5. $Rde, \exists xRxd \therefore Red$
6. $\exists x(Ex \wedge Fx), \exists xFx \rightarrow \exists xGx \therefore \exists x(Ex \wedge Gx)$
7. $\forall xOxc, \forall xOcx \therefore \forall xOxx$
8. $\exists x(Jx \wedge Kx), \exists x\neg Kx, \exists x\neg Jx \therefore \exists x(\neg Jx \wedge \neg Kx)$
9. $Lab \rightarrow \forall xLxb, \exists xLxb \therefore Lbb$

E. Using a truth tree, determine whether the following arguments are valid. If they are invalid, provide a countermodel.

1. $\forall x(Ax \vee Bx), \forall y(Ay \rightarrow Cy), \exists x\neg Cx \therefore \exists xBx$
2. $\exists x(Ax \wedge Bx) \therefore \exists xAx \wedge \exists xBx$
3. $\exists xAx \wedge \exists xBx \therefore \exists x(Ax \wedge Bx)$
4. $\forall x(Ax \rightarrow Bx), \forall x(Rxb \rightarrow Bx) \therefore \forall x((Ax \vee Bx) \rightarrow Rxb)$
5. $\forall x(Ax \rightarrow Bx), \forall x(\neg Rxb \rightarrow \neg Bx) \therefore \forall x((Ax \vee Bx) \rightarrow Rxb)$
6. $\forall x(Ax \rightarrow Bx), \exists y\neg By \rightarrow \exists yAy \therefore \exists xBx$
7. $\forall x\forall y(Lxy \rightarrow Lyx), \exists x\forall yLxy \therefore \forall x\exists yLxy$

Chapter 7

Natural deduction for TFL

The very idea of natural deduction 29

Way back in §2, we said that an argument is valid iff it is impossible to make all of the premises true and the conclusion false.

In the case of TFL, this led us to develop truth tables. Each line of a complete truth table corresponds to a valuation. So, when faced with a TFL argument, we have a very direct way to assess whether it is possible to make all of the premises true and the conclusion false: just thrash through the truth table.

But truth tables do not necessarily give us much *insight*. Consider two arguments in TFL:

$$P \vee Q, \neg P \therefore Q$$
$$P \rightarrow Q, P \therefore Q$$

Clearly, these are valid arguments. You can confirm that they are valid by constructing four-line truth tables. But we might say that they make use of different *forms* of reasoning. And it might be nice to keep track of these different forms of inference.

One aim of a *natural deduction system* is to show that particular arguments are valid, in a way that allows us to understand the reasoning that the arguments might involve. We begin with very basic rules of inference. These rules can be combined, to offer more complicated arguments. Indeed, with just a small starter pack of rules of inference, we hope to capture all valid arguments.

This is a very different way of thinking about arguments.

With truth tables, we directly consider different ways to make sentences true or false. With natural deduction systems, we manipulate sentences in accordance with rules that we have set down as good rules. The latter promises to give us a better insight—or at least, a different insight—into how arguments work.

The move to natural deduction might be motivated by more than the search for insight. It might also be motivated by *necessity*. Consider:

$$A_1 \rightarrow C_1 \therefore (A_1 \wedge A_2 \wedge A_3 \wedge A_4 \wedge A_5) \rightarrow (C_1 \vee C_2 \vee C_3 \vee C_4 \vee C_5)$$

To test this argument for validity, you might use a 1024-line truth table. If you do it correctly, then you will see that there is no line on which all the premises are true and on which the conclusion is false. So you will know that the argument is valid. (But, as just mentioned, there is a sense in which you

will not know *why* the argument is valid.) But now consider:

$$A_1 \rightarrow C_1 \therefore (A_1 \wedge A_2 \wedge A_3 \wedge A_4 \wedge A_5 \wedge A_6 \wedge A_7 \wedge A_8 \wedge A_9 \wedge A_{10}) \rightarrow (C_1 \vee C_2 \vee C_3 \vee C_4 \vee C_5 \vee C_6 \vee C_7 \vee C_8 \vee C_9 \vee C_{10})$$

This argument is also valid—as you can probably tell—but to test it requires a truth table with $2^{20} = 1048576$ lines. In principle, we can set a machine to grind through truth tables and report back when it is finished. In practice, complicated arguments in TFL can become *intractable* if we use truth tables.

When we get to FOL, though, the problem gets dramatically worse. There is nothing like the truth table test for FOL. To assess whether or not an argument is valid, we have to reason about *all* interpretations. But there are infinitely many possible interpretations. We cannot even in principle set a machine to grind through infinitely many possible interpretations and report back when it is finished: it will *never* finish. We either need to come up with some more efficient way of reasoning about all interpretations, or we need to look for something different.

There are, indeed, systems that codify ways to reason about all possible interpretations, and we have become quite familiar with them: truth trees. The tree systems were developed in the 1950s by Evert Beth and Jaakko Hintikka. While the tree method is great in many respects—it's based on our understanding of the meaning of logical expressions (their semantics) and it's purely mechanical—but it has one major drawback: it doesn't really resemble the way we make arguments or reason deductively.

Consider an argument like:

(5) $Fa \therefore \exists x Fx$

This is clearly valid. Sure, we *could* justify it in terms of reasoning through what is true in all possible interpretations, i.e. by constructing a tree. But how would we justify it in terms of reasoning patterns that we *already* recognize as good? What, after all, could be more obvious than the acceptability of the basic form of argument that corresponds to:

> Boris is a fool. Therefore someone is a fool.

Rather than reasoning directly about all valuations (in the case of TFL) or all interpretations (in the case of FOL), we shall try to select a few basic rules of inference. Some of these will govern the behaviour of the sentential connectives. Others will govern the behaviour of the quantifiers and identity. The resulting system of rules will give us a new way to think about the validity of arguments. The modern development of natural deduction dates from simultaneous and unrelated papers by Gerhard Gentzen and Stanisław Jaśkowski (1934). However, the natural deduction system that we shall consider is based largely around work by Frederic Fitch (first published in 1952).

Basic rules for TFL 30

We will develop a NATURAL DEDUCTION system. For each connective, there will be INTRODUCTION rules, that allow us to prove a sentence that has that connective as the main logical operator, and ELIMINATION rules, that allow us to prove something given a sentence that has that connective as the main logical operator

30.1 The idea of a formal proof

A *formal proof* is a sequence of sentences, some of which are marked as being initial assumptions (or premises). The last line of the formal proof is the conclusion. (Henceforth, I shall simply call these 'proofs', but you should be aware that there are *informal proofs* too.)

As an illustration, consider:

$$\neg(A \vee B) \therefore \neg A \wedge \neg B$$

We shall start a proof by writing the premise:

$$\begin{array}{ll} 1 & \neg(A \vee B) \\ \hline \end{array}$$

Note that we have numbered the premise, since we shall want to refer back to it. Indeed, every line on a proof is numbered, so that we can refer back to it.

Note also that we have drawn a line underneath the premise. Everything written above the line is an *assumption*. Everything written below the line will either be something which follows from the assumptions, or it will be some new assumption. We are hoping to conclude that '$\neg A \wedge \neg B$'; so we are hoping ultimately to conclude our proof with

$$\begin{array}{ll} n & \neg A \wedge \neg B \end{array}$$

for some number n. It doesn't matter, which line we end on, but we would obviously prefer a short proof to a long one.

Similarly, suppose we wanted to consider:

$$A \vee B, \neg(A \wedge C), \neg(B \wedge \neg D) \therefore \neg C \vee D$$

The argument has three premises, so we start by writing them all down, numbered, and drawing a line under them:

1	$A \vee B$
2	$\neg(A \wedge C)$
3	$\neg(B \wedge \neg D)$

and we are hoping to conclude with some line:

n	$\neg C \vee D$

All that remains to do is to explain each of the rules that we can use along the way from premises to conclusion. The rules are broken down by our logical connectives.

30.2 Conjunction

Suppose I want to show that Ludwig is both reactionary and libertarian. One obvious way to do this would be as follows: first I show that Ludwig is reactionary; then I show that Ludwig is libertarian; then I put these two demonstrations together, to obtain the conjunction.

Our natural deduction system will capture this thought straightforwardly. In the example given, I might adopt the following symbolisation key:

R: Ludwig is reactionary
L: Ludwig is libertarian

Perhaps I am working through a proof, and I have obtained 'R' on line 8 and 'L' on line 15. Then on any subsequent line I can obtain '$R \wedge L$' thus:

8	R	
15	L	
	$R \wedge L$	$\wedge$I 8, 15

Note that every line of our proof must either be an assumption, or must be justified by some rule. We cite '$\wedge$I 8, 15' here to indicate that the line is obtained by the rule of conjunction introduction ($\wedge$I) applied to lines 8 and 15. I could equally well obtain:

8	R	
15	L	
	$L \wedge R$	$\wedge$I 15, 8

with the citation reverse, to reflect the order of the conjuncts. More generally, here is our conjunction introduction rule:

m	ϕ	
n	ψ	
	$\phi \wedge \psi$	$\wedge$I m, n

To be clear, the statement of the rule is *schematic*. It is not itself a proof. 'ϕ' and 'ψ' are not sentences of TFL. Rather, they are symbols in the metalanguage, which we use when we want to talk about any sentence of TFL (see §7). Similarly, 'm' and 'n' are not a numerals that will appear on any actual proof. Rather, they are symbols in the metalanguage, which we use when we want to talk about any line number of any proof. In an actual proof, the lines are numbered '1', '2', '3', and so forth. But when we define the rule, we use variables to emphasise that the rule may be applied at any point. The rule requires only that we have both conjuncts available to us somewhere in the proof. They can be separated from one another, and they can appear in any order.

The rule is called 'conjunction *introduction*' because it introduces the symbol '$\wedge$' into our proof where it may have been absent. Correspondingly, we have a rule that *eliminates* that symbol. Suppose you have shown that Ludwig is both reactionary and libertarian. You are entitled to conclude that Ludwig is reactionary. Equally, you are entitled to conclude that Ludwig is libertarian. Putting this together, we obtain our conjunction elimination rule(s):

m	$\phi \wedge \psi$	
	ϕ	$\wedge$E m

and equally:

m	$\phi \wedge \psi$	
	ψ	$\wedge$E m

The point is simply that, when you have a conjunction on some line of a proof, you can obtain either of the conjuncts by $\wedge$E. (One point, might be worth emphasising: you can only apply this rule when conjunction is the main logical operator. So you cannot infer 'D' just from '$C \vee (D \wedge E)$'!)

Even with just these two rules, we can start to see some of the power of our formal proof system. Consider:

$$\begin{array}{ll} & [(A \vee B) \rightarrow (C \vee D)] \wedge [(E \vee F) \rightarrow (G \vee H)] \\ \therefore & [(E \vee F) \rightarrow (G \vee H)] \wedge [(A \vee B) \rightarrow (C \vee D)] \end{array}$$

The main logical operator in both the premise and conclusion of this argument is '$\wedge$'. In order to provide a proof, we begin by writing down the premise, which is our assumption. We draw a line below this: everything after this line must follow from our assumptions by (repeated applications of) our rules of inference. So the beginning of the proof looks like this:

1	$[(A \vee B) \rightarrow (C \vee D)] \wedge [(E \vee F) \rightarrow (G \vee H)]$

From the premise, we can get each of the conjuncts by $\wedge$E. The proof now looks like this:

1	$[(A \vee B) \rightarrow (C \vee D)] \wedge [(E \vee F) \rightarrow (G \vee H)]$	
2	$[(A \vee B) \rightarrow (C \vee D)]$	$\wedge$E 1
3	$[(E \vee F) \rightarrow (G \vee H)]$	$\wedge$E 1

So by applying the $\wedge$I rule to lines 3 and 2 (in that order), we arrive at the desired conclusion. The finished proof looks like this:

1	$[(A \vee B) \rightarrow (C \vee D)] \wedge [(E \vee F) \rightarrow (G \vee H)]$	
2	$[(A \vee B) \rightarrow (C \vee D)]$	$\wedge$E 1
3	$[(E \vee F) \rightarrow (G \vee H)]$	$\wedge$E 1
4	$[(E \vee F) \rightarrow (G \vee H)] \wedge [(A \vee B) \rightarrow (C \vee D)]$	$\wedge$I 3, 2

This is a very simple proof, but it shows how we can chain rules of proof together into longer proofs. In passing, note that investigating this argument with a truth table would have required a staggering 256 lines; our formal proof required only four lines.

It is worth giving another example. Way back in §10.3, we noted that this argument is valid:

$$A \wedge (B \wedge C) \therefore (A \wedge B) \wedge C$$

To provide a proof corresponding with this argument, we start by writing:

1	$A \wedge (B \wedge C)$	

From the premise, we can get each of the conjuncts by applying $\wedge$E twice. And we can then apply $\wedge$E twice more, so our proof looks like:

1	$A \wedge (B \wedge C)$	
2	A	$\wedge$E 1
3	$B \wedge C$	$\wedge$E 1
4	B	$\wedge$E 3
5	C	$\wedge$E 3

But now we can merrily reintroduce conjunctions in the order we wanted them, so that our final proof is:

1	$A \wedge (B \wedge C)$	
2	A	$\wedge$E 1
3	$B \wedge C$	$\wedge$E 1
4	B	$\wedge$E 3
5	C	$\wedge$E 3
6	$A \wedge B$	$\wedge$I 2, 4
7	$(A \wedge B) \wedge C$	$\wedge$I 6, 5

Recall that our official definition of sentences in TFL only allowed conjunctions with two conjuncts. When we discussed semantics, we became a bit more relaxed, and allowed ourselves to drop inner brackets in long conjunctions, since the order of the brackets did not affect the truth table. The proof just given suggests that we could also drop inner brackets in all of our proofs. However, this is not standard, and we shall not do this. Instead, we shall return to the more austere bracketing conventions. (Though we will allow ourselves to drop outermost brackets, for legibility.)

Let me offer one final illustration. When using the ∧I rule, there is no need to apply it to different sentences. So we can formally prove 'A' from 'A' as follows:

1	A	
2	$A \wedge A$	∧I 1, 1
3	A	∧E 2

Simple, but effective.

30.3 Conditional

Consider the following argument:

> If Jane is smart then she is fast. Jane is smart. So Jane is fast.

This argument is certainly valid. And it suggests a straightforward conditional elimination rule (→E):

m	$\phi \to \psi$	
n	ϕ	
	ψ	→E m, n

This rule is also sometimes called *modus ponens*. Again, this is an elimination rule, because it allows us to obtain a sentence that may not contain '→', having started with a sentence that did contain '→'. Note that the conditional, and the antecedent, can be separated from one another, and they can appear in any order. However, in the citation for →E, we always cite the conditional first, followed by the antecedent.

The rule for conditional introduction is also quite easy to motivate. The following argument should be valid:

> Ludwig is reactionary. Therefore if Ludwig is libertarian, then Ludwig is both reactionary *and* libertarian.

If someone doubted that this was valid, we might try to convince them otherwise by explaining ourselves as follows:

> Assume that Ludwig is reactionary. Now, *additionally* assume that Ludwig is libertarian. Then by conjunction introduction—which we just discussed—Ludwig is both reactionary and libertarian. Of course, that's conditional on the assumption that Ludwig is libertarian. But this just means that, if Ludwig is libertarian, then he is both reactionary and libertarian.

Transferred into natural deduction format, here is the pattern of reasoning that we just used. We started with one premise, 'Ludwig is reactionary', thus:

1	R

The next thing we did is to make an *additional* assumption ('Ludwig is libertarian'), for the sake of argument. To indicate that we are no longer dealing *merely* with our original assumption ('R'), but with some additional assumption, we continue our proof as follows:

1	R
2	$\quad L$

Note that we are *not* claiming, on line 2, to have proved 'L' from line 1. So we do not need to write in any justification for the additional assumption on line 2. We do, however, need to mark that it is an additional assumption. We do this by drawing a line under it (to indicate that it is an assumption) and by indenting it with a further vertical line (to indicate that it is additional).

With this extra assumption in place, we are in a position to use ∧I. So we could continue our proof:

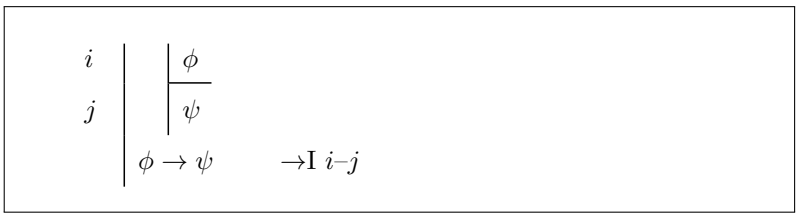
1	R	
2	$\quad L$	
3	$\quad R \wedge L$	∧I 1, 2

So we have now shown that, on the additional assumption, 'L', we can obtain '$R \wedge L$'. We can therefore conclude that, if 'L' obtains, then so does '$R \wedge L$'. Or, to put it more briefly, we can conclude '$L \rightarrow (R \wedge L)$':

1	R	
2	$\quad L$	
3	$\quad R \wedge L$	∧I 1, 2
4	$L \rightarrow (R \wedge L)$	→I 2–3

Observe that we have dropped back to using one vertical line. We have *discharged* the additional assumption, 'L', since the conditional itself follows just from our original assumption, 'R'.

The general pattern at work here is the following. We first make an additional assumption, A; and from that additional assumption, we prove B. In that case, we know the following: If A, then B. This is wrapped up in the rule for conditional introduction:



There can be as many or as few lines as you like between lines i and j.

It will help to offer a second illustration of →I in action. Suppose we want to consider the following:

$$P \to Q, Q \to R \therefore P \to R$$

We start by listing *both* of our premises. Then, since we want to arrive at a conditional (namely, '$P \to R$'), we additionally assume the antecedent to that conditional. Thus our main proof starts:

1	$P \to Q$	
2	$Q \to R$	
3	P	

Note that we have made 'P' available, by treating it as an additional assumption. But now, we can use →E on the first premise. This will yield 'Q'. And we can then use →E on the second premise. So, by assuming 'P' we were able to prove 'R', so we apply the →I rule—discharging 'P'—and finish the proof. Putting all this together, we have:

1	$P \to Q$	
2	$Q \to R$	
3	P	
4	Q	→E 1, 3
5	R	→E 2, 4
6	$P \to R$	→I 3–5

30.4 Additional assumptions and subproofs

The rule →I invoked the idea of making additional assumptions. These need to be handled with some care.

Consider this proof:

1	A	
2	$\quad B$	
3	$\quad B \wedge B$	$\wedge$I 2, 2
4	$\quad B$	$\wedge$E 3
5	$B \rightarrow B$	$\rightarrow$I 2–4

This is perfectly in keeping with the rules we have laid down already. And it should not seem particularly strange. Since '$B \rightarrow B$' is a tautology, no particular premises should be required to prove it.

But suppose we now tried to continue the proof as follows:

1	A	
2	$\quad B$	
3	$\quad B \wedge B$	$\wedge$I 2, 2
4	$\quad B$	$\wedge$E 3
5	$B \rightarrow B$	$\rightarrow$I 2–4
6	B	naughty attempt to invoke $\rightarrow$E 5, 4

If we were allowed to do this, it would be a disaster. It would allow us to prove any atomic sentence letter from any other atomic sentence letter. But if you tell me that Anne is fast (symbolised by 'A'), I shouldn't be able to conclude that Queen Boudica stood twenty-feet tall (symbolised by 'B')! So we must be prohibited from doing this. But how are we to implement the prohibition?

We can describe the process of making an additional assumption as one of performing a *subproof*: a subsidiary proof within the main proof. When we start a subproof, we draw another vertical line to indicate that we are no longer in the main proof. Then we write in the assumption upon which the subproof will be based. A subproof can be thought of as essentially posing this question: *what could we show, if we also make this additional assumption?*

When we are working within the subproof, we can refer to the additional assumption that we made in introducing the subproof, and to anything that we obtained from our original assumptions. (After all, those original assumptions are still in effect.) But at some point, we shall want to stop working with the additional assumption: we shall want to return from the subproof to the main proof. To indicate that we have returned to the main proof, the vertical line for the subproof comes to an end. At this point, we say that the subproof is CLOSED. Having closed a subproof, we have set aside the additional assumption, so it will be illegitimate to draw upon anything that depends upon that additional assumption. Thus we stipulate:

> Any rule whose citation requires mentioning individual lines can mention any earlier lines, *except* for those lines which occur within a closed subproof.

This stipulation rules out the disastrous attempted proof above. The rule of →E requires that we cite two individual lines from earlier in the proof. In the purported proof, above, one of these lines (namely, line 4) occurs within a subproof that has (by line 6) been closed. This is illegitimate.

Closing a subproof is called DISCHARGING the assumptions of that subproof. So we can put the point this way: *you cannot refer back to anything that was obtained using discharged assumptions.*

Subproofs, then, allow us to think about what we could show, if we made additional assumptions. The point to take away from this is not surprising—in the course of a proof, we have to keep very careful track of what assumptions we are making, at any given moment. Our proof system does this very graphically. (Indeed, that's precisely why we have chosen to use *this* proof system.)

Once we have started thinking about what we can show by making additional assumptions, nothing stops us from posing the question of what we could show if we were to make *even more* assumptions? This might motivate us to introduce a subproof within a subproof. Here is an example which only uses the rules of proof that we have considered so far:

1	A	
2	$\quad B$	
3	$\quad\quad C$	
4	$\quad\quad A \wedge B$	∧I 1, 2
5	$\quad C \to (A \wedge B)$	→I 3–4
6	$B \to (C \to (A \wedge B))$	→I 2–5

Notice that the citation on line 4 refers back to the initial assumption (on line 1) and an assumption of a subproof (on line 2). This is perfectly in order, since neither assumption has been discharged at the time (i.e. by line 4).

Again, though, we need to keep careful track of what we are assuming at any given moment. For suppose we tried to continue the proof as follows:

1	A	
2	$\quad B$	
3	$\quad\quad C$	
4	$\quad\quad A \wedge B$	∧I 1, 2
5	$\quad C \to (A \wedge B)$	→I 3–4
6	$B \to (C \to (A \wedge B))$	→I 2–5
7	$C \to (A \wedge B)$	naughty attempt to invoke →I 3–4

This would be awful. If I tell you that Anne is smart, you should not be able to infer that, if Cath is smart (symbolised by 'C') then *both* Anne is smart and Queen Boudica stood 20-feet tall! But this is just what such a proof would suggest, if it were permissible.

The essential problem is that the subproof that began with the assumption 'C' depended crucially on the fact that we had assumed 'B' on line 2. By line 6, we have *discharged* the assumption 'B': we have stopped asking ourselves what we could show, if we also assumed 'B'. So it is simply cheating, to try to help ourselves (on line 7) to the subproof that began with the assumption 'C'. Thus we stipulate, much as before:

> Any rule whose citation requires mentioning an entire subproof can mention any earlier subproof, *except* for those subproofs which occur within some *other* closed subproof.

The attempted disastrous proof violates this stipulation. The subproof of lines 3–4 occurs within a subproof that ends on line 5. So it cannot be invoked in line 7.

It is always permissible to open a subproof with any assumption. However, there is some strategy involved in picking a useful assumption. Starting a subproof with an arbitrary, wacky assumption would just waste lines of the proof. In order to obtain a conditional by →I, for instance, you must assume the antecedent of the conditional in a subproof.

Equally, it is always permissible to close a subproof and discharge its assumptions. However, it will not be helpful to do so, until you have reached something useful.

30.5 Biconditional

The rules for the biconditional will be like double-barrelled versions of the rules for the conditional.

In order to prove '$W \leftrightarrow X$', for instance, you must be able to prove 'X' on the assumption 'W' *and* prove 'W' on the assumption 'X'. The biconditional introduction rule (↔I) therefore requires two subproofs. Schematically, the rule works like this:

i	| ϕ	
j	| ψ	
k	| ψ	
l	| ϕ	
	$\phi \leftrightarrow \psi$	↔I i–j, k–l

There can be as many lines as you like between i and j, and as many lines as you like between k and l. Moreover, the subproofs can come in any order, and the second subproof does not need to come immediately after the first.

The biconditional elimination rule (↔E) lets you do a bit more than the conditional rule. If you have the left-hand subsentence of the biconditional, you can obtain the right-hand subsentence. If you have the right-hand subsentence, you can obtain the left-hand subsentence. So we allow:

m	$\phi \leftrightarrow \psi$	
n	ϕ	
	ψ	$\leftrightarrow$E m, n

and equally:

m	$\phi \leftrightarrow \psi$	
n	ψ	
	ϕ	$\leftrightarrow$E m, n

Note that the biconditional, and the right or left half, can be separated from one another, and they can appear in any order. However, in the citation for $\leftrightarrow$E, we always cite the biconditional first.

30.6 Disjunction

Suppose Ludwig is reactionary. Then Ludwig is either reactionary or libertarian. After all, to say that Ludwig is either reactionary or libertarian is to say something weaker than to say that Ludwig is reactionary.

Let me emphasise this point. Suppose Ludwig is reactionary. It follows that Ludwig is *either* reactionary *or* a kumquat. Equally, it follows that *either* Ludwig is reactionary *or* that kumquats are the only fruit. Equally, it follows that *either* Ludwig is reactionary *or* that God is dead. Many of these things are strange inferences to draw. But there is nothing *logically* wrong with them (even if they maybe violate all sorts of implicit conversational norms).

Armed with all this, I present the disjunction introduction rule(s):

m	ϕ	
	$\phi \vee \psi$	$\vee$I m

and

m	ϕ	
	$\psi \vee \phi$	$\vee$I m

Notice that ψ can be *any* sentence whatsoever. So the following is a perfectly kosher proof:

1	M	
2	$M \vee ([(A \leftrightarrow B) \rightarrow (C \wedge D)] \leftrightarrow [E \wedge F])$	$\vee$I 1

Using a truth table to show this would have taken 128 lines.

The disjunction elimination rule is, though, slightly trickier. Suppose that either Ludwig is reactionary or he is libertarian. What can you conclude? Not that Ludwig is reactionary; it might be that he is libertarian instead. And equally, not that Ludwig is libertarian; for he might merely be reactionary. Disjunctions, just by themselves, are hard to work with.

But suppose that we could somehow show both of the following: first, that Ludwig's being reactionary entails that he is an Austrian economist: second, that Ludwig's being libertarian entails that he is an Austrian economist. Then if we know that Ludwig is either reactionary or libertarian, then we know that, whichever he is, Ludwig is an Austrian economist. This insight can be expressed in the following rule, which is our disjunction elimination ($\vee$E) rule:

m	$\phi \vee \psi$	
i	ϕ	
j	ω	
k	ψ	
l	ω	
	ω	$\vee$E m, i–j, k–l

This is obviously a bit clunkier to write down than our previous rules, but the point is fairly simple. Suppose we have some disjunction, $\phi \vee \psi$. Suppose we have two subproofs, showing us that ω follows from the assumption that ϕ, and that ω follows from the assumption that ψ. Then we can infer ω itself. As usual, there can be as many lines as you like between i and j, and as many lines as you like between k and l. Moreover, the subproofs and the disjunction can come in any order, and do not have to be adjacent.

Here's an analogy that might help you when thinking about disjunction elimination. Suppose you're walking on a path in the woods. Suddenly, you come to a fork where one path stretches out to the left and another extends out to the right. Call the left-side path 'ϕ' and the right-side path 'ψ.' Upon arriving at the fork, you consult your map. The map shows that if you take path ϕ you will eventually arrive at point ω. But the map also shows that if you take ψ you will also arrive at ω. Since you must take either ϕ or ψ, you know that whichever one you choose, you will eventually arrive at ω. Congratulations! You just made an inference using disjunction elimination. You began knowing only that you must take at least one of the two paths, ϕ or ψ. By assuming that you took ϕ, you were able to show, i.e. infer via the information on the map, that you'd get to ω. But by assuming that you took ψ, you were able to show that you'd also get to ω. Since in either case, you get to ω, you don't need to continue entertaining those assumptions or suppositions. Pick either path and you'll get to where you need to go. In other words, you can discharge your assumptions and conclude from the original disjunction (along with the two subproofs) that ω holds.

Some examples might help illustrate this. Consider this argument:

$$(P \wedge Q) \vee (P \wedge R) \therefore P$$

An example proof might run thus:

1	$(P \wedge Q) \vee (P \wedge R)$	
2	$\quad P \wedge Q$	
3	$\quad P$	$\wedge$E 2
4	$\quad P \wedge R$	
5	$\quad P$	$\wedge$E 4
6	P	$\vee$E 1, 2–3, 4–5

Here is a slightly harder example. Consider:

$$A \wedge (B \vee C) \therefore (A \wedge B) \vee (A \wedge C)$$

Here is a proof corresponding to this argument:

1	$A \wedge (B \vee C)$	
2	A	$\wedge$E 1
3	$B \vee C$	$\wedge$E 1
4	$\quad B$	
5	$\quad A \wedge B$	$\wedge$I 2, 4
6	$\quad (A \wedge B) \vee (A \wedge C)$	$\vee$I 5
7	$\quad C$	
8	$\quad A \wedge C$	$\wedge$I 2, 7
9	$\quad (A \wedge B) \vee (A \wedge C)$	$\vee$I 8
10	$(A \wedge B) \vee (A \wedge C)$	$\vee$E 3, 4–6, 7–9

Don't be alarmed if you think that you wouldn't have been able to come up with this proof yourself. The ability to come up with novel proofs will come with practice. The key question at this stage is whether, looking at the proof, you can see that it conforms with the rules that we have laid down. And that just involves checking every line, and making sure that it is justified in accordance with the rules we have laid down.

30.7 Contradiction

We have only one connective left to deal with: negation. But we shall not tackle negation directly. Instead, we shall first think about *contradiction*.

An effective form of argument is to argue your opponent into contradicting themselves. At that point, you have them on the ropes. They have to give up

at least one of their assumptions. We are going to make use of this idea in our proof system, by adding a new symbol, '$\bot$', to our proofs. This should be read as something like 'contradiction!' or 'reductio!' or 'but that's absurd!' And the rule for introducing this symbol is that we can use it whenever we explicitly contradict ourselves, i.e. whenever we find both a sentence and its negation appearing in our proof:

m	ϕ	
n	$\neg\phi$	
	$\bot$	$\bot$I m, n

It does not matter what order the sentence and its negation appear in, and they do not need to appear on adjacent lines. However, we always cite the sentence first, followed by its negation.

Our elimination rule for '$\bot$' is known as *ex falso quod libet*. This means 'anything follows from a contradiction'. And the idea is precisely that: if we obtained a contradiction, symbolised by '$\bot$', then we can infer whatever we like. How can this be motivated, as a rule of argumentation? Well, consider the English rhetorical device '... and if *that's* true, I'll eat my hat'. Since contradictions simply cannot be true, if one *is* true then not only will I eat my hat, I'll have it too.[1] Here is the formal rule:

m	$\bot$	
	ϕ	$\bot$E m

Note that ϕ can be *any* sentence whatsoever.

A final remark. I have said that '$\bot$' should be read as something like 'contradiction!' But this does not tell us much about the symbol. There are, roughly, three ways to approach the symbol.

- We might regard '$\bot$' as a new atomic sentence of TFL, but one which can only ever have the truth value False.
- We might regard '$\bot$' as an abbreviation for some canonical contradiction, such as '$A \wedge \neg A$'. This will have the same effect as the above—obviously, '$A \wedge \neg A$' only ever has the truth value False—but it means that, officially, we do not need to add a new symbol to TFL.
- We might regard '$\bot$', not as a symbol of TFL, but as something more like a *punctuation mark* that appears in our proofs. (It is on a par with the line numbers and the vertical lines, say.)

There is something very philosophically attractive about the third option. But here I shall *officially* plump for the second. '$\bot$' is to be read as abbreviating some canonical contradiction. This means that we can manipulate it, in our proofs, just like any other sentence.

[1] Thanks to Adam Caulton for this.

30.8 Negation

There is obviously a tight link between contradiction and negation. Indeed, the $\bot$I rule essentially behaves as a rule for negation elimination: we introduce '$\bot$' when a sentence and its negation both appear in our proof. So there is no need for us to add a further rule for negation elimination. In essence, $\bot$I does double duty. It serves both to introduce $\bot$ and to eliminate $\neg$.

However, we do need to state a rule for negation introduction. The rule is very simple: if assuming something leads you to a contradiction, then the assumption must be wrong. This thought motivates the following rule:

i	ϕ	
j	$\bot$	
	$\neg\phi$	$\neg$I i–j

There can be as many lines between i and j as you like. To see this in practice, and interacting with negation, consider this proof:

1	D	
2	$\neg D$	
3	$\bot$	$\bot$I 1, 2
4	$\neg\neg D$	$\neg$I 2–3

We shall also add another rule for negation. It is much like the rule used in disjunction elimination, and it requires a little motivation.

Suppose that we can show that if it's sunny outside, then Bill will have brought an umbrella (for fear of burning). Suppose we can also show that, if it's not sunny outside, then Bill will have brought an umbrella (for fear of rain). Well, there is no third way for the weather to be. So, *whatever the weather*, Bill will have brought an umbrella.

This line of thinking motivates the following rule:

i	ϕ	
j	ψ	
k	$\neg\phi$	
l	ψ	
	ψ	TND i–j, k–l

The rule is sometimes called *tertium non datur*, which means 'no third way'. There can be as many lines as you like between i and j, and as many lines as you like between k and l. Moreover, the subproofs can come in any order, and the second subproof does not need to come immediately after the first.

To see the rule in action, consider:

$$P \therefore (P \wedge D) \vee (P \wedge \neg D)$$

Here is a proof corresponding with the argument:

1	P	
2	$\quad D$	
3	$\quad P \wedge D$	$\wedge$I 1, 2
4	$\quad (P \wedge D) \vee (P \wedge \neg D)$	$\vee$I 3
5	$\quad \neg D$	
6	$\quad P \wedge \neg D$	$\wedge$I 1, 5
7	$\quad (P \wedge D) \vee (P \wedge \neg D)$	$\vee$I 6
8	$(P \wedge D) \vee (P \wedge \neg D)$	TND 2–4, 5–7

These are all of the basic rules for the proof system for TFL.

Practice exercises

A. The following two 'proofs' are *incorrect*. Explain the mistakes they make.

1	$\neg L \to (A \wedge L)$	
2	$\quad \neg L$	
3	$\quad A$	$\to$E 1, 2
4	$\quad L$	
5	$\quad \bot$	$\bot$I 4, 2
6	$\quad A$	$\bot$E 5
7	A	TND 2–3, 4–6

1	$A \wedge (B \wedge C)$	
2	$(B \vee C) \to D$	
3	B	$\wedge$E 1
4	$B \vee C$	$\vee$I 3
5	D	$\to$E 4, 2

B. The following three proofs are missing their citations (rule and line numbers). Add them, to turn them into bona fide proofs. Additionally, write down the argument that corresponds to each proof.

1	$P \wedge S$
2	$S \to R$
3	P
4	S
5	R
6	$R \vee E$

1	$A \to D$	
2		$A \wedge B$
3		A
4		D
5		$D \vee E$
6	$(A \wedge B) \to (D \vee E)$	

1	$\neg L \to (J \vee L)$		
2	$\neg L$		
3	$J \vee L$		
4		J	
5		$J \wedge J$	
6		J	
7		L	
8		$\bot$	
9		J	
10	J		

C. Give a proof for each of the following arguments:

1. $J \to \neg J \therefore \neg J$
2. $Q \to (Q \wedge \neg Q) \therefore \neg Q$
3. $A \to (B \to C) \therefore (A \wedge B) \to C$
4. $K \wedge L \therefore K \leftrightarrow L$
5. $(C \wedge D) \vee E \therefore E \vee D$
6. $A \leftrightarrow B, B \leftrightarrow C \therefore A \leftrightarrow C$
7. $\neg F \to G, F \to H \therefore G \vee H$
8. $(Z \wedge K) \vee (K \wedge M), K \to D \therefore D$
9. $P \wedge (Q \vee R), P \to \neg R \therefore Q \vee E$
10. $S \leftrightarrow T \therefore S \leftrightarrow (T \vee S)$
11. $\neg(P \to Q) \therefore \neg Q$
12. $\neg(P \to Q) \therefore P$

Additional rules for TFL 31

In §30, we introduced the basic rules of our proof system for TFL. In this section, we shall add some additional rules to our system. These will make our system much easier to work with. (However, in §34 we will see that they are not strictly speaking *necessary*.)

31.1 Reiteration

The first additional rule is *reiteration* (R). This just allows us to repeat ourselves:

m	ϕ	
	ϕ	R m

Such a rule is obviously legitimate; but one might well wonder how such a rule could ever be useful. Well, consider:

1	$A \to \neg A$	
2	$\quad A$	
3	$\quad \neg A$	$\to$E 1, 2
4	$\quad \neg A$	
5	$\quad \neg A$	R 4
6	$\neg A$	TND 2–3, 4–5

This is a fairly typical use of the R rule.

31.2 Disjunctive syllogism

Here is a very natural argument form.

> Mitt is either in Massachusetts or in DC. He is not in DC. So, he is in Massachusetts.

This inference pattern is called *disjunctive syllogism*. We add it to our proof system as follows:

m	$\phi \vee \psi$	
n	$\neg\phi$	
	ψ	DS m, n

and

m	$\phi \vee \psi$	
n	$\neg\psi$	
	ϕ	DS m, n

As usual, the disjunction and the negation of one disjunct may occur in either order and need not be adjacent. However, we always cite the disjunction first. (This is, if you like, a new rule of disjunction elimination.)

31.3 Modus tollens

Another useful pattern of inference is embodied in the following argument:

> If Mitt has won the election, then he is in the White House. He is not in the White House. So he has not won the election.

This inference pattern is called *modus tollens*. The corresponding rule is:

m	$\phi \to \psi$	
n	$\neg\psi$	
	$\neg\phi$	MT m, n

As usual, the premises may occur in either order, but we always cite the conditional first. (This is, if you like, a new rule of conditional elimination.)

31.4 Double-negation elimination

Another useful rule is *double-negation elimination*. This rule does exactly what it says on the tin:

m	$\neg\neg\phi$	
	ϕ	DNE m

The justification for this is that, in natural language, double-negations tend to cancel out.

That said, you should be aware that context and emphasis can prevent them from doing so. Consider: 'Jane is not *not* happy'. Arguably, one cannot infer 'Jane is happy', since the first sentence should be understood as meaning the same as 'Jane is not *un*happy'. This is compatible with 'Jane is in a state of profound indifference'. As usual, moving to TFL forces us to sacrifice certain nuances of English expressions.

31.5 De Morgan Rules

Our final additional rules are called De Morgan's Laws. (These are named after August De Morgan.) The shape of the rules should be familiar from truth tables.

The first De Morgan rule is:

m	$\neg(\phi \wedge \psi)$	
	$\neg\phi \vee \neg\psi$	DeM m

The second De Morgan is the reverse of the first:

m	$\neg\phi \vee \neg\psi$	
	$\neg(\phi \wedge \psi)$	DeM m

The third De Morgan rule is the *dual* of the first:

m	$\neg(\phi \vee \psi)$	
	$\neg\phi \wedge \neg\psi$	DeM m

And the fourth is the reverse of the third:

m	$\neg\phi \wedge \neg\psi$	
	$\neg(\phi \vee \psi)$	DeM m

These are all of the additional rules of our proof system for TFL.

Practice exercises

A. The following proofs are missing their citations (rule and line numbers). Add them wherever they are required:

1	$W \to \neg B$
2	$A \wedge W$
3	$B \vee (J \wedge K)$
4	W
5	$\neg B$
6	$J \wedge K$
7	K

1	$L \leftrightarrow \neg O$
2	$L \vee \neg O$
3	$\neg L$
4	$\neg O$
5	L
6	$\bot$
7	$\neg\neg L$
8	L

1	$Z \to (C \wedge \neg N)$
2	$\neg Z \to (N \wedge \neg C)$
3	$\neg(N \vee C)$
4	$\neg N \wedge \neg C$
5	$\neg N$
6	$\neg C$
7	Z
8	$C \wedge \neg N$
9	C
10	$\bot$
11	$\neg Z$
12	$N \wedge \neg C$
13	N
14	$\bot$
15	$\neg\neg(N \vee C)$
16	$N \vee C$

B. Give a proof for each of these arguments:

1. $E \vee F$, $F \vee G$, $\neg F \therefore E \wedge G$
2. $M \vee (N \to M) \therefore \neg M \to \neg N$
3. $(M \vee N) \wedge (O \vee P)$, $N \to P$, $\neg P \therefore M \wedge O$
4. $(X \wedge Y) \vee (X \wedge Z)$, $\neg(X \wedge D)$, $D \vee M \therefore M$

Proof-theoretic concepts 32

We shall introduce some new vocabulary. The following expression:

$$\phi_1, \phi_2, \ldots, \phi_n \vdash \omega$$

means that there is some proof which starts with assumptions among $\phi_1, \phi_2, \ldots, \phi_n$ and ends with ω (and no undischarged assumptions other than those we started with). Derivatively, we shall write:

$$\vdash \phi$$

to mean that there is a proof of ϕ with no assumptions.

The symbol '$\vdash$' is called the *single turnstile*. I want to emphasise that this is not the double turnstile symbol ('$\models$') that we used to symbolise entailment in chapters 3 and 6. The single turnstile, '$\vdash$', concerns the existence of proofs; the double turnstile, '$\models$', concerns the existence of valuations (or interpretations, when used for FOL). *They are very different notions.*

Armed with our '$\vdash$' symbol, we can introduce a new terminology.

ϕ is a THEOREM iff $\vdash \phi$

To illustrate this, suppose I want to prove that '$\neg(A \wedge \neg A)$' is a theorem. So I must start my proof without *any* assumptions. However, since I want to prove a sentence whose main logical operator is a negation, I shall want to immediately begin a subproof, with the additional assumption '$A \wedge \neg A$', and show that this leads to contradiction. All told, then, the proof looks like this:

1	$A \wedge \neg A$	
2	A	$\wedge$E 1
3	$\neg A$	$\wedge$E 1
4	$\bot$	$\bot$I 2, 3
5	$\neg(A \wedge \neg A)$	$\neg$I 1–4

We have therefore proved '$\neg(A \wedge \neg A)$' on no (undischarged) assumptions. This particular theorem is an instance of what is sometimes called *the Law of Non-Contradiction*.

To show that something is a theorem, you just have to find a suitable proof. It is typically much harder to show that something is *not* a theorem. To do this, you would have to demonstrate, not just that certain proof strategies fail,

but that *no* proof is possible. Even if you fail in trying to prove a sentence in a thousand different ways, perhaps the proof is just too long and complex for you to make out. Perhaps you just didn't try hard enough.

Here is another new bit of terminology:

> Two sentences ϕ and ψ are PROVABLY EQUIVALENT iff each can be proved from the other; i.e., both $\phi \vdash \psi$ and $\psi \vdash \phi$.

As in the case of showing that a sentence is a theorem, it is relatively easy to show that two sentences are provably equivalent: it just requires a pair of proofs. Showing that sentences are *not* provably equivalent would be much harder: it is just as hard as showing that a sentence is not a theorem.

Here is a third, related, bit of terminology:

> The sentences $\phi_1, \phi_2, \ldots, \phi_n$ are JOINTLY CONTRARY iff a contradiction can be proved from them, i.e. $\phi_1, \phi_2, \ldots, \phi_n \vdash \bot$.

It is easy to show that some sentences are jointly contrary: you just need to prove a contradiction from assuming all the sentences. Showing that some sentences are not jointly contrary is much harder. It would require more than just providing a proof or two; it would require showing that no proof of a certain kind is *possible*.

This table summarises whether one or two proofs suffice, or whether we must reason about all possible proofs.

	Yes	**No**
theorem?	one proof	all possible proofs
equivalent?	two proofs	all possible proofs
not contrary?	all possible proofs	one proof

Practice exercises

A. Show that each of the following sentences is a theorem:

1. $O \to O$
2. $N \vee \neg N$
3. $J \leftrightarrow [J \vee (L \wedge \neg L)]$
4. $((A \to B) \to A) \to A$

B. Provide proofs to show each of the following:

1. $C \to (E \wedge G), \neg C \to G \vdash G$
2. $M \wedge (\neg N \to \neg M) \vdash (N \wedge M) \vee \neg M$
3. $(Z \wedge K) \leftrightarrow (Y \wedge M), D \wedge (D \to M) \vdash Y \to Z$
4. $(W \vee X) \vee (Y \vee Z), X \to Y, \neg Z \vdash W \vee Y$

C. Show that each of the following pairs of sentences are provably equivalent:

1. $R \leftrightarrow E$, $E \leftrightarrow R$
2. G, $\neg\neg\neg\neg G$
3. $T \to S$, $\neg S \to \neg T$

4. $U \to I$, $\neg(U \wedge \neg I)$
5. $\neg(C \to D), C \wedge \neg D$
6. $\neg G \leftrightarrow H$, $\neg(G \leftrightarrow H)$

D. If you know that $\phi \vdash \psi$, what can you say about $(\phi \wedge \omega) \vdash \psi$? What about $(\phi \vee \omega) \vdash \psi$? Explain your answers.

E. In this section, I claimed that it is just as hard to show that two sentences are not provably equivalent, as it is to show that a sentence is not a theorem. Why did I claim this? (*Hint*: think of a sentence that would be a theorem iff ϕ and ψ were provably equivalent.)

Proof strategies 33

There is no simple recipe for proofs, and there is no substitute for practice. Here, though, are some rules of thumb and strategies to keep in mind.

Work backwards from what you want. The ultimate goal is to obtain the conclusion. Look at the conclusion and ask what the introduction rule is for its main logical operator. This gives you an idea of what should happen *just before* the last line of the proof. Then you can treat this line as if it were your goal. Ask what you could do to get to this new goal.

For example: If your conclusion is a conditional $\phi \to \psi$, plan to use the $\to$I rule. This requires starting a subproof in which you assume ϕ. The subproof ought to end with ψ. So, what can you do to get ψ?

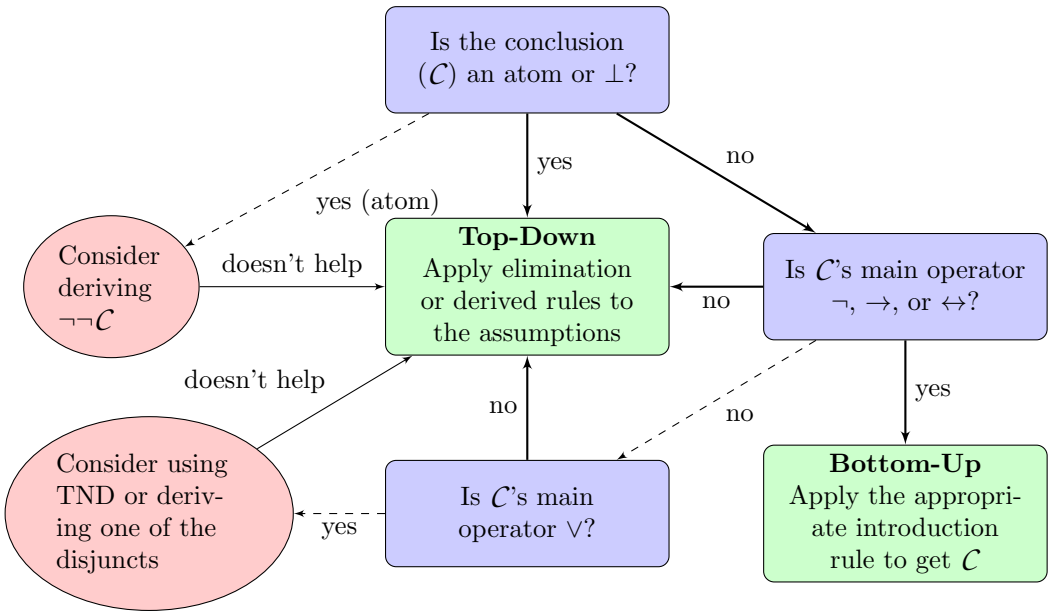
Work forwards from what you have. When you are starting a proof, look at the premises; later, look at the sentences that you have obtained so far. Think about the elimination rules for the main operators of these sentences. These will tell you what your options are.

For a short proof, you might be able to eliminate the premises and introduce the conclusion. A long proof is formally just a number of short proofs linked together, so you can fill the gap by alternately working back from the conclusion and forward from the premises.

Try proceeding indirectly. If you cannot find a way to show ϕ directly, try starting by assuming $\neg\phi$. If a contradiction follows, then you will be able to obtain $\neg\neg\phi$ by $\neg$I, and then ϕ by DNE.

Persist. Try different things. If one approach fails, then try something else.

Flowchart Often, the best way to construct a proof is to begin by working backwards, then to work forwards, then backwards again, and so on until you finish somewhere in the middle. This means toggling back and forth between bottom-up and top-down approaches. It can be rather tricky at first, but once you get the hang, it may become second nature. Here's a flowchart that aims to guide you in this process. The blue rectangles are questions, the green triangles are strategies, and the red ovals are suggestions. The solid lines are directions you should definitely pursue, while the dashed lines are optional considerations.



To use the flowchart, begin by examining the conclusion, $\mathcal{C}$, you are given. If $\mathcal{C}$ is an atom or $\bot$, then there's not much you can do to work backwards, unless it's $\bot$ and you can see how to get a contradiction. If $\mathcal{C}$'s an atom, you might consider whether you can prove $\neg\neg\mathcal{C}$, from which you can derive $\mathcal{C}$ via Double Negation Elimination (DNE). If that doesn't help, or if $\mathcal{C}$ is $\bot$ and you can't see how to get a contradiction, then turn to the assumptions and try applying elimination or derived rules. If $\mathcal{C}$ is neither an atom nor $\bot$, then ask whether $\mathcal{C}$'s main operator is $\neg$, $\rightarrow$ or $\leftrightarrow$. If it is, then open a subproof in order to apply the appropriate introduction rule ($\neg$I, $\rightarrow$I or $\leftrightarrow$I). If $\mathcal{C}$'s main operator isn't any of these, then ask whether it's $\vee$. If it is, you should consider how TND might be applied or see if you can derive just one of the disjuncts by itself, in which case you can get $\mathcal{C}$ by $\vee$I. If this doesn't help, or if $\mathcal{C}$'s main operator isn't $\neg, \rightarrow, \leftrightarrow$ or $\vee$, then return to the assumptions and try applying elimination or derived rules.

As with any flowchart, this one's recursive. That means that when you find yourself with a new conclusion, say, a formula you need to derive in order to close a subproof, you can just run it through the flowchart again. We'll see how this works in the example below. But before moving on, a word of caution. If you end up landing on the strategy of opening a subproof for $\neg$I, $\rightarrow$I or $\leftrightarrow$I you **must** know what rule you are going to apply with that subproof. I can not stress the following enough: **Never start a subproof unless you know what rule you are going to apply with it**. In other words, don't just open a subproof and assume something in the hopes that you'll make progress. You won't. Trust me. Instead, before you even open the subproof, you should write the rule in which it figures on the line just below where the subproof will end. The example will help make this clearer.

Suppose you're asked to construct a proof of the following:

$$A \wedge \neg B \vdash \neg(A \rightarrow B)$$

Using the flowchart, we start by asking whether the conclusion, $\neg(A \to B)$, is an atom or $\bot$. The answer is 'no', so we move to consider where it's main operator is $\neg, \to$ or $\leftrightarrow$. The answer to this question is 'yes,' so we set about applying the appropriate introduction rule, in this case, that's $\neg$I since the main operator is negation. Remember that to derive $\neg\phi$ via $\neg$I, we must open a subproof by assuming ϕ and (somehow) derive $\bot$. So we must open a subproof that starts with the assumption $A \to B$ and that ends with the derivation of $\bot$. Go ahead and write the subproof, leaving yourself plenty of room in the middle. But make sure to write down the rule $\neg$I next to the conclusion, $\neg(A \to B)$, so you don't forget why you opened the subproof! (Of course, at this point, you won't know what lines to cite on the right hand side of that rule.) This should give you something like the following:

1	$A \wedge \neg B$	
2	$\quad A \to B$	
	$\quad \bot$	
	$\neg(A \to B)$	$\neg$I

Notice what's happened. Now, you need to prove that you can derive a contradiction, (i.e. $\bot$) from $A \wedge \neg B$ and $A \to B$. Again, we consult the flowchart, but this time, we use $\bot$ as our conclusion. The flowchart tells us that we should try applying elimination or derived rules to the assumptions. (Remember that because of the subproof, we now have a new assumption to work with, namely $A \to B$.) Looking at the assumptions, it should be pretty obvious that the only elimination rule we can apply is $\wedge$E to the formula on first line (i.e. $A \wedge \neg B$). Hopefully at this point you're starting to see where we're going, but if not, just apply the $\wedge$E rule to derive the each conjunct and then look at the problem again.

1	$A \wedge \neg B$	
2	$\quad A \to B$	
3	$\quad A$	$\wedge$E 1
4	$\quad \neg B$	$\wedge$E 1
	$\quad \bot$	
	$\neg(A \to B)$	$\neg$I

Ok, at this point our conclusion hasn't changed—it's still $\bot$. But we do have some new assumptions, namely A and $\neg B$. So we should still be pursuing

the strategy that the flowchart recommended, namely, applying elimination or derived rules to the assumptions. At this point you hopefully see that line 2 and 3 allow us to apply →E to derive B. That gives us the following.

1	$A \wedge \neg B$	
2	$\quad A \rightarrow B$	
3	$\quad A$	∧E 1
4	$\quad \neg B$	∧E 1
5	$\quad B$	→E 2, 3
	$\quad \bot$	
	$\neg(A \rightarrow B)$	¬I

Lo and behold we have a contradiction! Lines 4 and 5 contradict one another, permitting us to introduce $\bot$ on line 6, which we'd already written in anticipation of closing the subproof for ¬I. Now we just need to cite the appropriate lines for the last two rules and we're all done.

1	$A \wedge \neg B$	
2	$\quad A \rightarrow B$	
3	$\quad A$	∧E 1
4	$\quad \neg B$	∧E 1
5	$\quad B$	→E 2, 3
6	$\quad \bot$	⊥I 4, 5
7	$\neg(A \rightarrow B)$	¬I 2–6

Following the flowchart above can help give you a sense of security as you master the skills of proof-construction. But the flowchart is only a heuristic—a rule of thumb. It won't solve the problem for you. And in some cases, it only offers suggestions. The best way to learn how to construct proofs is to practice, practice, practice!

Derived rules 34

In this section, we shall see why I introduced the rules of our proof system in two separate batches. In particular, I want to show that the additional rules of §31 are not strictly speaking necessary, but can be derived from the basic rules of §30.

34.1 Derivation of Reiteration

Suppose you have some sentence on some line of your deduction:

m	ϕ

You now want to repeat yourself, on some line k. You could just invoke the rule R, introduced in §31. But equally well, you can do this with the *basic* rules of §30:

m	ϕ	
k	$\phi \wedge \phi$	$\wedge$I m
$k+1$	ϕ	$\wedge$E k

To be clear: this is not a proof. Rather, it is a proof *scheme*. After all, it uses a variable, 'ϕ', rather than a sentence of TFL. But the point is simple. Whatever sentences of TFL we plugged in for 'ϕ', and whatever lines we were working on, we could produce a bona fide proof. So you can think of this as a recipe for producing proofs.

Indeed, it is a recipe which shows us that, anything we can prove using the rule R, we can prove (with one more line) using just the *basic* rules of §30. So we can describe the rule R as a *derived* rule, since its justification is derived from our basic rules.

34.2 Derivation of Disjunctive syllogism

Suppose that you are in a proof, and you have something of this form:

m	$\phi \vee \psi$
n	$\neg\phi$

You now want, on line k, to prove ψ. You can do this with the rule of DS, introduced in §31. But equally well, you can do this with the *basic* rules of §30:

m	$\phi \vee \psi$	
n	$\neg\phi$	
k	$\quad \phi$	
$k+1$	$\quad \bot$	$\bot$I k, n
$k+2$	$\quad \psi$	$\bot$E $k+1$
$k+3$	$\quad \psi$	
$k+4$	$\quad \psi \wedge \psi$	$\wedge$I $k+3$, $k+3$
$k+5$	$\quad \psi$	$\wedge$E $k+4$
$k+6$	ψ	$\vee$E m, k–$k+2$, $k+3$–$k+5$

So the DS rule, again, can be derived from our more basic rules. Adding it to our system did not make any new proofs possible. Anytime you use the DS rule, you could always take a few extra lines and prove the same thing using only our basic rules. It is a *derived* rule.

34.3 Derivation of Modus tollens

Suppose you have the following in your proof:

m	$\phi \to \psi$
n	$\neg\psi$

You now want, on line k, to prove $\neg\phi$. You can do this with the rule of MT, introduced in §31. But equally well, you can do this with the *basic* rules of §30:

m	$\phi \to \psi$	
n	$\neg\psi$	
k	$\quad \phi$	
$k+1$	$\quad \psi$	$\to$E m, k
$k+2$	$\quad \bot$	$\bot$I $k+1$, n
$k+3$	$\neg\phi$	$\neg$I k–$k+2$

Again, the rule of MT can be derived from the *basic* rules of §30.

34.4 Derivation of Double-negation elimination

Consider the following deduction scheme:

m	$\neg\neg\phi$	
k	$\quad\phi$	
$k+1$	$\quad\phi$	R k
$k+2$	$\quad\neg\phi$	
$k+3$	$\quad\bot$	$\bot$I $k+2$, m
$k+4$	$\quad\phi$	$\bot$E $k+3$
$k+5$	ϕ	TND k–$k+1$, $k+2$–$k+4$

Again, then, we can derive the DNE rule from the *basic* rules of §30.

34.5 Derivation of De Morgan rules

Here is a demonstration of how we could derive the first De Morgan rule:

m	$\neg(\phi \wedge \psi)$	
k	$\quad\phi$	
$k+1$	$\quad\quad\psi$	
$k+2$	$\quad\quad\phi \wedge \psi$	$\wedge$I k, $k+1$
$k+3$	$\quad\quad\bot$	$\bot$I $k+2$, m
$k+4$	$\quad\neg\psi$	$\neg$I $k+1$–$k+3$
$k+5$	$\quad\neg\phi \vee \neg\psi$	$\vee$I $k+4$
$k+6$	$\quad\neg\phi$	
$k+7$	$\quad\neg\phi \vee \neg\psi$	$\vee$I $k+6$
$k+8$	$\neg\phi \vee \neg\psi$	TND k–$k+5$, $k+6$–$k+7$

Here is a demonstration of how we could derive the second De Morgan rule:

m	$\neg\phi \vee \neg\psi$	
k	$\quad \phi \wedge \psi$	
$k+1$	$\quad \phi$	$\wedge$E k
$k+2$	$\quad \psi$	$\wedge$E k
$k+3$	$\quad\quad \neg\phi$	
$k+4$	$\quad\quad \bot$	$\bot$I $k+1$, $k+3$
$k+5$	$\quad\quad \neg\psi$	
$k+6$	$\quad\quad \bot$	$\bot$I $k+2$, $k+5$
$k+7$	$\quad \bot$	$\vee$E m, $k+3$–$k+4$, $k+5$–$k+6$
$k+8$	$\neg(\phi \wedge \psi)$	$\neg$I k–$k+7$

Similar demonstrations can be offered explaining how we could derive the third and fourth De Morgan rules. These are left as exercises.

Practice exercises

A. Provide proof schemes that justify the addition of the third and fourth De Morgan rules as derived rules.

B. The proofs you offered in response to the practice exercises of §§31–32 used derived rules. Replace the use of derived rules, in such proofs, with only basic rules. You will find some 'repetition' in the resulting proofs; in such cases, offer a streamlined proof using only basic rules. (This will give you a sense, both of the power of derived rules, and of how all the rules interact.)

Chapter 8

Natural deduction for FOL

Basic rules for FOL 35

FOL makes use of all of the connectives of TFL. So proofs in FOL will use all of the basic and derived rules from chapter 7. We shall also use the proof-theoretic notions (particularly, the symbol '$\vdash$') introduced in that chapter. However, we will also need some new basic rules to govern the quantifiers, and to govern the identity sign.

35.1 Universal elimination

From the claim that everything is F, you can infer that any particular thing is F. You name it; it's F. So the following should be fine:

1	$\forall x Rxxd$	
2	$Raad$	$\forall$E 1

We obtained line 2 by dropping the universal quantifier and replacing every instance of 'x' with 'a'. Equally, the following should be allowed:

1	$\forall x Rxxd$	
2	$Rddd$	$\forall$E 1

We obtained line 2 here by dropping the universal quantifier and replacing every instance of 'x' with 'd'. We could have done the same with any other name we wanted.

This motivates the universal elimination rule ($\forall$E):

m	$\forall \chi \phi$	
	$\phi(\mathfrak{c}/\chi)$	$\forall$E m

The notation here was introduced in §24. The point is that you can obtain any *substitution instance* of a universally quantified formula: replace every instance of the quantified variable with any name you like.

I should emphasise that (as with every elimination rule) you can only apply the $\forall$E rule when the universal quantifier is the main logical operator. Thus the following is outright banned:

1	$\forall x Bx \to Bk$	
2	$Bb \to Bk$	naughtily attempting to invoke ∀E 1

This is illegitimate, since '$\forall x$' is not the main logical operator in line 1. (If you need a reminder as to why this sort of inference should be banned, reread §18.)

35.2 Existential introduction

From the claim that some particular thing is an F, you can infer that something is an F. So we ought to allow:

1	$Raad$	
2	$\exists x Raax$	∃I 1

Here, we have replaced the name 'd' with a variable 'x', and then existentially quantified over it. Equally, we would have allowed:

1	$Raad$	
2	$\exists x Rxxd$	∃I 1

Here we have replaced both instances of the name 'a' with a variable, and then existentially generalised. But we do not need to replace *both* instances of a name with a variable. (After all, if Narcissus loves himself, then there is someone who loves Narcissus.) So we would also allow:

1	$Raad$	
2	$\exists x Rxad$	∃I 1

Here we have replaced *one* instance of the name 'a' with a variable, and then existentially generalised. These observations motivate our introduction rule, although to explain it, we shall need to introduce some new notation.

Where ϕ is a sentence containing the name $\mathfrak{c}$, we can emphasise this by writing '$\phi(\mathfrak{c})$'. We shall write '$\phi(\dots\mathfrak{x}\dots\mathfrak{c}\dots)$' to indicate any formula obtained by replacing *some or all* of the instances of the name $\mathfrak{c}$ with the variable $\mathfrak{x}$. Armed with this, our introduction rule is:

m	$\phi(\mathfrak{c})$	
	$\exists\mathfrak{x}\phi(\dots\mathfrak{x}\dots\mathfrak{c}\dots)$	∃I m

$\mathfrak{x}$ must not occur in $\phi(\mathfrak{c})$

The constraint is included to guarantee that any application of the rule yields a sentence of FOL. Thus the following is allowed:

1	$Raad$	
2	$\exists x Rxad$	$\exists$I 1
3	$\exists y \exists x Rxyd$	$\exists$I 2

But this is banned:

1	$Raad$	
2	$\exists x Rxad$	$\exists$I 1
3	$\exists x \exists x Rxxd$	naughtily attempting to invoke $\exists$I 2

since the expression on line 3 contains clashing variables, and so fails to count as a sentence of FOL.

35.3 Empty domains

The following proof combines our two new rules for quantifiers:

1	$\forall x Fx$	
2	Fa	$\forall$E 1
3	$\exists x Fx$	$\exists$I 2

Could this be a bad proof? If anything exists at all, then certainly we can infer that something is F, from the fact that everything is F. But what if *nothing* exists at all? Then it is surely vacuously true that everything is F; however, it does not following that something is F, for there is nothing to *be* F. So if we claim that, as a matter of logic alone, '$\exists x Fx$' follows from '$\forall x Fx$', then we are claiming that, as a matter of *logic alone*, there is something rather than nothing. This might strike us as a bit odd.

Actually, we are already committed to this oddity. In §17, we stipulated that domains in FOL must have at least one member. We then defined a logical truth (of FOL) as a sentence which is true in every interpretation. Since '$\exists x\ x = x$' will be true in every interpretation, this *also* had the effect of stipulating that it is a matter of logic that there is something rather than nothing.

Since it is far from clear that logic should tell us that there must be something rather than nothing, we might well be cheating a bit here.

If we refuse to cheat, though, then we pay a high cost. Here are three things that we want to hold on to:

- $\forall x Fx \vdash Fa$: after all, that was $\forall$E.
- $Fa \vdash \exists x Fx$: after all, that was $\exists$I.
- the ability to copy-and-paste proofs together: after all, reasoning works by putting lots of little steps together into rather big chains.

If we get what we want on all three counts, then we have to countenance that $\forall x Fx \vdash \exists x Fx$. So, if we get what we want on all three counts, the proof system alone tells us that there is something rather than nothing. And if we refuse to

accept that, then we have to surrender one of the three things that we want to hold on to!

Before we start thinking about which to surrender, we might want to ask how *much* of a cheat this is. Granted, it may make it harder to engage in theological debates about why there is something rather than nothing. But the rest of the time, we will get along just fine. So maybe we should just regard our proof system (and FOL, more generally) as having a very slightly limited purview. If we ever want to allow for the possibility of *nothing*, then we shall have to cast around for a more complicated proof system. But for as long as we are content to ignore that possibility, our proof system is perfectly in order. (As, similarly, is the stipulation that every domain must contain at least one object.)

35.4 Universal introduction

Suppose you had shown of each particular thing that it is F (and that there are no other things to consider). Then you would be justified in claiming that everything is F. This would motivate the following proof rule. If you had established each and every single substitution instance of '$\forall x Fx$', then you can infer '$\forall x Fx$'.

Unfortunately, that rule would be utterly unusable. To establish each and every single substitution instance would require proving 'Fa', 'Fb', ..., 'Fj_2', ..., 'Fr_{79002}', ..., and so on. Indeed, since there are infinitely many names in FOL, this process would never come to an end. So we could never apply that rule. We need to be a bit more cunning in coming up with our rule for introducing universal quantification.

Our cunning thought will be inspired by considering:

$$\forall x Fx \therefore \forall y Fy$$

This argument should *obviously* be valid. After all, alphabetical variation ought to be a matter of taste, and of no logical consequence. But how might our proof system reflect this? Suppose we begin a proof thus:

1	$\forall x Fx$	
2	Fa	$\forall$E 1

We have proved 'Fa'. And, of course, nothing stops us from using the same justification to prove 'Fb', 'Fc', ..., 'Fj_2', ..., '$Fr_{79002}, \ldots$', and so on until we run out of space, time, or patience. But reflecting on this, we see that there is a way to prove $F\mathbf{c}$, for any name $\mathbf{c}$. And if we can do it for *any* thing, we should surely be able to say that 'F' is true of *everything*. This therefore justifies us in inferring '$\forall y Fy$', thus:

1	$\forall x Fx$	
2	Fa	$\forall$E 1
3	$\forall y Fy$	$\forall$I 2

The crucial thought here is that 'a' was just some *arbitrary* name. There was nothing special about it—we might have chosen any other name—and still the proof would be fine. And this crucial thought motivates the universal introduction rule (∀I). In this rule, we write $\forall\chi\phi(\chi/\mathfrak{c})$ to represent the result of replacing every instance of the name $\mathfrak{c}$ with the variable χ.

m	$\phi(\mathfrak{c})$	
	$\forall\chi\phi(\chi/\mathfrak{c})$	∀I m

$\mathfrak{c}$ must not occur in any undischarged assumption
χ must not occur in $\phi(\mathfrak{c})$

A crucial aspect of this rule, though, is bound up in the first constraint. This constraint ensures that we are always reasoning at a sufficiently general level.[1] To see the constraint in action, consider this terrible argument:

> Everyone loves Kylie Minogue; therefore everyone loves themselves.

We might symbolise this obviously invalid inference pattern as:

$$\forall xLxk \therefore \forall xLxx$$

Now, suppose we tried to offer a proof that vindicates this argument:

1	$\forall xLxk$	
2	Lkk	∀E 1
3	$\forall xLxx$	naughtily attempting to invoke ∀I 2

This is not allowed, because 'k' occurred already in an undischarged assumption, namely, on line 1. The crucial point is that, if we have made any assumptions about the object we are working with, then we are not reasoning generally enough to license ∀I.

Although the name may not occur in any *undischarged* assumption, it may occur as a discharged assumption. That is, it may occur in a subproof that we have already closed. For example:

1	Gd	
2	Gd	R 1
3	$Gd \to Gd$	→I 1–2
4	$\forall z(Gz \to Gz)$	∀I 3

This tells us that '$\forall z(Gz \to Gz)$' is a *theorem*. And that is as it should be.

[1] Recall from §30 that we are treating '⊥' as a canonical contradiction. But if it were the canonical contradiction as involving some *constant*, it might interfere with the constraint mentioned here. To avoid such problems, we shall treat '⊥' as a canonical contradiction *that involves no particular names*.

35.5 Existential elimination

Suppose we know that *something* is F. The problem is that simply knowing this does not tell us which thing is F. So it would seem that from '$\exists x Fx$' we cannot immediately conclude 'Fa', 'Fe_{23}', or any other substitution instance of the sentence. What can we do?

Suppose we know that something is F, and that everything which is F is G. In (almost) natural English, we might reason thus:

> Since something is F, there is some particular thing which is an F. We do not know anything about it, other than that it's an F, but for convenience, let's call it 'obbie'. So: obbie is F. Since everything which is F is G, it follows that obbie is G. But since obbie is G, it follows that something is G. And nothing depended on which object, exactly, obbie was. So, something is G.

We might try to capture this reasoning pattern in a proof as follows:

1	$\exists x Fx$	
2	$\forall x(Fx \to Gx)$	
3	$\quad Fo$	
4	$\quad Fo \to Go$	$\forall$E 2
5	$\quad Go$	$\to$E 4, 3
6	$\quad \exists x Gx$	$\exists$I 5
7	$\exists x Gx$	$\exists$E 1, 3–6

Breaking this down: we started by writing down our assumptions. At line 3, we made an additional assumption: 'Fo'. This was just a substitution instance of '$\exists x Fx$'. On this assumption, we established '$\exists x Gx$'. But note that we had made no *special* assumptions about the object named by 'o'; we had *only* assumed that it satisfies 'Fx'. So nothing depends upon which object it is. And line 1 told us that *something* satisfies 'Fx'. So our reasoning pattern was perfectly general. We can discharge the specific assumption 'Fo', and simply infer '$\exists x Gx$' on its own.

Putting this together, we obtain the existential elimination rule ($\exists$E):

m	$\exists \chi \phi$	
i	$\quad \phi(c/\chi)$	
j	$\quad \psi$	
	ψ	$\exists$E m, i–j

c must not occur in any assumption undischarged before line i
c must not occur in $\exists \chi \phi$
c must not occur in ψ

As with universal introduction, the constraints are extremely important. To see why, consider the following terrible argument:

> Jared Millson is a lecturer. There is someone who is not a lecturer.
> So Jared Millson is both a lecturer and not a lecturer.

We might symbolise this obviously invalid inference pattern as follows:

$$Lb, \exists x \neg Lx \therefore Lb \wedge \neg Lb$$

Now, suppose we tried to offer a proof that vindicates this argument:

1	Lb	
2	$\exists x \neg Lx$	
3	$\quad \neg Lb$	
4	$\quad Lb \wedge \neg Lb$	$\wedge$E 1, 3
5	$Lb \wedge \neg Lb$	illicit attempt to invoke $\exists$E 2, 3–4

The last line of the proof is not allowed. The name that we used in our substitution instance for '$\exists x \neg Lx$' on line 3, namely 'b', occurs in line 4. And the following proof would be no better:

1	Lb	
2	$\exists x \neg Lx$	
3	$\quad \neg Lb$	
4	$\quad Lb \wedge \neg Lb$	$\wedge$E 1, 3
5	$\quad \exists x(Lx \wedge \neg Lx)$	$\exists$I 4
6	$\exists x(Lx \wedge \neg Lx)$	illicit attempt to invoke $\exists$E 2, 3–5

The last line of the proof would still not be allowed. For the name that we used in our substitution instance for '$\exists x \neg Lx$', namely 'b', occurs in an undischarged assumption, namely line 1.

Opening a subproof for existential elimination is like entering a casino. In order to play the games you need to use the casino's currency, i.e. their 'chips.' You can only get chips by exchanging them for some actual money. The existentially quantified formula is like your money and the name you introduce at the start of your subproof is like the chips at the casino. When you exchange money for chips, you must make sure that the value of the chips you get is equivalent to the value of the money you hand over. Analogously, when you open a subproof you must make sure that the formula you suppose at the outset is just like the existentially quantified formula whose operator you want to eliminate, except that the variables are replaced by a name that does not occur in any undischarged assumptions (including premises). Starting a subproof with a name that occurs in an undischarged assumption is like going to the casino and exchanging your money for chips that you can't use in that casino! Once you've properly opened your subproof (i.e. exchanged your money

for chips and entered the casino), you use your supposition to reason as usual (i.e. play any of the games you want). When you're done playing, you must convert your chips back into money. Keeping with the analogy, this means that to close a subproof you must have arrived at a formula in which the original name does not occur. Just as you can't leave the casino with chips in your hand, so too, you can't close a subproof until you've derived a formula that doesn't contain the name with which you began it. The moral of the story is: Don't leave the casino with chips! Get your money before you leave! In other words, derive a formula that is free of the name with which you began the subproof.

Practice exercises

A. The following two 'proofs' are *incorrect*. Explain why both are incorrect. Also, provide interpretations which would invalidate the fallacious argument forms the 'proofs' enshrine:

1	$\forall x Rxx$	
2	Raa	$\forall$E 1
3	$\forall y Ray$	$\forall$I 2
4	$\forall x \forall y Rxy$	$\forall$I 3

1	$\forall x \exists y Rxy$	
2	$\exists y Ray$	$\forall$E 1
3	$\quad Raa$	
4	$\quad \exists x Rxx$	$\exists$I 3
5	$\exists x Rxx$	$\exists$E 2, 3–4

B. The following three proofs are missing their citations (rule and line numbers). Add them, to turn them into bona fide proofs.

1	$\forall x \exists y (Rxy \vee Ryx)$
2	$\forall x \neg Rmx$
3	$\exists y (Rmy \vee Rym)$
4	$\quad Rma \vee Ram$
5	$\quad \neg Rma$
6	$\quad Ram$
7	$\quad \exists x Rxm$
8	$\exists x Rxm$

1	$\forall x (\exists y Lxy \rightarrow \forall z Lzx)$
2	Lab
3	$\exists y Lay \rightarrow \forall z Lza$
4	$\exists y Lay$
5	$\forall z Lza$
6	Lca
7	$\exists y Lcy \rightarrow \forall z Lzc$
8	$\exists y Lcy$
9	$\forall z Lzc$
10	Lcc
11	$\forall x Lxx$

1		$\forall x(Jx \to Kx)$
2		$\exists x \forall y Lxy$
3		$\forall x Jx$
4		$\forall y Lay$
5		Laa
6		Ja
7		$Ja \to Ka$
8		Ka
9		$Ka \wedge Laa$
10		$\exists x(Kx \wedge Lxx)$
11		$\exists x(Kx \wedge Lxx)$

C. In §18 problem part A, we considered fifteen syllogistic figures of Aristotelian logic. Provide proofs for each of the argument forms. NB: You will find it *much* easier if you symbolise (for example) 'No F is G' as '$\forall x(Fx \to \neg Gx)$'.

D. Aristotle and his successors identified other syllogistic forms which depended upon 'existential import'. Symbolise each of the following argument forms in FOL and offer proofs.

- **Barbari.** Something is H. All G are F. All H are G. So: Some H is F
- **Celaront.** Something is H. No G are F. All H are G. So: Some H is not F
- **Cesaro.** Something is H. No F are G. All H are G. So: Some H is not F.
- **Camestros.** Something is H. All F are G. No H are G. So: Some H is not F.
- **Felapton.** Something is G. No G are F. All G are H. So: Some H is not F.
- **Darapti.** Something is G. All G are F. All G are H. So: Some H is F.
- **Calemos.** Something is H. All F are G. No G are H. So: Some H is not F.
- **Fesapo.** Something is G. No F is G. All G are H. So: Some H is not F.
- **Bamalip.** Something is F. All F are G. All G are H. So: Some H are F.

E. Provide a proof of each claim.

1. $\vdash \forall x Fx \vee \neg \forall x Fx$
2. $\vdash \forall z(Pz \vee \neg Pz)$
3. $\forall x(Ax \to Bx), \exists x Ax \vdash \exists x Bx$
4. $\forall x(Mx \leftrightarrow Nx), Ma \wedge \exists x Rxa \vdash \exists x Nx$
5. $\forall x \forall y Gxy \vdash \exists x Gxx$
6. $\vdash \forall x Rxx \to \exists x \exists y Rxy$
7. $\vdash \forall y \exists x(Qy \to Qx)$
8. $Na \to \forall x(Mx \leftrightarrow Ma), Ma, \neg Mb \vdash \neg Na$

9. $\forall x \forall y(Gxy \to Gyx) \vdash \forall x \forall y(Gxy \leftrightarrow Gyx)$
10. $\forall x(\neg Mx \vee Ljx), \forall x(Bx \to Ljx), \forall x(Mx \vee Bx) \vdash \forall x Ljx$

F. Write a symbolisation key for the following argument, symbolise it, and prove it:

> There is someone who likes everyone who likes everyone that she likes. Therefore, there is someone who likes herself.

G. For each of the following pairs of sentences: If they are provably equivalent, give proofs to show this. If they are not, construct an interpretation to show that they are not logically equivalent.

1. $\forall x Px \to Qc, \forall x(Px \to Qc)$
2. $\forall x \forall y \forall z Bxyz, \forall x Bxxx$
3. $\forall x \forall y Dxy, \forall y \forall x Dxy$
4. $\exists x \forall y Dxy, \forall y \exists x Dxy$
5. $\forall x(Rca \leftrightarrow Rxa), Rca \leftrightarrow \forall x Rxa$

H. For each of the following arguments: If it is valid in FOL, give a proof. If it is invalid, construct an interpretation to show that it is invalid.

1. $\exists y \forall x Rxy \therefore \forall x \exists y Rxy$
2. $\exists x(Px \wedge \neg Qx) \therefore \forall x(Px \to \neg Qx)$
3. $\forall x(Sx \to Ta), Sd \therefore Ta$
4. $\forall x(Ax \to Bx), \forall x(Bx \to Cx) \therefore \forall x(Ax \to Cx)$
5. $\exists x(Dx \vee Ex), \forall x(Dx \to Fx) \therefore \exists x(Dx \wedge Fx)$
6. $\forall x \forall y(Rxy \vee Ryx) \therefore Rjj$
7. $\exists x \exists y(Rxy \vee Ryx) \therefore Rjj$
8. $\forall x Px \to \forall x Qx, \exists x \neg Px \therefore \exists x \neg Qx$

Conversion of quantifiers 36

In this section, we shall add some additional rules to the basic rules of the previous section. These govern the interaction of quantifiers and negation.

In §17, we noted that $\neg\exists x\phi$ is logically equivalent to $\forall x\neg\phi$. We shall add some rules to our proof system that govern this. In particular, we add:

m	$\forall\chi\neg\phi$	
	$\neg\exists\chi\phi$	CQ m

and

m	$\neg\exists\chi\phi$	
	$\forall\chi\neg\phi$	CQ m

Equally, we add:

m	$\exists\chi\neg\phi$	
	$\neg\forall\chi\phi$	CQ m

and

m	$\neg\forall\chi\phi$	
	$\exists\chi\neg\phi$	CQ m

Practice exercises

A. Show that the following are jointly contrary:

1. $Sa \to Tm, Tm \to Sa, Tm \wedge \neg Sa$
2. $\neg\exists x Rxa, \forall x\forall y Ryx$
3. $\neg\exists x\exists y Lxy, Laa$
4. $\forall x(Px \to Qx), \forall z(Pz \to Rz), \forall y Py, \neg Qa \wedge \neg Rb$

B. Show that each pair of sentences is provably equivalent:

1. $\forall x(Ax \to \neg Bx), \neg\exists x(Ax \wedge Bx)$
2. $\forall x(\neg Ax \to Bd), \forall x Ax \vee Bd$

C. In §18, I considered what happens when we move quantifiers 'across' various logical operators. Show that each pair of sentences is provably equivalent:

1. $\forall x(Fx \wedge Ga), \forall x Fx \wedge Ga$
2. $\exists x(Fx \vee Ga), \exists x Fx \vee Ga$
3. $\forall x(Ga \to Fx), Ga \to \forall x Fx$
4. $\forall x(Fx \to Ga), \exists x Fx \to Ga$
5. $\exists x(Ga \to Fx), Ga \to \exists x Fx$
6. $\exists x(Fx \to Ga), \forall x Fx \to Ga$

NB: the variable 'x' does not occur in 'Ga'.

When all the quantifiers occur at the beginning of a sentence, that sentence is said to be in *prenex normal form*. These equivalences are sometimes called *prenexing rules*, since they give us a means for putting any sentence into prenex normal form.

Rules for identity 37

In §**??**, I mentioned the philosophically contentious thesis of the *identity of indiscernibles*. This is the claim that objects which are indiscernible in every way are, in fact, identical to each other. I also mentioned that we will not subscribe to this thesis. It follows that, no matter how much you tell me about two objects, I cannot prove that they are identical. Unless, of course, you tell me that the two objects are, in fact, identical. But then the proof will hardly be very illuminating.

The consequence of this, for our proof system, is that there are no sentences that do not already contain the identity predicate that could justify the conclusion '$a = b$'. This means that the identity introduction rule will not justify '$a = b$', or any other identity claim containing two different names.

However, every object is identical to itself. No premises, then, are required in order to conclude that something is identical to itself. So this will be the identity introduction rule:

	$\boldsymbol{c} = \boldsymbol{c}$	=I

Notice that this rule does not require referring to any prior lines of the proof. For any name $\boldsymbol{c}$, you can write $\boldsymbol{c} = \boldsymbol{c}$ on any point, with only the =I rule as justification.

Our elimination rule is more fun. If you have established '$a = b$', then anything that is true of the object named by 'a' must also be true of the object named by 'b'. For any sentence with 'a' in it, you can replace some or all of the occurrences of 'a' with 'b' and produce an equivalent sentence. For example, from 'Raa' and '$a = b$', you are justified in inferring 'Rab', 'Rba' or 'Rbb'. More generally:

m	$\boldsymbol{a} = \boldsymbol{b}$	
n	$\phi(\ldots \boldsymbol{a} \ldots \boldsymbol{a} \ldots)$	
	$\phi(\ldots \boldsymbol{b} \ldots \boldsymbol{a} \ldots)$	=E m, n

The notation here is as for ∃I. So $\phi(\ldots \boldsymbol{a} \ldots \boldsymbol{a} \ldots)$ is a formula containing the name $\boldsymbol{a}$, and $\phi(\ldots \boldsymbol{b} \ldots \boldsymbol{a} \ldots)$ is a formula obtained by replacing one or more instances of the name $\boldsymbol{a}$ with the name $\boldsymbol{b}$. Lines m and n can occur in either

order, and do not need to be adjacent, but we always cite the statement of identity first. Symmetrically, we allow:

m	$\boldsymbol{a} = \boldsymbol{b}$	
n	$\phi(\ldots \boldsymbol{b} \ldots \boldsymbol{b} \ldots)$	
	$\phi(\ldots \boldsymbol{a} \ldots \boldsymbol{b} \ldots)$	=E m, n

This rule is sometimes called *Leibniz's Law*, after Gottfried Leibniz.

To see the rules in action, we shall prove some quick results. First, we shall prove that identity is *symmetric*:

1	$a = b$	
2	$a = a$	=I
3	$b = a$	=E 1, 2
4	$a = b \to b = a$	→I 1–3
5	$\forall y(a = y \to y = a)$	∀I 4
6	$\forall x \forall y(x = y \to y = x)$	∀I 5

We obtain line 3 by replacing one instance of 'a' in line 2 with an instance of 'b'; this is justified given '$a = b$'.

Second, we shall prove that identity is *transitive*:

1	$a = b \wedge b = c$	
2	$a = b$	∧E 1
3	$b = c$	∧E 1
4	$a = c$	=E 2, 3
5	$(a = b \wedge b = c) \to a = c$	→I 1–4
6	$\forall z((a = b \wedge b = z) \to a = z)$	∀I 5
7	$\forall y \forall z((a = y \wedge y = z) \to a = z)$	∀I 6
8	$\forall x \forall y \forall z((x = y \wedge y = z) \to x = z)$	∀I 7

We obtain line 4 by replacing 'b' in line 3 with 'a'; this is justified given '$a = b$'.

Practice exercises

A. Provide a proof of each claim.

1. $Pa \vee Qb, Qb \to b = c, \neg Pa \vdash Qc$
2. $m = n \vee n = o, An \vdash Am \vee Ao$
3. $\forall x\ x = m, Rma \vdash \exists x Rxx$

4. $\forall x \forall y(Rxy \rightarrow x = y) \vdash Rab \rightarrow Rba$
5. $\neg\exists x \neg x = m \vdash \forall x \forall y(Px \rightarrow Py)$
6. $\exists x Jx, \exists x \neg Jx \vdash \exists x \exists y\ \neg x = y$
7. $\forall x(x = n \leftrightarrow Mx), \forall x(Ox \vee \neg Mx) \vdash On$
8. $\exists x Dx, \forall x(x = p \leftrightarrow Dx) \vdash Dp$
9. $\exists x\big[(Kx \wedge \forall y(Ky \rightarrow x = y)) \wedge Bx\big], Kd \vdash Bd$
10. $\vdash Pa \rightarrow \forall x(Px \vee \neg x = a)$

B. Show that the following are provably equivalent:

- $\exists x\big([Fx \wedge \forall y(Fy \rightarrow x = y)] \wedge x = n\big)$
- $Fn \wedge \forall y(Fy \rightarrow n = y)$

And hence that both have a decent claim to symbolise the English sentence 'Nick is the F'.

C. In §20, I claimed that the following are logically equivalent symbolisations of the English sentence 'there is exactly one F':

- $\exists x Fx \wedge \forall x \forall y\big[(Fx \wedge Fy) \rightarrow x = y\big]$
- $\exists x\big[Fx \wedge \forall y(Fy \rightarrow x = y)\big]$
- $\exists x \forall y(Fy \leftrightarrow x = y)$

Show that they are all provably equivalent. (*Hint*: to show that three claims are provably equivalent, it suffices to show that the first proves the second, the second proves the third and the third proves the first; think about why.)

D. Symbolise the following argument

> There is exactly one F. There is exactly one G. Nothing is both F and G. So: there are exactly two things that are either F or G.

And offer a proof of it.

Derived rules 38

As in the case of TFL, I first introduced some rules for FOL as basic (in §35), and then added some further rules for conversion of quantifiers (in §36). In fact, the CQ rules should be regarded as *derived* rules, for they can be derived from the *basic* rules of §35. (The point here is as in §34.) Here is a justification for the first CQ rule:

1	$\forall x \neg Ax$	
2	$\quad \exists x Ax$	
3	$\quad\quad Ac$	
4	$\quad\quad \neg Ac$	$\forall$E 1
5	$\quad\quad \bot$	$\bot$I 3, 4
6	$\quad \bot$	$\exists$E 2, 3–5
7	$\neg \exists x Ax$	$\neg$I 2–6

Here is a justification of the second CQ rule:

1	$\exists x \neg Ax$	
2	$\quad \forall x Ax$	
3	$\quad\quad \neg Ac$	
4	$\quad\quad Ac$	$\forall$E 2
5	$\quad\quad \bot$	$\bot$I 4, 3
6	$\quad \bot$	$\exists$E 1, 3–5
7	$\neg \forall x Ax$	$\neg$I 2–6

This explains why the CQ rules can be treated as derived. Similar justifications can be offered for the other two CQ rules.

Practice exercises

A. Offer proofs which justify the addition of the third and fourth CQ rules as derived rules.

Proof-theoretic concepts and semantic concepts 39

We have used two different turnstiles in this book. This:

$$\phi_1, \phi_2, \ldots, \phi_n \vdash \omega$$

means that there is some proof which starts with assumptions $\phi_1, \phi_2, \ldots, \phi_n$ and ends with ω (and no undischarged assumptions other than $\phi_1, \phi_2, \ldots, \phi_n$). This is a *proof-theoretic notion.*

By contrast, this:

$$\phi_1, \phi_2, \ldots, \phi_n \vDash \omega$$

means that there is no valuation (or interpretation) which makes all of $\phi_1, \phi_2, \ldots, \phi_n$ true and makes ω false. This concerns assignments of truth and falsity to sentences. It is a *semantic notion.*

I cannot emphasise enough that these are different notions. But I can emphasise it a bit more: *They are different notions.*

Once you have internalised this point, continue reading.

Although our semantic and proof-theoretic notions are different, there is a deep connection between them. To explain this connection, I shall start by considering the relationship between logical truths and theorems.

To show that a sentence is a theorem, you need only perform a proof. Granted, it may be hard to produce a twenty line proof, but it is not so hard to check each line of the proof and confirm that it is legitimate; and if each line of the proof individually is legitimate, then the whole proof is legitimate. Showing that a sentence is a logical truth, though, requires reasoning about all possible interpretations. Given a choice between showing that a sentence is a theorem and showing that it is a logical truth, it would be easier to show that it is a theorem.

Contrawise, to show that a sentence is *not* a theorem is hard. We would need to reason about all (possible) proofs. That is very difficult. But to show that a sentence is not a logical truth, you need only construct an interpretation in which the sentence is false. Granted, it may be hard to come up with the interpretation; but once you have done so, it is relatively straightforward to check what truth value it assigns to a sentence. Given a choice between showing that a sentence is not a theorem and showing that it is not a logical truth, it would be easier to show that it is not a logical truth.

Fortunately, *a sentence is a theorem if and only if it is a logical truth.* As a result, if we provide a proof of ϕ on no assumptions, and thus show that ϕ is a theorem, we can legitimately infer that ϕ is a logical truth; i.e., $\vDash \phi$. Similarly,

if we construct a model in which ϕ is false and thus show that it is not a logical truth, it follows that ϕ is not a theorem.

More generally, we have the following powerful result:

$$\phi_1, \phi_2, \ldots, \phi_n \vdash \psi \textbf{ iff } \phi_1, \phi_2, \ldots, \phi_n \vDash \psi$$

This shows that, whilst provability and entailment are *different* notions, they are extensionally equivalent. As such:

- An argument is *valid* iff *the conclusion can be proved from the premises.*
- Two sentences are *logically equivalent* iff they are *provably equivalent.*
- Sentences are *jointly consistent* iff they are *not jointly contrary.*

For this reason, you can pick and choose when to think in terms of proofs and when to think in terms of valuations/interpretations, doing whichever is easier for a given task. The table on the next page summarises which is (usually) easier.

It is intuitive that provability and semantic entailment should agree. But—let me repeat this—do not be fooled by the similarity of the symbols '$\vDash$' and '$\vdash$'. These two symbols have very different meanings. And the fact that provability and semantic entailment agree is not an easy result to come by.

In fact, demonstrating that provability and semantic entailment agree is, very decisively, the point at which introductory logic becomes intermediary logic. Agreement, in the case of TFL, is covered in the sequel to this book, `Metatheory`, which is the textbook for (part of) the second-year Logic paper. Agreement, in the case of FOL, is one of the first big results from the third-year Mathematical Logic paper.

	Yes	**No**
Is ϕ a **logical truth**?	give a proof which shows $\vdash \phi$	give an interpretation in which ϕ is false
Is ϕ a **contradiction**?	give a proof which shows $\vdash \neg\phi$	give an interpretation in which ϕ is true
Are ϕ and ψ **equivalent**?	give two proofs, one for $\phi \vdash \psi$ and one for $\psi \vdash \phi$	give an interpretation in which ϕ and ψ have different truth values
Are $\phi_1, \phi_2, \ldots, \phi_n$ **jointly consistent**?	give an interpretation in which all of $\phi_1, \phi_2, \ldots, \phi_n$ are true	prove a contradiction from assumptions $\phi_1, \phi_2, \ldots, \phi_n$
Is $\phi_1, \phi_2, \ldots, \phi_n \therefore \omega$ **valid**?	give a proof with assumptions $\phi_1, \phi_2, \ldots, \phi_n$ and concluding with ω	give an interpretation in which each of $\phi_1, \phi_2, \ldots, \phi_n$ is true and ω is false

Appendices

Symbolic notation A

A.1 Alternative nomenclature

Truth-functional logic. TFL goes by other names. Sometimes it is called *sentence logic*, because it deals fundamentally with sentences. Sometimes it is called *propositional logic*, on the idea that it deals fundamentally with propositions. I have stuck with *truth-functional logic*, to emphasise the fact that it deals only with assignments of truth and falsity to sentences, and that its connectives are all truth-functional.

First-order logic. FOL goes by other names. Sometimes it is called *predicate logic*, because it allows us to apply predicates to objects. Sometimes it is called *quantified logic*, because it makes use of quantifiers.

Formulas. Some texts call formulas *well-formed formulas*. Since 'well-formed formula' is such a long and cumbersome phrase, they then abbreviate this as *wff*. This is both barbarous and unnecessary (such texts do not countenance 'ill-formed formulas'). I have stuck with 'formula'.

In §6, I defined *sentences* of TFL. These are also sometimes called 'formulas' (or 'well-formed formulas') since in TFL, unlike FOL, there is no distinction between a formula and a sentence.

Valuations. Some texts call valuations *truth-assignments*.

Expressive adequacy. Some texts describe TFL as *truth-functionally complete*, rather than expressively adequate.

n-place predicates. I have called predicates 'one-place', 'two-place', 'three-place', etc. Other texts respectively call them 'monadic', 'dyadic', 'triadic', etc. Still other texts call them 'unary', 'binary', 'trinary', etc.

Names. In FOL, I have used 'a', 'b', 'c', for names. Some texts call these 'constants'. Other texts do not mark any difference between names and variables in the syntax. Those texts focus simply on whether the symbol occurs *bound* or *unbound*.

Domains. Some texts describe a domain as a 'domain of discourse', or a 'universe of discourse'.

A.2 Alternative symbols

In the history of formal logic, different symbols have been used at different times and by different authors. Often, authors were forced to use notation that their printers could typeset.

This appendix presents some common symbols, so that you can recognise them if you encounter them in an article or in another book.

Negation. Two commonly used symbols are the *hoe*, '$\neg$', and the *swung dash*, '$\sim$.' In some more advanced formal systems it is necessary to distinguish between two kinds of negation; the distinction is sometimes represented by using both '$\neg$' and '$\sim$'. Some texts use '$x \neq y$' to abbreviate '$\neg x = y$'.

Disjunction. The symbol '$\vee$' is typically used to symbolize inclusive disjunction. One etymology is from the Latin word 'vel', meaning 'or'.

Conjunction. Conjunction is often symbolized with the *ampersand*, '&'. The ampersand is a decorative form of the Latin word 'et', which means 'and'. (Its etymology still lingers in certain fonts, particularly in italic fonts; thus an italic ampersand might appear as '*&*'.) Using this symbol is not recommended, since it is commonly used in natural English writing (e.g. 'Smith & Sons'). As a symbol in a formal system, the ampersand is not the English word '&', so it is much neater to use a completely different symbol. The most common choice now is '$\wedge$', which is a counterpart to the symbol used for disjunction. Sometimes a single dot, '$\bullet$', is used. In some older texts, there is no symbol for conjunction at all; 'A and B' is simply written 'AB'.

Material Conditional. There are two common symbols for the material conditional: the *arrow*, '$\rightarrow$', and the *hook*, '$\supset$'.

Material Biconditional. The *double-headed arrow*, '$\leftrightarrow$', is used in systems that use the arrow to represent the material conditional. Systems that use the hook for the conditional typically use the *triple bar*, '$\equiv$', for the biconditional.

Quantifiers. The universal quantifier is typically symbolised as a rotated 'A', and the existential quantifier as a rotated, 'E'. In some texts, there is no separate symbol for the universal quantifier. Instead, the variable is just written in parentheses in front of the formula that it binds. For example, they might write '$(x)Px$' where we would write '$\forall xPx$'.

These alternative typographies are summarised below:

negation	$\neg$, $\sim$
conjunction	$\wedge$, &, $\bullet$
disjunction	$\vee$
conditional	$\rightarrow$, $\supset$
biconditional	$\leftrightarrow$, $\equiv$
universal quantifier	$\forall x$, (x)

Alternative proof systems B

In formulating my natural deduction system, I treated certain rules of natural deduction as *basic*, and others as *derived*. However, I could equally well have taken various different rules as basic or derived. I shall illustrate this point by considering some alternative treatments of disjunction, negation, and the quantifiers. I shall also explain why I have made the choices that I have.

B.1 Alternative disjunction elimination

Some systems take DS as their basic rule for disjunction elimination. Such systems can then treat the ∨E rule as a derived rule. For they might offer the following proof scheme:

m	$\phi \vee \psi$	
i	| ϕ	
j	| ω	
k	| ψ	
l	| ω	
n	$\phi \to \omega$	$\to$I i–j
$n+1$	$\psi \to \omega$	$\to$I k–l
$n+2$	| ω	
$n+3$	| ω	R $n+2$
$n+4$	| $\neg\omega$	
$n+5$	| | ϕ	
$n+6$	| | ω	$\to$E n, $n+5$
$n+7$	| | $\bot$	$\bot$I $n+6$, $n+4$
$n+8$	| $\neg\phi$	$\neg$I $n+5$–$n+7$
$n+9$	| ψ	DS m, $n+8$
$n+10$	| ω	$\to$E $n+1$, $n+9$
$n+11$	ω	TND $n+2$–$n+3$, $n+4$–$n+10$

So why did I choose to take ∨E as basic, rather than DS?[1] My reasoning is that DS involves the use of '¬' in the statement of the rule. It is in some sense 'cleaner' for our disjunction elimination rule to avoid mentioning *other* connectives.

B.2 Alternative negation rules

Some systems take the following rule as their basic negation introduction rule:

m	ϕ	
$n-1$	ψ	
n	$\neg\psi$	
	$\neg\phi$	$\neg$I* m–n

and the following as their basic negation elimination rule:

m	$\neg\phi$	
$n-1$	ψ	
n	$\neg\psi$	
	ϕ	$\neg$E* m–n

Using these two rules, we could have derived all of the rules governing negation and contradiction that we have taken as basic (i.e. ⊥I, ⊥E, ¬I and TND). Indeed, we could have avoided all use of the symbol '⊥' altogether. Negation would have had a single introduction and elimination rule, and would have behaved much more like the other connectives.

The resulting system would have had fewer rules than ours. So why did I chose to separate out contradiction, and to use an explicit rule TND?[2]

My first reason is that adding the symbol '⊥' to our natural deduction system makes proofs considerably easier to work with.

My second reason is that a lot of fascinating philosophical discussion has focussed on the acceptability or otherwise of *tertium non datur* (i.e. TND) and *ex falso quodlibet* (i.e. ⊥E). By treating these as separate rules in the proof system, we will be in a better position to engage with that philosophical discussion. In particular: having invoked these rules explicitly, it will be much easier for us to know what a system which lacked these rules would look like.

B.3 Alternative quantification rules

An alternative approach to the quantifiers is to take as basic the rules for ∀I and ∀E from §35, and also two CQ rule which allow us to move from $\forall\chi\neg\phi$ to $\neg\exists\chi\phi$ and vice versa.[3]

[1] P.D. Magnus's original version of this book went the other way.
[2] Again, P.D. Magnus's original version of this book went the other way.
[3] Warren Goldfarb follows this line in *Deductive Logic*, 2003, Hackett Publishing Co.

Taking only these rules as basic, we could have derived the ∃I and ∃E rules provided in §35. To derive the ∃I rule is fairly simple. Suppose ϕ contains the name c, and contains no instances of the variable χ, and that we want to do the following:

m	$\phi(\ldots c \ldots c \ldots)$
k	$\exists\chi\phi(\ldots \chi \ldots c \ldots)$

This is not yet permitted, since in this new system, we do not have the ∃I rule. We can, however, offer the following:

m	$\phi(\ldots c \ldots c \ldots)$	
$m+1$	$\quad\neg\exists\chi\phi(\ldots \chi \ldots c \ldots)$	
$m+2$	$\quad\forall\chi\neg\phi(\ldots \chi \ldots c \ldots)$	CQ $m+1$
$m+3$	$\quad\neg\phi(\ldots c \ldots c \ldots)$	∀E $m+2$
$m+4$	$\quad\bot$	⊥I m, $m+3$
$m+5$	$\neg\neg\exists\chi\phi(\ldots \chi \ldots c \ldots)$	¬I $m+1$–$m+4$
$m+6$	$\exists\chi\phi(\ldots \chi \ldots c \ldots)$	DNE $m+5$

To derive the ∃E rule is rather more subtle. This is because the ∃E rule has an important constraint (as, indeed, does the ∀I rule), and we need to make sure that we are respecting it. So, suppose we are in a situation where we *want* to do the following:

m	$\exists\chi\phi(\ldots \chi \ldots \chi \ldots)$
i	$\quad\phi(\ldots c \ldots c \ldots)$
j	$\quad\psi$
k	ψ

where c does not occur in any undischarged assumptions, or in ψ, or in $\exists\chi\phi(\ldots \chi \ldots \chi \ldots)$. Ordinarily, we would be allowed to use the ∃E rule; but we are not here assuming that we have access to this rule as a basic rule. Nevertheless, we could offer the following, more complicated derivation:

m	$\exists\chi\phi(\ldots\chi\ldots\chi\ldots)$	
i	$\quad\phi(\ldots c\ldots c\ldots)$	
j	$\quad\psi$	
k	$\phi(\ldots c\ldots c\ldots)\rightarrow\psi$	$\rightarrow$I i–j
$k+1$	$\quad\neg\psi$	
$k+2$	$\quad\neg\phi(\ldots c\ldots c\ldots)$	MT k, $k+1$
$k+3$	$\quad\forall\chi\neg\phi(\ldots\chi\ldots\chi\ldots)$	$\forall$I $k+2$
$k+4$	$\quad\neg\exists\chi\phi(\ldots\chi\ldots\chi\ldots)$	CQ $k+3$
$k+5$	$\quad\bot$	$\bot$I m, $k+4$
$k+6$	$\neg\neg\psi$	$\neg$I $k+1$–$k+5$
$k+7$	ψ	DNE $k+6$

We are permitted to use $\forall$I on line $k+3$ because c does not occur in any undischarged assumptions or in ψ. The entries on lines $k+4$ and $k+1$ contradict each other, because c does not occur in $\exists\chi\phi(\ldots\chi\ldots\chi\ldots)$.

Armed with these derived rules, we could now go on to derive the two remaining CQ rules, exactly as in §38.

So, why did I start with all of the quantifier rules as basic, and then derive the CQ rules?

My first reason is that it seems more intuitive to treat the quantifiers as on a par with one another, giving them their own basic rules for introduction and elimination.

My second reason relates to the discussion of alternative negation rules. In the derivations of the rules of $\exists$I and $\exists$E that I have offered in this section, I invoked DNE. This is a derived rule, whose derivation essentially depends upon the use of TND. But, as I mentioned earlier, TND is a contentious rule. So, if we want to move to a system which abandons TND, but which still allows us to use existential quantifiers, we shall want to take the introduction and elimination rules for the quantifiers as basic, and take the CQ rules as derived. (Indeed, in a system without TND, we shall be *unable* to derive the CQ rule which moves from $\neg\forall\chi\phi$ to $\exists\chi\neg\phi$.)

Quick reference C

C.1 Characteristic Truth Tables

ϕ	$\neg\phi$
T	F
F	T

ϕ	ψ	$\phi \wedge \psi$	$\phi \vee \psi$	$\phi \rightarrow \psi$	$\phi \leftrightarrow \psi$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

C.2 Symbolisation

Sentential Connectives

It is not the case that P	$\neg P$
Either P, or Q	$(P \vee Q)$
Neither P, nor Q	$\neg(P \vee Q)$ or $(\neg P \wedge \neg Q)$
Both P, and Q	$(P \wedge Q)$
If P, then Q	$(P \rightarrow Q)$
P only if Q	$(P \rightarrow Q)$
P if and only if Q	$(P \leftrightarrow Q)$
P unless Q	$(P \vee Q)$

Predicates

All Fs are Gs	$\forall x(Fx \rightarrow Gx)$
Some Fs are Gs	$\exists x(Fx \wedge Gx)$
Not all Fs are Gs	$\neg\forall x(Fx \rightarrow Gx)$ or $\exists x(Fx \wedge \neg Gx)$
No Fs are Gs	$\forall x(Fx \rightarrow \neg Gx)$ or $\neg\exists x(Fx \wedge Gx)$

Identity

Only c is G	$\forall x(Gx \leftrightarrow x = c)$
Everything besides c is G	$\forall x(\neg x = c \rightarrow Gx)$
The F is G	$\exists x(Fx \wedge \forall y(Fy \rightarrow x = y) \wedge Gx)$
It is not the case that the F is G	$\neg\exists x(Fx \wedge \forall y(Fy \rightarrow x = y) \wedge Gx)$
The F is non-G	$\exists x(Fx \wedge \forall y(Fy \rightarrow x = y) \wedge \neg Gx)$

Semantic Properties

ϕ is a TAUTOLOGY iff it is true on every valuation.

ϕ is a CONTRADICTION iff it is false on every valuation.

$\phi_1, \phi_2, \ldots, \phi_n$ are JOINTLY TAUTOLOGICALLY CONSISTENT iff there is some valuation which makes them all true.

$\phi_1, \phi_2, \ldots, \phi_n$ are JOINTLY TAUTOLOGICALLY INCONSISTENT iff there is no valuation which makes them all true.

ϕ and ψ are TAUTOLOGICALLY EQUIVALENT iff they have the same truth value on every valuation.

The sentences $\phi_1, \phi_2, \ldots, \phi_n$ TAUTOLOGICALLY ENTAIL the sentence ψ if there is no valuation of the atomic sentences which makes all of $\phi_1, \phi_2, \ldots, \phi_n$ true and ψ false.

$\phi_1, \ldots, \phi_n \vDash \psi$	'$\phi_1, \ldots, \phi_n$ entails ψ'
$\phi_1, \ldots, \phi_n \nvDash \psi$	'$\phi_1, \ldots, \phi_n$ does not entail ψ'
$\vDash \phi$	'ϕ is a tautology'
$\phi \vDash$	'ϕ is a contradiction'
$\phi_1, \ldots, \phi_n \vDash$	'$\phi_1, \ldots, \phi_n$ are inconsistent'
$\phi_1, \ldots, \phi_n \nvDash$	'$\phi_1, \ldots, \phi_n$ are consistent'
$\vDash \phi \leftrightarrow \psi$ or $\phi \Leftrightarrow \psi$	'ϕ and ψ are equivalent'

C.3 Definitions for Truth Trees

ROOT: the intitial sentence or sentences for which a tree is constructed.

BRANCH: the set of sentences obtained by starting at the bottom of the tree and reading upward through the tree along a series of lines.

LEAF: the sentence at the bottom of a branch.

LITERAL: an atomic sentence or its negation.

COMPLETED BRANCH: a branch all of whose complex sentences have been checked and whose leaf is either an atomic sentence or the negation of an atomic sentence, i.e. a literal.

COMPLETED TREE: a tree all of whose branches are completed.

CLOSED BRANCH: a branch on which an atomic sentence and its negation appear. The bottom of a closed branch contains a '×' below which is written the line numbers on which the contradictory literals appear.

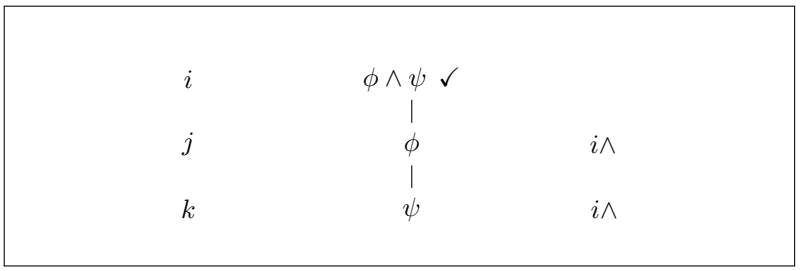
COMPLETED OPEN BRANCH: a completed branch that does not have a '×' below its leaf.

COMPLETED OPEN TREE: a completed tree that contains at least one open branch.

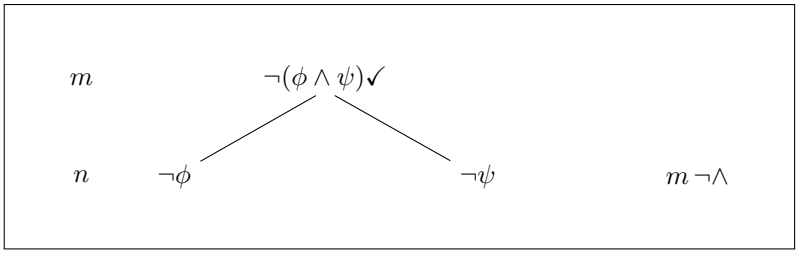
CLOSED TREE: a completed tree with no open branches, i.e. only closed branches.

C.4 Connective Rules for Truth Trees

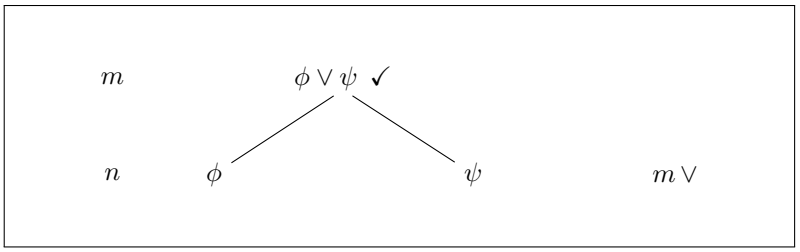
Conjunction



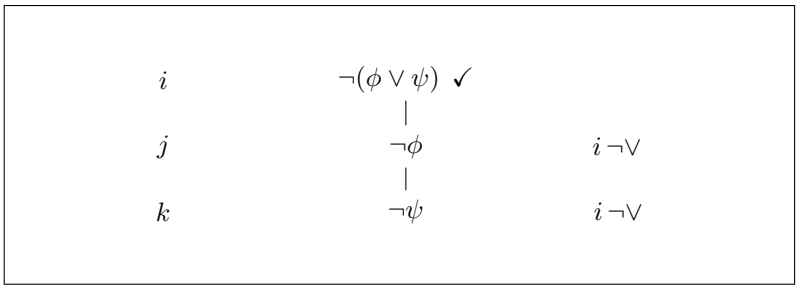
Negated Conjunction



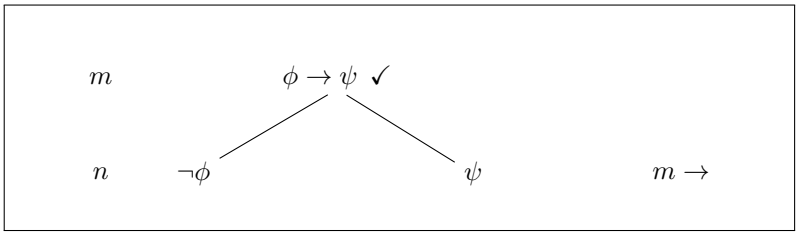
Disjunction



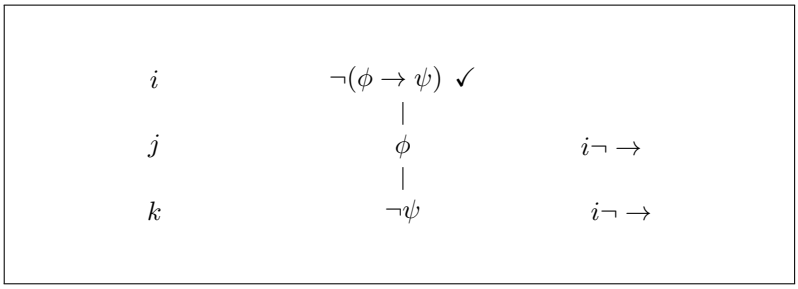
Negated Disjunction



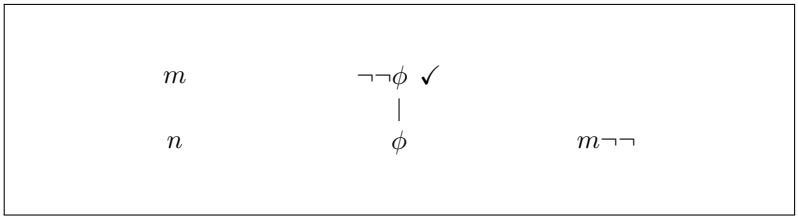
Conditional



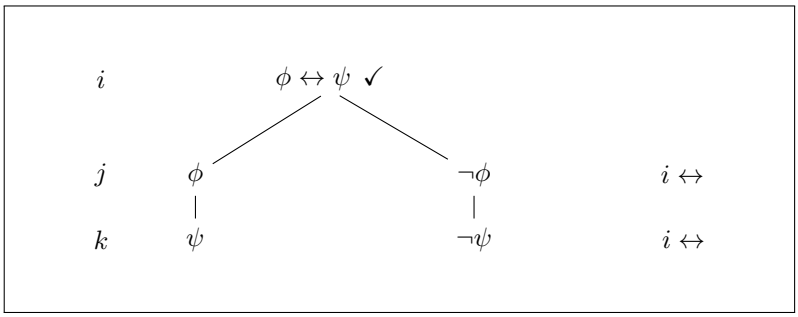
Negated Conditional



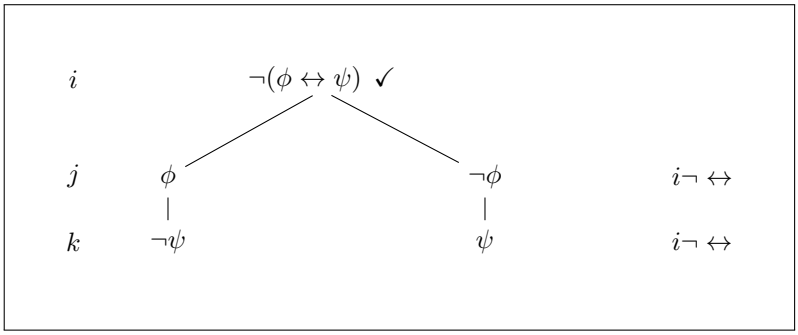
Double Negation



Biconditional



Negated Biconditional



Semantic Properties for Trees

ϕ is a TAUTOLOGY iff the truth tree whose ROOT is $\neg\phi$ is CLOSED.

ϕ is a CONTRADICTION iff the truth tree whose ROOT is ϕ is CLOSED.

$\phi_1, \phi_2, \ldots, \phi_n$ are CONSISTENT iff the truth tree whose ROOT is $\phi_1, \phi_2, \ldots, \phi_n$ is *not* CLOSED, i.e. it has at least one COMPLETED OPEN BRANCH.

$\phi_1, \phi_2, \ldots, \phi_n$ are INCONSISTENT iff the truth tree whose ROOT is $\phi_1, \phi_2, \ldots, \phi_n$ is CLOSED.

ϕ and ψ are TAUTOLOGICALLY EQUIVALENT iff the truth tree whose ROOT is $\neg(\phi \leftrightarrow \psi)$ is CLOSED.

$\phi_1, \phi_2, \ldots, \phi_n \models \psi$ iff the truth tree whose ROOT is $\phi_1, \phi_2, \ldots, \phi_n, \neg\psi$ is CLOSED.

C.5 Using identity to symbolize quantities

There are at least _____ Fs.

one: $\exists x Fx$
two: $\exists x_1 \exists x_2 (Fx_1 \wedge Fx_2 \wedge \neg x_1 = x_2)$
three: $\exists x_1 \exists x_2 \exists x_3 (Fx_1 \wedge Fx_2 \wedge Fx_3 \wedge \neg x_1 = x_2 \wedge \neg x_1 = x_3 \wedge \neg x_2 = x_3)$
four: $\exists x_1 \exists x_2 \exists x_3 \exists x_4 (Fx_1 \wedge Fx_2 \wedge Fx_3 \wedge Fx_4 \wedge \neg x_1 = x_2 \wedge \neg x_1 = x_3 \wedge \neg x_1 = x_4 \wedge \neg x_2 = x_3 \wedge \neg x_2 = x_4 \wedge \neg x_3 = x_4)$
n: $\exists x_1 \ldots \exists x_n (Fx_1 \wedge \ldots \wedge Fx_n \wedge \neg x_1 = x_2 \wedge \ldots \wedge \neg x_{n-1} = x_n)$

There are at most _____ Fs.

One way to say 'there are at most n Fs' is to put a negation sign in front of the symbolisation for 'there are at least $n+1$ Fs'. Equivalently, we can offer:

one: $\forall x_1 \forall x_2 \big[(Fx_1 \wedge Fx_2) \rightarrow x_1 = x_2\big]$
two: $\forall x_1 \forall x_2 \forall x_3 \big[(Fx_1 \wedge Fx_2 \wedge Fx_3) \rightarrow (x_1 = x_2 \vee x_1 = x_3 \vee x_2 = x_3)\big]$
three: $\forall x_1 \forall x_2 \forall x_3 \forall x_4 \big[(Fx_1 \wedge Fx_2 \wedge Fx_3 \wedge Fx_4) \rightarrow (x_1 = x_2 \vee x_1 = x_3 \vee x_1 = x_4 \vee x_2 = x_3 \vee x_2 = x_4 \vee x_3 = x_4)\big]$
n: $\forall x_1 \ldots \forall x_{n+1} \big[(Fx_1 \wedge \ldots \wedge Fx_{n+1}) \rightarrow (x_1 = x_2 \vee \ldots \vee x_n = x_{n+1})\big]$

There are exactly _____ Fs.

One way to say 'there are exactly n Fs' is to conjoin two of the symbolizations above and say 'there are at least n Fs and there are at most n Fs.' The following equivalent formulae are shorter:

zero: $\forall x \neg Fx$
one: $\exists x \big[Fx \wedge \forall y (Fy \rightarrow x = y)\big]$
two: $\exists x_1 \exists x_2 \big[Fx_1 \wedge Fx_2 \wedge \neg x_1 = x_2 \wedge \forall y \big(Fy \rightarrow (y = x_1 \vee y = x_2)\big)\big]$
three: $\exists x_1 \exists x_2 \exists x_3 \big[Fx_1 \wedge Fx_2 \wedge Fx_3 \wedge \neg x_1 = x_2 \wedge \neg x_1 = x_3 \wedge \neg x_2 = x_3 \wedge \forall y \big(Fy \rightarrow (y = x_1 \vee y = x_2 \vee y = x_3)\big)\big]$
n: $\exists x_1 \ldots \exists x_n \big[Fx_1 \wedge \ldots \wedge Fx_n \wedge \neg x_1 = x_2 \wedge \ldots \wedge \neg x_{n-1} = x_n \wedge \forall y \big(Fy \rightarrow (y = x_1 \vee \ldots \vee y = x_n)\big)\big]$

C.6 Basic deduction rules for TFL

Conjunction

$$\begin{array}{l|ll} m & \phi & \\ n & \psi & \\ & \phi \wedge \psi & \wedge\text{I } m, n \end{array}$$

$$\begin{array}{l|ll} m & \phi \wedge \psi & \\ & \phi & \wedge\text{E } m \end{array}$$

$$\begin{array}{l|ll} m & \phi \wedge \psi & \\ & \psi & \wedge\text{E } m \end{array}$$

Conditional

$$\begin{array}{l|ll} i & \left|\begin{array}{l}\phi\\\hline\psi\end{array}\right. & \\ j & & \\ & \phi \to \psi & \to\text{I } i\text{–}j \end{array}$$

$$\begin{array}{l|ll} m & \phi \to \psi & \\ n & \phi & \\ & \psi & \to\text{E } m, n \end{array}$$

Contradiction

$$\begin{array}{l|ll} m & \phi & \\ n & \neg\phi & \\ & \bot & \bot\text{I } m, n \end{array}$$

$$\begin{array}{l|ll} m & \bot & \\ & \phi & \bot\text{E } m \end{array}$$

Negation

$$\begin{array}{l|ll} i & \left|\begin{array}{l}\phi\\\hline\bot\end{array}\right. & \\ j & & \\ & \neg\phi & \neg\text{I } i\text{–}j \end{array}$$

Tertium non datur

$$\begin{array}{l|ll} i & \left|\begin{array}{l}\phi\\\hline\psi\end{array}\right. & \\ j & & \\ k & \left|\begin{array}{l}\neg\phi\\\hline\psi\end{array}\right. & \\ l & & \\ & \psi & \text{TND } i\text{–}j, k\text{–}l \end{array}$$

Disjunction

$$\begin{array}{l|ll} m & \phi & \\ & \phi \vee \psi & \vee\text{I } m \end{array}$$

$$\begin{array}{l|ll} m & \phi & \\ & \psi \vee \phi & \vee\text{I } m \end{array}$$

$$\begin{array}{l|ll} m & \phi \vee \psi & \\ i & \left|\begin{array}{l}\phi\\\hline\omega\end{array}\right. & \\ j & & \\ k & \left|\begin{array}{l}\psi\\\hline\omega\end{array}\right. & \\ l & & \\ & \omega & \vee\text{E } m, i\text{–}j, k\text{–}l \end{array}$$

Biconditional

$$\begin{array}{l|ll} i & \left|\begin{array}{l}\phi\\\hline\psi\end{array}\right. & \\ j & & \\ k & \left|\begin{array}{l}\psi\\\hline\phi\end{array}\right. & \\ l & & \\ & \phi \leftrightarrow \psi & \leftrightarrow\text{I } i\text{–}j, k\text{–}l \end{array}$$

$$\begin{array}{l|ll} m & \phi \leftrightarrow \psi & \\ n & \phi & \\ & \psi & \leftrightarrow\text{E } m, n \end{array}$$

$$\begin{array}{l|ll} m & \phi \leftrightarrow \psi & \\ n & \psi & \\ & \phi & \leftrightarrow\text{E } m, n \end{array}$$

C.7 Derived rules for TFL

Disjunctive syllogism

m	$\phi \vee \psi$	
n	$\neg\phi$	
	ψ	DS m, n

m	$\phi \vee \psi$	
n	$\neg\psi$	
	ϕ	DS m, n

Reiteration

m	ϕ	
	ϕ	R m

Modus Tollens

m	$\phi \to \psi$	
n	$\neg\psi$	
	$\neg\phi$	MT m, n

Double-negation elimination

m	$\neg\neg\phi$	
	ϕ	DNE m

De Morgan Rules

m	$\neg(\phi \vee \psi)$	
	$\neg\phi \wedge \neg\psi$	DeM m

m	$\neg\phi \wedge \neg\psi$	
	$\neg(\phi \vee \psi)$	DeM m

m	$\neg(\phi \wedge \psi)$	
	$\neg\phi \vee \neg\psi$	DeM m

m	$\neg\phi \vee \neg\psi$	
	$\neg(\phi \wedge \psi)$	DeM m

C.8 Basic deduction rules for FOL

Universal elimination

$$\begin{array}{l|ll} m & \forall \chi \phi & \\ & \phi(c/\chi) & \forall\text{E } m \end{array}$$

Universal introduction

$$\begin{array}{l|ll} m & \phi(c) & \\ & \forall \chi \phi(\chi/c) & \forall\text{I } m \end{array}$$

c must not occur in any undischarged assumption
χ must not occur in $\phi(c)$

Identity introduction

$$\begin{array}{l|ll} & c = c & =\text{I} \end{array}$$

Identity elimination

$$\begin{array}{l|ll} m & a = b & \\ n & \phi(\ldots a \ldots a \ldots) & \\ & \phi(\ldots b \ldots a \ldots) & =\text{E } m, n \end{array} \qquad \begin{array}{l|ll} m & a = b & \\ n & \phi(\ldots b \ldots b \ldots) & \\ & \phi(\ldots a \ldots b \ldots) & =\text{E } m, n \end{array}$$

Existential introduction

$$\begin{array}{l|ll} m & \phi(c) & \\ & \exists \chi \phi(\ldots \chi \ldots c \ldots) & \exists\text{I } m \end{array}$$

χ must not occur in $\phi(c)$ You can replace one or more instance of c with χ.

Existential elimination

$$\begin{array}{l|ll} m & \exists \chi \phi & \\ i & \quad \phi(c/\chi) & \\ j & \quad \psi & \\ & \psi & \exists\text{E } m, i\text{–}j \end{array}$$

c must not occur in any undischarged assumption, in $\exists \chi \phi$, or in ψ

C.9 Derived rules for FOL

$$\begin{array}{l|ll} m & \forall \chi \neg \phi & \\ & \neg \exists \chi \phi & \text{CQ } m \end{array} \qquad \begin{array}{l|ll} m & \exists \chi \neg \phi & \\ & \neg \forall \chi \phi & \text{CQ } m \end{array}$$

$$\begin{array}{l|ll} m & \neg \exists \chi \phi & \\ & \forall \chi \neg \phi & \text{CQ } m \end{array} \qquad \begin{array}{l|ll} m & \neg \forall \chi \phi & \\ & \exists \chi \neg \phi & \text{CQ } m \end{array}$$