PROOF THEORY FOR PHILOSOPHY

NOTES FOR THE SELLC 2010 COURSE
Structures for Proofs

Greg Restall
Philosophy Department
University of Melbourne
restall@unimelb.edu.au
http://consequently.org/writing/ptp

VERSION OF NOVEMBER 30, 2010
© GREG RESTALL
INTRODUCTION

This manuscript is a draft of a guided introduction to logic and its applications in philosophy. The focus will be a detailed examination of the different ways to understand proof. Along the way, we will also take a few glances around to the other side of inference, the kinds of counterexamples to be found when an inference fails to be valid.

The book is designed to serve a number of different purposes, and it can be used in a number of different ways. In writing the book I have at least these four aims in mind.

A GENTLE INTRODUCTION TO KEY IDEAS IN THE THEORY OF PROOF: There are a number of very good introductions to proof theory: Bostock’s Intermediate Logic [11], Tennant’s Natural Logic [81], Troelstra and Schwichtenberg’s Basic Proof Theory [83], and von Plato and Negri’s Structural Proof Theory [55] are all excellent books, with their own virtues. However, they all introduce the core ideas of proof theory in what can only be described as a rather complicated fashion. The core technical results of proof theory (normalisation for natural deduction and cut elimination for sequent systems) are relatively simple ideas at their heart, but the expositions of these ideas in the available literature are quite difficult and detailed. This is through no fault of the existing literature. It is due to a choice. In each book a proof system for the whole of classical or intuitionistic logic is introduced, and then formal properties are demonstrated about such a system. Each proof system has different rules for each of the connectives, and this makes the proof-theoretical results such as normalisation and cut elimination case-ridden and lengthy. (The standard techniques are complicated inductions with different cases for each connective: the more connectives and rules, the more cases.)

In this book, the exposition will be somewhat different. Instead of taking a proof system as given and proving results about it, we will first look at the core ideas (normalisation for natural deduction, and cut elimination for sequent systems) and work with them in their simplest and purest manifestation. In Section 2.3 we will see a two-page normalisation proof. In Section 3.2 we will see a two-page cut-elimination proof. In each case, the aim is to understand the key concepts behind the central results.

AN INTRODUCTION TO LOGIC FROM A NON-PARTISAN, PLURALIST, PROOF-THEORETIC PERSPECTIVE: We are able to take this liberal approach to introducing proof theory because we take a pluralist attitude to the choice of logical system. This book is designed to be an introduction to logic that does not have a distinctive axe to grind in favour of a
particular logical system. Instead of attempting to justify this or that formal system, we will give an overview of the panoply of different accounts of consequence for which a theory of proof has something interesting and important to say. As a result, in Chapter 4 we will examine the behaviour of conditionals from intuitionistic, relevant and linear logic. The system of natural deduction we will start off with is well suited to them. In Chapter 4, we also look at a sequent system for the non-distributive logic of conjunction and disjunction, because this results in a very simple cut elimination proof. From there, we go on to show how to relate sequent systems and natural deduction proofs, ending with a comprehensive view of a natural deduction system for intuitionistic logic. Once there, we go on to consider classical logic, where we see that proofs are more naturally viewed as circuits instead of trees. This view of the structure of proofs is a much less prevalent in the literature, and as a result, it is less likely to be familiar to you, the reader. So, these techniques are given a chapter of their own, Chapter ??.

From there, we go on to add richer structures, always with an eye to isolate the core techniques and distinctive innovations appropriate to each stage of our investigation, from quantifiers, to identity, to modality and truth.

An introduction to the applications of proof theory: We will always have our eye on the concerns others have concerning proof theory. What are the connections between proof theories and theories of meaning? What does an account of proof tell us about how we might apply the formal work of logical theorising? All accounts of meaning have something to say about the role of inference. For some, it is what things mean that tells you what inferences are appropriate. For others, it is what inferences are appropriate that constitutes what things mean. For everyone, there is an intimate connection between inference and semantics.

A presentation of new results: Recent work in proofnets and other techniques in non-classical logics like linear logic can usefully illuminate the theory of much more traditional logical systems, like classical logic itself. I aim to present these results in an accessible form, and extend them to show how you can give a coherent picture of classical and non-classical propositional logics, quantifiers and modal operators.

The book is filled with marginal notes which expand on and comment on the central text. Feel free to read or ignore them as you wish, and to add your own comments. Each chapter (other than this one) contains definitions, examples, theorems, lemmas, and proofs. Each of these (other than the proofs) are numbered consecutively, first with the chapter number, and then with the number of the item within the chapter. Proofs end with a little box at the right margin, like this:

The manuscript is divided into three parts, each of which is divided into chapters. The parts cover different aspects of logical vocabulary. The first part, Propositional Logic, covers propositions and the way
they are combined—the focus there is on the core propositional connectives, the conditional, conjunction, disjunction, and negation; in the second part, Quantifiers, Identity and Existence—our attention shifts to the structure of propositions themselves, and ‘splits the atom’ to consider predication and quantification; the third and final part, modality and truth covers these two important topics using the tools developed throughout the book.

In each of the three parts, the initial chapters cover logical tools and techniques suited to the topic under examination. Then the later chapters in each part take up the issues that are raised by those tools and techniques, and applies them to different issues in philosophy of language, metaphysics, epistemology, philosophy of mathematics and much else, besides.

Some chapters contain exercises to complete. Logic is never learned without hard work, so if you want to learn the material, work through the exercises: especially the basic, intermediate and advanced ones. The project questions are examples of current research topics.

The book has an accompanying website: http://consequently.org/writing/ptp. From here you can look for an updated version of the book, leave comments, read the comments others have left, check for solutions to exercises and supply your own. Please visit the website and give your feedback. Visitors to the website have already helped me make this volume much better than it would have been were it written in isolation. It is a delight to work on logic within such a community, spread near and far.

ACKNOWLEDGEMENTS: [To be added.]

Greg Restall

Melbourne

NOVEMBER 30, 2010
WHY PROOF THEORY?

Why? My first and overriding reason to be interested in proof theory is the beauty and simplicity of the subject. It is one of the central strands of the discipline of logic, along with its partner, model theory. Since the flowering of the field with the work of Gentzen, many beautiful definitions, techniques and results are to be found in this field, and they deserve a wider audience. In this book I aim to provide an introduction to proof theory that allows the reader with only a minimal background in logic to start with the flavour of the central results, and then understand techniques in their own right.

It is one thing to be interested in proof theory in its own right, or as a part of a broader interest in logic. It’s another thing entirely to think that proof theory has a role in philosophy. Why would a philosopher be interested in the theory of proofs? Here are just three examples of concerns in philosophy where proof theory finds a place.

**Example 1: Meaning.** Suppose you want to know when someone is using “or” in the same sense that you do. When does “or” in their vocabulary have the same significance as “or” in yours? One answer could be given in terms of truth-conditions. The significance of “or” can be given as follows:

\[ \neg p \lor \neg q \text{ is true if and only if } \neg p \text{ is true or } \neg q \text{ is true.} \]

Perhaps you have seen this information presented in a truth-table.

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>p or q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Clearly, this table can be used to distinguish between some uses of disjunctive vocabulary from others. We can use it to rule out exclusive disjunction. If we take \( \neg p \lor \neg q \) to be false when we take \( \neg p \) and \( \neg q \) to be both true, then we are using “or” in a manner that is at odds with the truth table.

However, what can we say of someone who is ignorant of the truth or falsity of \( \neg p \) and of \( \neg q \)? What does the truth table tell us about \( \neg p \lor \neg q \) in that case? It seems that the application of the truth table to our practice is less-than-straightforward.

It is for reasons like this that people have considered an alternate explanation of a logical connective such as “or.” Perhaps we can say that
someone is using “or” in the way that you do if you are disposed to make the following deductions to reason to a disjunction

\[
\frac{p}{p \lor q} \quad \frac{q}{p \lor q}
\]

and to reason from a disjunction

\[
\frac{[p]}{p \lor q} \quad \frac{[q]}{r}
\]

\[
\frac{r}{r}
\]

That is, you are prepared to infer to a disjunction on the basis of either disjunct; and you are prepared to reason by cases from a disjunction. Is there any more you need to do to fix the use of “or”? That is, if you and I both use “or” in a manner consonant with these rules, then is there any way that our usages can differ with respect to meaning?

Clearly, this is not the end of the story. Any proponent of a proof-first explanation of the meaning of a word such as “or” will need to say something about what it is to accept an inference rule, and what sorts of inference rules suffice to define a concept such as disjunction (or negation, or universal quantification, and so on). When does a definition work? What are the sorts of things that can be defined using inference rules? What are the sorts of rules that may be used to define these concepts? We will consider these issues in Chapter ??.

**Example 2: Generality.** It is a commonplace that it is impossible or very difficult to prove a nonexistence claim. After all, if there is no object with property F, then every object fails to have property F. How can we demonstrate that every object in the entire universe has some property? Surely we cannot survey each object in the universe one-by-one. Furthermore, even if we come to believe that object a has property F for each object a that happens to exist, it does not follow that we ought to believe that every object has that property. The universal judgement tells us more than the truth of each particular instance of that judgement, for given all of the objects \(a_1, a_2, \ldots\), it certainly seems possible that \(a_1\) has property F, that \(a_2\) has property F and so on, without everything having property F since it seems possible that there might be some new object which does not actually exist. If you care to talk of ‘facts’ then we can express the matter by saying that the fact that everything is F cannot amount to just the fact that \(a_1\) is F and the fact that \(a_2\) is F, etc., it must also include the fact that \(a_1, a_2, \ldots\) are all of the objects. There seems to be some irreducible universality in universal judgements.

If this was all that we could say about universality, then it would seem to be very difficult to come to universal conclusions. However, we manage to derive universal conclusions regularly. Consider mathematics, it is not difficult to prove that every whole number is either even
or odd. We can do this without examining every number individually. Just how do we do this?

It is a fact that we do accomplish this, for we are able to come to universal conclusions as a matter of course. In the course of this book we will see how such a thing is possible. Our facility at reasoning with quantifiers, such as ‘for every’ and ‘for some,’ is intimately tied up with the structures of the claims we can make, and how the formation of judgements from names and predicates gives us a foothold which may be exploited in reasoning. When we understand the nature of proofs involving quantifiers, this will give us insight into how we can gain general information about our world.

**Example 3: Modality.** A third example is similar. Philosophical discussion is full of talk of possibility and necessity. What is the significance of this talk? What is its logical structure? One way to give an account of the logical structure of possibility and necessity talk is to analyse it in terms of possible worlds. To say that it is possible that Australia win the World Cup is to say that there is some possible world in which Australia wins the World Cup. Talk of possible worlds helps clarify the logical structure of possibility and necessity. It is possible that either Australia or New Zealand win the World Cup only if there’s a possible world in which either Australia or New Zealand win the World Cup. In other words, either there’s a possible world in which Australia wins, or a possible world in which New Zealand wins, and hence, it is either possible that Australia wins the World Cup or that New Zealand wins. We have reasoned from the possibility of a disjunction to the disjunction of the corresponding possibilities. Such an inference seems correct. Is talk of possible worlds required to explain this kind of step, or is there some other account of the logical structure of possibility and necessity? I will argue in this book that when we attend to the structure of proofs involving modal notions, we will see how this use helps determine the concepts of necessity and possibility, and this thereby gives us an understanding the notion of a possible world. We don’t first understand modal concepts by invoking possible worlds—we can invoke possible worlds when we first understand modal concepts, and the logic of modal concepts can be best understood when we understand what modal reasoning is for and how we do it.

**Example 4: A New Angle on Old Ideas** Lastly, one reason for studying proof theory is the perspective it brings on familiar themes. There is a venerable and well-trodden road between truth, models and logical consequence. Truth is well-understood, models (truth tables for propositional logic, or Tarski’s models for first-order predicate logic, Kripke models for modal logic, or whatever else) are taken to be models of truth, and logical consequence is understood as the preservation of truth in all models. Nothing in this book will count against the road from truth to logical consequence. However, we will travel that road in the other direction. By starting with logical consequence—and arriving at

It is one thing to know that $2 + 3 = 3 + 2$. It is quite another to conclude that for every pair of natural numbers $n$ and $m$ that $n + m = m + n$. Yet we do this sort of thing quite regularly.

However, the fact that the notion of truth is beset by paradox should be a warning sign that using it to define core features of logic may not provide the most stable foundation.
that point by way of an analysis of proofs—we will retrace those steps in reverse, to construct models from a prior understanding of truth, and then with an approach to truth once we have a notion of a model in hand. At the very least, we will see old ideas from a new angle. Perhaps, when we see matters from this new perspective, the insights will be of more lasting value.

These are four examples of the kinds of issues that we will consider in the light of proof theory in the pages ahead. Before we can broach these topics, we need to learn some proof theory. In Part I we will focus on propositional logic. We will start this part with a chapter on proofs for conditional judgements.
NATURAL DEDUCTION
FOR CONDITIONALS

We start with modest ambitions. In this section we focus on one way of understanding proof—natural deduction, in the style of Gentzen [28]—and we will consider just one kind of judgement: conditionals.

2.1 | THE LANGUAGE

Conditionals take the form

If . . . then . . .

To make things precise, we will use a formal language in which we can form conditional judgements. Our language will have an unlimited supply of atomic formulas

\[ p, q, r, p_0, p_1, p_2, \ldots \quad q_0, q_1, q_2, \ldots \quad r_0, r_1, r_2, \ldots \]

When we need to refer to the collection of all atomic formulas, we will call it atom. Whenever we have two formulas A and B, whether A and B are in atom or not, we will say that \((A \rightarrow B)\) is also a formula. Succinctly, this grammar can be represented as follows:

\[
\text{FORMULA} ::= \text{ATOM} \mid (\text{FORMULA} \rightarrow \text{FORMULA})
\]

That is, a formula is either an \text{ATOM}, or is found by placing an arrow (written like this ‘→’) between two formulas, and surrounding the result with parentheses. So, these are formulas

\[
p_3 \quad (q \rightarrow r) \quad ((p_1 \rightarrow (q_1 \rightarrow r_1)) \rightarrow (q_1 \rightarrow (p_1 \rightarrow r_1))) \quad (p \rightarrow (q \rightarrow (r \rightarrow (p_1 \rightarrow (q_1 \rightarrow r_1))))))
\]

but these are not:

\[
t \quad p \rightarrow q \rightarrow r \quad p \rightarrow p
\]

The first, t, fails to be a formula since it is not in our set \text{ATOM} of atomic formulas (so it doesn’t enter the collection of formulas by way of being an atom) and it does not contain an arrow (so it doesn’t enter the collection through the clause for complex formulas). The second, \(p \rightarrow q \rightarrow r\) does not enter the collection because it is short of a few parentheses. The only expressions that enter our language are those that bring a pair of parentheses along with every arrow: “\(p \rightarrow q \rightarrow r\)” has two arrows but no parentheses, so it does not qualify. You can see why it should be excluded because the expression is ambiguous. Does it express the conditional judgement to the effect that if \(p\) then \(q\) then \(r\), or is it the

Gerhard Gentzen, German Logician: Born 1909, student of David Hilbert at Göttingen, died in 1945 in World War II. http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Gentzen.html

This is BNF, or “Backus Naur Form,” first used in the specification of formal computer programming languages such as Algol. http://cui.unige.ch/db-research/Enseignement/analyseinfo/AboutBNF.html

You can do without parentheses if you use ‘prefix’ notation for the conditional: ‘\(Cpq\)’ instead of ‘\(p \rightarrow q\)’. The conditional are then \(CpCqr\) and \(CCpqr\). This is Polish notation.
judgement that if it’s true that if \( p \) then \( q \), then it’s also true that \( r \)? In other words, it is ambiguous between these two formulas:

\[
(p \rightarrow (q \rightarrow r)) \quad (p \rightarrow q) \rightarrow r)
\]

Our last example of an offending formula—\( p \rightarrow p \)—does not offend nearly so much. It is not ambiguous. It merely offends against the letter of the law laid down, and not its spirit. I will feel free to use expressions such as “\( p \rightarrow p \)” or “\((p \rightarrow q) \rightarrow (q \rightarrow r)\)” which are missing their outer parentheses, even though they are, strictly speaking, not in FORMULA.

Given a formula containing at least one arrow, such as \( (p \rightarrow q) \rightarrow (q \rightarrow r) \), it is important to be able to isolate its main connective (the last arrow introduced as it was constructed). In this case, it is the middle arrow. The formula to the left of the arrow (in this case \( p \rightarrow q \)) is said to be the antecedent of the conditional, and the formula to the right is the consequent (here, \( q \rightarrow r \)).

We can think of these formulas in at least two different ways. We can think of them as the sentences in a toy language. This language is either something completely separate from our natural languages, or it is a fragment of a natural language, consisting only of atomic expressions and the expressions you can construct using a conditional construction like “if . . . then . . . .” On the other hand, you can think of formulas as not constituting a language themselves, but as constructions used to display the form of expressions in a language. Nothing here will stand on which way you understand formulas.

Sometimes, we will want to talk quite generally about all formulas of a particular form. We will want to do this very often, when it comes to logic, because we are interested in the structures or forms of valid arguments. The structural or formal features of arguments apply generally, to more than just a particular argument. (If we know that an argument is valid in virtue of its possessing some particular form, then other arguments with that form are valid as well.) So, these formal or structural principles must apply generally. Our formal language goes some way to help us express this, but it will turn out that we will not want to talk merely about specific formulas in our language, such as \( (p_3 \rightarrow q_7) \rightarrow r_{26} \). We will, instead, want to say things like

**Given a conditional formula, and its antecedent, its consequent follows.**

This can get very complicated very quickly. It is not at all convenient to say

**Given a conditional formula whose consequent is also a conditional, the conditional formula whose antecedent is the antecedent of the consequent of the original conditional, and whose consequent is a conditional whose antecedent is the antecedent of the original conditional and whose consequent is the consequent of the conditional inside the first conditional follows from the original conditional.**
Instead of that mouthful, we will use *variables* to talk generally about formulas in much the same way that mathematicians use variables to talk generally about numbers and other such things. We will use capital letters like

\[ A, B, C, D, \ldots \]

as variables ranging over the class formula. So, instead of the long paragraph above, it suffices to say

*From \( A \to (B \to C) \) you can infer \( B \to (A \to C) \).*

which seems much more perspicuous and memorable. Now we have the raw formal materials to address the question of deduction using conditional judgements. How may we characterise valid reasoning using conditional constructions? We will look at one way of addressing this topic in this section.

### 2.2 PROOFS FOR CONDITIONALS

Start with a piece of reasoning using conditional judgements. One example might be this:

*Suppose \( A \to (B \to C) \). Suppose \( A \). It follows that \( B \to C \).*  
*Suppose \( B \). It follows that \( C \).*

This kind of reasoning has two important features. We make *suppositions or assumptions*. We also infer *from* these assumptions. From \( A \to (B \to C) \) and \( A \) we inferred \( B \to C \). From this new information, together with the supposition that \( B \), we inferred a new conclusion, \( C \).

One way to represent the structure of this piece of reasoning is in this *tree diagram* shown here.

\[
\frac{A \to (B \to C) \quad A}{\frac{B \to C \quad B}{C}}
\]

The *leaves* of the tree are the formulas \( A \to (B \to C) \), \( A \) and \( B \). They are the assumptions upon which the deduction rests. The other formulas in the tree are *deduced* from formulas occurring above them in the tree. The formula \( B \to C \) is written immediately below a line, above which are the formulas from which we deduced it. So, \( B \to C \) follows from the leaves \( A \to (B \to C) \) and \( A \). Then the *root* of the tree (the formula at the bottom), \( C \), follows from that formula \( B \to C \) and the other leaf \( B \). The ordering of the formulas bears witness to the relationships of inference between those formulas in our process of reasoning.

The two steps in our example proof use the same kind of reasoning. The inference from a conditional, and from its antecedent to its consequent. This step is called *modus ponens*. It’s easy to see that using *modus ponens* we always move from more complicated formulas to less complicated formulas. However, sometimes we wish to infer

*"Modus ponens" is short for "modus ponendo ponens," which means "the mode of affirming by affirming." You get to the affirmation of \( B \) by way of the affirmation of \( A \) (and the other premise, \( A \to B \)). It may be contrasted with *Modus tollendo tollens*, the mode of denying by denying: from \( A \to B \) and not \( B \) to not \( A \).*
the conditional \( A \rightarrow B \) on the basis of our information about \( A \) and about \( B \). And it seems that sometimes this is legitimate. Suppose we want to know about the connection between \( A \) and \( C \) in a context in which we are happy to assume both \( A \rightarrow (B \rightarrow C) \) and \( B \). What kind of connection is there (if any) between \( A \) and \( C \)? It would seem that it \( \text{would} \) be appropriate to infer \( A \rightarrow C \), since we have a valid argument to the conclusion that \( C \) if we make the assumption that \( A \).

\[
A \rightarrow (B \rightarrow C) \quad [A]^{(1)}
\]

\[
\begin{align*}
B \rightarrow C & \quad B \\
C & \quad [1] \\
A \rightarrow C
\end{align*}
\]

So, it seems we can reason like this. At the step marked with \([1]\), we make the inference to the conditional conclusion, on the basis of the reasoning up until that point. Since we can infer to \( C \) using \( A \) as an assumption, we can conclude \( A \rightarrow C \). At this stage of the reasoning, \( A \) is no longer active as an assumption: we discharge it. It is still a leaf of the tree (there is no node of the tree above it), but it is no longer an active assumption in our reasoning. So, we bracket it, and annotate the brackets with a label, indicating the point in the demonstration at which the assumption is discharged. Our proof now has two assumptions, \( A \rightarrow (B \rightarrow C) \) and \( B \), and one conclusion, \( A \rightarrow C \).

\[
A \rightarrow B \quad A \rightarrow E
\]

\[
\begin{align*}
\vdash B & \quad \vdash \\
\vdash A \rightarrow B & \quad \vdash \quad I, i
\end{align*}
\]

\[\text{Figure 2.1: NATURAL DEDUCTION RULES FOR CONDITIONALS}\]

We have motivated two rules of inference. These rules are displayed in Figure 2.1. The first rule, \( \text{modus ponens, or conditional elimination } \rightarrow E \) allows us to step from a conditional and its antecedent to the consequent of the conditional. We call the conditional premise \( A \rightarrow B \) the major premise of the \( \rightarrow E \) inference, and the antecedent \( A \) the minor premise of that inference. When we apply the inference \( \rightarrow E \), we combine two proofs: the proof of \( A \rightarrow B \) and the proof of \( A \). The new proof has as assumptions any assumptions made in the proof of \( A \rightarrow B \) and also any assumptions made in the proof of \( A \). The conclusion is \( B \).

The second rule, \( \text{conditional introduction } \rightarrow I \) allows us to use a proof from \( A \) to \( B \) as a proof of \( A \rightarrow B \). The assumption of \( A \) is discharged in this step. The proof of \( A \rightarrow B \) has as its assumptions all of the assumptions used in the proof of \( B \) except for the instances of \( A \) that we discharged in this step. Its conclusion is \( A \rightarrow B \).
Definition 2.1 [Proofs for Conditionals] A proof is a tree, whose nodes are either formulas, or bracketed formulas. The formula at the root of the tree is said to be the conclusion of the proof. The unbracketed formulas at the leaves of the tree are the premises of the proof.

» Any formula $A$ is a proof, with premise $A$ and conclusion $A$.

» If $\pi_1$ is a proof, with conclusion $A \rightarrow B$ and $\pi_r$ is a proof, with conclusion $A$, then the following tree

$$
\begin{align*}
\vdots & \quad \vdots \\
\pi_1 & \quad \pi_r \\
A \rightarrow B & \quad A \\
\hline
B & \rightarrow E
\end{align*}
$$

is a proof with conclusion $B$, and having the premises consisting of the premises of $\pi_1$ together with the premises of $\pi_r$.

» If $\pi$ is a proof, for which $A$ is one of the premises and $B$ is the conclusion, then the following tree

$$
\begin{align*}
\vdots \\
A \\
\hline \vdots
B
\end{align*}
$$

is a proof of $A \rightarrow B$.

» Nothing else is a proof.

This is a recursive definition, in just the same manner as the recursive definition of the class formula.

Figure 2.2: Three Implicational Proofs

Figure 2.2 gives three proofs of implicational proofs constructed using our rules. The first is a proof from $A \rightarrow B$ to $(B \rightarrow C) \rightarrow (A \rightarrow C)$. This is the inference of suffixing. (We “suffix” both $A$ and $B$ with $\rightarrow C$.) The other proofs conclude in formulas justified on the basis of no undischarged assumptions. It is worth your time to read through

\[\PageIndex{2.2} \cdot Proofs for Conditionals\]
these proofs to make sure that you understand the way each proof is constructed.

You can try a number of different strategies when making proofs for yourself. For example, you might like to try your hand at constructing a proof to the conclusion that $B \rightarrow (A \rightarrow C)$ from the assumption $A \rightarrow (B \rightarrow C)$. Here are two ways to piece the proof together.

**CONSTRUCTING PROOFS TOP-DOWN:** You start with the assumptions and see what you can do with them. In this case, with $A \rightarrow (B \rightarrow C)$ you can, clearly, get $B \rightarrow C$, if you are prepared to assume $A$. And then, with the assumption of $B$ we can deduce $C$. Now it is clear that we can get $B \rightarrow (A \rightarrow C)$ if we discharge our assumptions, $A$ first, and then $B$.

**CONSTRUCTING PROOFS BOTTOM-UP:** Start with the conclusion, and find what you could use to prove it. Notice that to prove $B \rightarrow (A \rightarrow C)$ you could prove $A \rightarrow C$ using $B$ as an assumption. Then to prove $A \rightarrow C$ you could prove $C$ using $A$ as an assumption. So, our goal is now to prove $C$ using $A$, $B$ and $A \rightarrow (B \rightarrow C)$ as assumptions. But this is an easy pair of applications of $\rightarrow E$.

I have been intentionally unspecific when it comes to discharging formulas in proofs. In the examples in Figure 2.2 you will notice that at each step when a discharge occurs, one and only one formula is discharged. By this I do not mean that at each $\rightarrow I$ step a formula $A$ is discharged and a different formula $B$ is not. I mean that in the proofs we have seen so far, at each $\rightarrow I$ step, a single instance of the formula is discharged. Not all proofs are like this. Consider this proof from the assumption $A \rightarrow (A \rightarrow B)$ to the conclusion $A \rightarrow B$. At the final step of this proof, two instances of the assumption $A$ are discharged at once.

$$
\frac{A \rightarrow (A \rightarrow B)}{A \rightarrow B} \xrightarrow{\rightarrow E} \frac{A}{B} \xrightarrow{\rightarrow I, 1} A \rightarrow B
$$

For this to count as a proof, we must read the rule $\rightarrow I$ as licensing the discharge of one or more instances of a formula in the inference to the conditional. Once we think of the rule in this way, one further generalisation comes to mind: If we think of an $\rightarrow I$ move as discharging a collection of instances of our assumption, someone of a generalising spirit will ask if that collection can be empty. Can we discharge an assumption that isn’t there? If we can, then this counts as a proof:

$$
\frac{A}{B \rightarrow A} \xrightarrow{\rightarrow I, 1}
$$

Here, we assume $A$, and then, we infer $B \rightarrow A$ discharging all of the active assumptions of $B$ in the proof at this point. The collection of
active assumptions of B is, of course, empty. No matter, they are all discharged, and we have our conclusion: B → A.

You might think that this is silly: how can you discharge a nonexistent assumption? Nonetheless, discharging assumptions that are not there plays a role. To give you a taste of why, notice that the inference from A to B → A is valid if we read “→” as the material conditional of standard two-valued classical propositional logic. In a pluralist spirit we will investigate different policies for discharging formulas.

**Definition 2.2 [Discharge Policy]** A discharge policy may either allow or disallow duplicate discharge (more than one instance of a formula at once) or vacuous discharge (zero instances of a formula in a discharge step). Here are the names for the four discharge policies:

<table>
<thead>
<tr>
<th>VACUOUS</th>
<th>NO</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUPLCATES</td>
<td>YES</td>
</tr>
<tr>
<td>NO</td>
<td>&quot;Affine&quot;</td>
</tr>
</tbody>
</table>

The “standard” discharge policy is to allow both vacuous and duplicate discharge. There are reasons to explore each of the different combinations. As I indicated above, you might think that vacuous discharge is a bit silly. It is not merely silly: it seems downright wrong if you think that a judgement of the form A → B records the claim that B may be inferred from A. If A is not used in the inference to B, then we hardly have reason to think that B follows from A in this sense. So, if you are after a conditional which is relevant in this way, you would be interested in discharge policies that ban vacuous discharge [1, 2, 66].

There are also reasons to ban duplicate discharge: Victor Pambuccian has found an interesting example of doing without duplicate discharge in early 20th Century geometry [57]. He traces cases where geometers took care to keep track of the number of times a postulate was used in a proof. So, they draw a distinction between A → (A → B) and A → B. More recently, work in fuzzy logic [8, 37, 51] motivates keeping track of the number of times premises are used. If a conditional A → B fails to be true to the degree that A is truer than B, then A → (A → B) may be truer than A → B.

Finally, for some [4, 52, 63, 67], Curry’s Paradox motivates banning indiscriminate duplicate discharge. If we have a claim A which both implies A → B and is implied by it then we can reason as follows:

\[
\begin{align*}
[A]^{(1)} & \quad [A]^{(2)} \\
A \to B & \quad A \to B \\
B & \quad B \\
A & \quad \vdash \text{I,1} \\
\hline
A \to B & \quad \hline
B & \quad \to \text{E}
\end{align*}
\]

For more in a “pluralist spirit” see my work with Jc Beall [5, 6, 70].

I am not happy with the label “affine,” but that’s what the literature has given us. Does anyone have any better ideas for this? “Standard” is not “classical” because it suffices for intuitionistic logic in this context, not classical logic. It’s not “intuitionistic” because “intuitionistic” is difficult to pronounce, and it is not distinctively intuitionistic. As we shall see later, it’s the shape of proof and not the discharge policy that gives us intuitionistic implicational logic.

You must be careful if you think that more than one discharge policy is ok. Consider Exercise 19 in this chapter, in which it is shown that if you have two conditionals →₁ and →₂ with different discharge policies, the conditionals collapse into one (in effect having the most lax discharge policy of either →₁ or →₂). Consider Exercise 23 to explore how you might have the one logic with more than one conditional connective.

If this sentence is true, then I am a monkey’s uncle.

\[\text{§2.2 • Proofs for Conditionals} \]
Where we have used ‘†’ to mark the steps where we have gone from \( A \) to \( A \rightarrow B \) or back. Notice that this is a proof of \( B \) from no premises at all! So, if we have a claim \( A \) which is equivalent to \( A \rightarrow B \), and if we allow vacuous discharge, then we can derive \( B \).

We shall see more of the distinctive properties of different discharge policies as the book progresses.

**Definition 2.3 [Discharge in Proofs]** A proof in which every discharge is linear is a linear proof. Similarly, a proof in which every discharge is relevant is a relevant proof, a proof in which every discharge is affine is an affine proof. If a proof has some duplicate discharge and some vacuous discharge, it is at least a standard proof.

Proofs underwrite arguments. If we have a proof from a collection \( X \) of assumptions to a conclusion \( A \), then the argument \( X \vdash A \) is valid by the light of the rules we have used. So, in this section, we will think of arguments as structures involving a collection of assumptions and a single conclusion. But what kind of thing is that collection \( X \)? It isn’t a set, because the number of premises makes a difference: (The example here involves linear discharge policies. We will see later that even when we allow for duplicate discharge, there is a sense in which the number of occurrences of a formula in the premises might still matter.) There is a linear proof from \( A \rightarrow (A \rightarrow B), A, A \rightarrow B \):

\[
\frac{A \rightarrow (A \rightarrow B) \quad A \rightarrow B}{A} \rightarrow E
\]

We shall see later that there is no linear proof from \( A \rightarrow (A \rightarrow B), A \rightarrow B \). (If we ban duplicate discharge, then the number of assumptions in a proof matters.) The collection appropriate for our analysis at this stage is what is called a multiset, because we want to pay attention to the number of times we make an assumption in an argument.

**Definition 2.4 [Multiset]** Given a class \( X \) of objects (such as the class \( \text{FORMULA} \)), a multiset \( M \) of objects from \( X \) is a special kind of collection of elements of \( X \). For each \( x \) in \( X \), there is a natural number \( o_M(x) \), the number of occurrences of the object \( x \) in the multiset \( M \). The number \( o_M(x) \) is sometimes said to be the degree to which \( x \) is a member of \( M \). The multiset \( M \) is finite if \( o_M(x) > 0 \) for only finitely many objects \( x \). The multiset \( M \) is identical to the multiset \( M' \) if and only if \( o_M(x) = o_{M'}(x) \) for every \( x \) in \( X \).

Multisets may be presented in lists, in much the same way that sets can. For example, \([1, 2, 2] \) is the finite multiset containing 1 only once and 2 twice. \([1, 2, 2] \neq [2, 1, 2] \). We shall only consider finite multisets of formulas, and not multisets that contain other multisets as members. This means that we can do without the brackets and write our multisets as lists. We will write “\( A, B, B, C \)” for the finite multiset containing \( B \) twice and \( A \) and \( C \) once. The empty multiset, to which everything is a member to degree zero, is \([\,] \).
**Definition 2.5 [Comparing multisets]** When \( M \) and \( M' \) are multisets and \( o_M(x) \leq o_{M'}(x) \) for each \( x \) in \( X \), we say that \( M \) is a sub-multiset of \( M' \), and \( M' \) is a super-multiset of \( M \).

The **ground** of the multiset \( M \) is the set of all objects that are members of \( M \) to a non-zero degree. So, for example, the ground of the multiset \( A, B, B, C \) is the set \{A, B, C\}.

We use finite multisets as a part of a discriminating analysis of proofs and arguments. (An even more discriminating analysis will consider premises to be structured in lists, according to which \( A, B \) differs from \( B, A \). You can examine this in Exercise 24 on page 51.) We have no need to consider infinite multisets in this section, as multisets represent the premise collections in arguments, and it is quite natural to consider only arguments with finitely many premises. So, we will consider arguments in the following way.

**Definition 2.6 [Argument]** An argument \( X : A \) is a structure consisting of a finite multiset \( X \) of formulas as its **premises**, and a single formula \( A \) as its **conclusion**. The premise multiset \( X \) may be empty. An argument \( X : A \) is **standardly valid** if and only if there is some proof with undischarged assumptions forming the multiset \( X \), and with the conclusion \( A \). It is **relevantly valid** if and only if there is a relevant proof from the multiset \( X \) of premises to \( A \), and so on.

Here are some features of validity.

**Lemma 2.7 [Validity facts]** Let \( v \)-validity be any of linear, relevant, affine or standard validity.

1. \( A : A \) is \( v \)-valid.
2. \( X, A : B \) is \( v \)-valid if and only if \( X : A \rightarrow B \) is \( v \)-valid.
3. If \( X, A : B \) and \( Y : A \) are both \( v \)-valid, so is \( X, Y : B \).
4. If \( X : B \) is affine or standardly valid, so is \( X, A : B \).
5. If \( X, A, A : B \) is relevantly or standardly valid, so is \( X, A : B \).

**Proof:** It is not difficult to verify these claims: (1) is given by the proof consisting of \( A \) as premise and conclusion. For (2), take a proof \( \pi \) from \( X, A \) to \( B \), and in a single step \( \rightarrow I \), discharge the (single instance of) \( A \) to construct the proof of \( A \rightarrow B \) from \( X \). Conversely, if you have a proof from \( X \) to \( A \rightarrow B \), add a (single) premise \( A \) and apply \( \rightarrow E \) to derive \( B \). In both cases here, if the original proofs satisfy a constraint (vacuous or multiple discharge) so do the new proofs.

For (3), take a proof from \( X, A \) to \( B \), but replace the instance of assumption of \( A \) indicated in the premises, and replace this with the proof from \( Y \) to \( A \). The result is a proof, from \( X, Y \) to \( B \) as desired. This proof satisfies the constraints satisfied by both of the original proofs.

---

John Slaney has joked that the empty multiset [\[] should be distinguished from the empty set \{\}, since nothing is a member of \{\}, but everything is a member of [\[] zero times.
For (4), if we have a proof $\pi$ from $X$ to $B$, we extend it as follows

\[
\begin{array}{c}
X \\
\vdots \pi \\
B \\
\hline
\end{array}
\quad \Rightarrow I \\
\begin{array}{c}
A \rightarrow B \\
\vdash \pi, i \\
\hline
A \\
\hline
\end{array}
\rightarrow E
\]

to construct a proof to $B$ involving the new premise $A$, as well as the original premises $X$. The $\Rightarrow I$ step requires a vacuous discharge.

Finally (5): if we have a proof $\pi$ from $X, A, A$ to $B$ (that is, a proof with $X$ and two instances of $A$ as premises to derive the conclusion $B$) we discharge the two instances of $A$ to derive $A \rightarrow B$ and then reinstate a single instance of $A$ to as a premise to derive $B$ again.

\[
\begin{array}{c}
X, [A, A]^{(1)} \\
\vdash \pi \\
B \\
\hline
\end{array}
\quad \Rightarrow I, i \\
\begin{array}{c}
A \rightarrow B \\
\vdash \pi, i \\
\hline
A \\
\hline
\end{array}
\rightarrow E
\]

Now, we might focus our attention on the distinction between those arguments that are valid and those that are not—to focus on facts about validity such as those we have just proved. That would be to ignore the distinctive features of proof theory. We care not only that an argument is proved, but how it is proved. For each of these facts about validity, we showed not only the bare existential fact (for example, if there is a proof from $X, A$ to $B$, then there is a proof from $X$ to $A \rightarrow B$) but the stronger and more specific fact (if there is a proof from $X, A$ to $B$ then from this proof we construct the proof from $X$ to $A \rightarrow B$ in this uniform way).

It is often a straightforward matter to show that an argument is valid. Find a proof from the premises to the conclusion, and you are done. Showing that an argument is not valid seems more difficult. According to the literal reading of this definition, if an argument is not valid there is no proof from the premises to the conclusion. So, the direct way to show that an argument is invalid is to show that it has no proof from the premises to the conclusion. But there are infinitely many proofs! You cannot simply go through all of the proofs and check that none of them are proofs from $X$ to $A$ in order to convince yourself that the argument is not valid. To accomplish this task, subtlety is called for. We will end this section by looking at how we might summon up the required skill.

One subtlety would be to change the terms of discussion entirely, and introduce a totally new concept. If you could show that all valid arguments have some special property—and one that is easy to detect when present and when absent—then you could show that an argument
is invalid by showing it lacks that special property. How this might manage to work depends on the special property. We shall look at one of these properties in Section ?? when we show that all valid arguments preserve truth in models. Then to show that an argument is invalid, you could provide a model in which truth is not preserved from the premises to the conclusion. If all valid arguments are truth-in-a-model-preserving, then such a model would count as a counterexample to the validity of your argument.

In this section, on the other hand, we will not go beyond the conceptual bounds of the study of proof. We will find instead a way to show that an argument is invalid, using an analysis of proofs themselves. The collection of all proofs is too large to survey. From premises X and conclusion A, the collection of direct proofs – those that go straight from X to A without any detours down byways or highways – might be more tractable. If we could show that there are not many direct proofs from a given collection of premises to a conclusion, then we might be able to exploit this fact to show that for a given set of premises and a conclusion there are no direct proofs from X to A. If, in addition, you were to show that any proof from a premise set to a conclusion could somehow be converted into a direct proof from the same premises to that conclusion, then you would have success in showing that there is no proof from X to A.

Happily, this technique works. But to make this work we need to understand what it is for a proof to be “direct” in some salient sense. Direct proofs have a name—they are ‘normal’.

### 2.3 | NORMAL PROOFS

It is best to introduce normal proofs by contrasting them with non-normal proofs. And non-normal proofs are not difficult to find. Suppose you want to show that the following argument is valid

\[
p \rightarrow q \therefore p \rightarrow ((q \rightarrow r) \rightarrow r)
\]

You might note first that we have already seen an argument which takes us from \(p \rightarrow q\) to \((q \rightarrow r) \rightarrow (p \rightarrow r)\). This is suffixing. ![](image)

I think that the terminology ‘normal’ comes from Prawitz [62], though the idea comes from Gentzen.
So, we have \( p \to q :. (q \to r) \to (p \to r) \). But we also have the general principle *permuting* antecedents: \( A \to (B \to C) :. B \to (A \to C) \).

\[
\begin{align*}
A \to (B \to C) & \quad \{ A \}^{(3)} \\
B \to C & \quad \rightarrow E \{ B \}^{(4)} \\
C & \quad \rightarrow l,3 \\
A \to C & \quad \rightarrow l,4 \\
B \to (A \to C) & \quad \rightarrow E
\end{align*}
\]

We can apply this in the case where \( A = (q \to r) \), \( B = p \) and \( C = r \) to get \( (q \to r) \to (p \to r) :. p \to ((q \to r) \to r) \). We then chain our two arguments, to get us from \( p \to q \) to \( p \to ((q \to r) \to r) \), which we wanted. But take a look at the resulting *proof*.

\[
\begin{align*}
& \quad p \to q \quad \{ p \}^{(1)} \\
& \quad \{ q \to r \}^{(2)} \\
\Rightarrow & \quad q \quad \rightarrow E \\
\Rightarrow & \quad r \quad \rightarrow l,1 \\
\Rightarrow & \quad p \to r \quad \rightarrow l,2 \\
\Rightarrow & \quad (q \to r) \to (p \to r) \quad \rightarrow l,3 \\
\Rightarrow & \quad [q \to r]^{(3)} \\
\Rightarrow & \quad [q \to r]^{(4)} \quad \rightarrow E \\
\Rightarrow & \quad p \to r \\
\Rightarrow & \quad r \quad \rightarrow l,3 \\
\Rightarrow & \quad (q \to r) \to r \quad \rightarrow l,4 \\
\Rightarrow & \quad p \to ((q \to r) \to r) \quad \rightarrow E
\end{align*}
\]

This proof is *odd*. It gets us from our premise \( p \to q \) to our conclusion \( p \to ((q \to r) \to r) \), but it does it in a roundabout way. We break down the conditionals \( p \to q \), \( q \to r \) to construct \( (q \to r) \to (p \to r) \) halfway through the proof, only to break that down again (deducing \( r \) on its own, for a second time) to build the required conclusion. This is most dramatic around the intermediate conclusion \( p \to ((q \to r) \to r) \) which is built up *from* \( p \to r \) only to be used to justify \( p \to r \) at the next step. We may eliminate this redundancy by cutting out the intermediate formula \( p \to ((q \to r) \to r) \) like this:

\[
\begin{align*}
& \quad p \to q \quad \{ p \}^{(1)} \\
\Rightarrow & \quad q \quad \rightarrow E \\
\Rightarrow & \quad r \quad \rightarrow l,1 \\
\Rightarrow & \quad p \to r \\
\Rightarrow & \quad r \quad \rightarrow l,3 \\
\Rightarrow & \quad (q \to r) \to r \quad \rightarrow l,4 \\
\Rightarrow & \quad p \to ((q \to r) \to r)
\end{align*}
\]

The resulting proof is a lot simpler already. But now the \( p \to r \) is constructed from \( r \) only to be broken up immediately to return \( r \). We
can delete the redundant \( p \rightarrow r \) in the same way.

\[
\begin{align*}
 p \rightarrow q & \quad [p]^{(4)} \\
 [q \rightarrow r]^{(3)} & \quad q \rightarrow E \\
 r & \quad \rightarrow E \\
 (q \rightarrow r) \rightarrow r & \quad \rightarrow I, 3 \\
 p \rightarrow ((q \rightarrow r) \rightarrow r) & \quad \rightarrow I, 4
\end{align*}
\]

Now the proof takes us directly from the premise to the conclusion, through no extraneous formulas. Every formula used in this proof is either found in the premise, or in the conclusion. This wasn’t true in the original, roundabout proof. We say this new proof is normal, the original proof was not.

This is a general phenomenon. Take a proof that concludes with an implication introduction: it infers from \( A \) to \( B \) by way of the sub-proof \( \pi_1 \). Then we discharge the \( A \) to conclude \( A \rightarrow B \). Imagine that at the very next step, it uses a different proof – call it \( \pi_2 \) – with conclusion \( A \) to deduce \( B \) by means of an implication elimination. This proof contains a redundant step. Instead of taking the detour through the formula \( A \rightarrow B \), we could use the proof \( \pi_1 \) of \( B \), but instead of taking \( A \) as an assumption, we could use the proof of \( A \) we have at hand, namely \( \pi_2 \). The before-and-after comparison is this:

\[
\begin{align*}
 [A]^{(1)} \\
 \vdots \pi_1 \\
 B & \quad \rightarrow I, 1 \\
 A \rightarrow B & \quad \rightarrow I, 2 \\
 A & \quad \rightarrow I, 1 \\
 B & \quad \rightarrow E \quad \vdots \pi_1 \\
 \end{align*}
\]

The result is a proof of \( B \) from the same premises as our original proof. The premises are the premises of \( \pi_1 \) (other than the instances of \( A \) that were discharged in the other proof) together with the premises of \( \pi_2 \). This proof does not go through the formula \( A \rightarrow B \), so it is, in a sense, simpler.

Well . . . there are some subtleties with counting, as usual with our proofs. If the discharge of \( A \) was vacuous, then we have nowhere to plug in the new proof \( \pi_2 \), so the premises of \( \pi_2 \) don’t appear in the final proof. On the other hand, if a number of duplicates of \( A \) were discharged, then the new proof will contain that many copies of \( \pi_2 \), and hence, that many copies of the premises of \( \pi_2 \). Let’s make this discussion more explicit, by considering an example where \( \pi_1 \) has two instances of \( A \) in the premise list. The original proof containing the
introduction and then elimination of \( A \rightarrow B \) is

\[
\begin{array}{c}
A \rightarrow (A \rightarrow B) \quad [A]^{(1)} \\
\hline
A \rightarrow B \quad \rightarrow E \quad [A]^{(1)} \\
\hline
B \quad \rightarrow I,1 \\
\hline
A \rightarrow B \quad \rightarrow E \\
\hline
A \rightarrow A \quad \rightarrow E \\
\hline
\quad \rightarrow I,2 \\
\hline
A \rightarrow A \quad \rightarrow E \\
\hline
\quad \rightarrow I,2 \\
\hline
A \rightarrow A \quad \rightarrow E \\
\hline
\quad \rightarrow E \\
\hline
\end{array}
\]

We can Cut out the \( \rightarrow I/\rightarrow E \) pair (we call such pairs indirect pairs) using the technique described above, we place a copy of the inference to \( A \) at both places that the \( A \) is discharged (with label 1). The result is this proof, which does not make that detour.

\[
\begin{array}{c}
A \rightarrow (A \rightarrow B) \quad [A]^{(2)} \\
\hline
(A \rightarrow A) \rightarrow A \quad \rightarrow I,2 \\
\hline
A \rightarrow A \quad \rightarrow E \\
\hline
\quad \rightarrow I,2 \\
\hline
A \rightarrow A \quad \rightarrow E \\
\hline
\quad \rightarrow E \\
\hline
\quad \rightarrow E \\
\hline
\quad \rightarrow E \\
\hline
\end{array}
\]

which is a proof from the same premises \((A \rightarrow (A \rightarrow B)) \) and \((A \rightarrow A) \rightarrow A\) to the same conclusion \( B \), except for multiplicity. In this proof the premise \((A \rightarrow A) \rightarrow A\) is used twice instead of once. (Notice too that the label ‘2’ is used twice. We could relabel one subproof to \( A \rightarrow A\) to use a different label, but there is no ambiguity here because the two proofs to \( A \rightarrow A\) do not overlap. Our convention for labelling is merely that at the time we get to an \( \rightarrow I \) label, the numerical tag is unique in the proof above that step.)

We have motivated the concept of normality: now comes the formal definition:

**DEFINITION 2.8 [NORMAL PROOF]** A proof is normal if and only if the concluding formula \( A \rightarrow B \) of an \( \rightarrow I \) step is not at the same time the major premise of an \( \rightarrow E \) step.

**DEFINITION 2.9 [INDIRECT PAIR; DETOUR FORMULA]** If a formula \( A \rightarrow B \) introduced in an \( \rightarrow I \) step is at the same time the major premise of an \( \rightarrow E \) step, then we shall call this pair of inferences an indirect pair and we will call the instance \( A \rightarrow B \) in the middle of this indirect pair a detour formula in the proof.

So, a normal proof is one without any indirect pairs. It has no detour formulas.

Normality is not only important for proving that an argument is invalid by showing that it has no normal proofs. The claim that every valid argument has a normal proof could well be vital. If we think of the rules for conditionals as somehow defining the connective, then proving something by means of a roundabout \( \rightarrow I/\rightarrow E \) step that you cannot
prove without it would seem to be quite illicit. If the conditional is defined by way of its rules then it seems that the things one can prove from a conditional ought to be merely the things one can prove from whatever it was you used to introduce the conditional. If we could prove more from a conditional \( A \rightarrow B \) than one could prove on the basis on the information used to introduce the conditional, then we are conjuring new arguments out of thin air.

For this reason, many have thought that being able to convert non-normal proofs to normal proofs is not only desirable, it is critical if the proof system is to be properly logical. We will not continue in this philosophical vein here. We will take up this topic in a later section, after we understand the behaviour of normal proofs a little better. Let us return to the study of normal proofs.

Normal proofs are, intuitively at least, proofs without a kind of redundancy. It turns out that avoiding this kind of redundancy in a proof means that you must avoid another kind of redundancy too. A normal proof from \( X \) to \( A \) may use only a very restricted repertoire of formulas. It will contain only the subformulas of \( X \) and \( A \).

definition 2.10 [subformulas and parse trees] The parse tree for an atom is that atom itself. The parse tree for a conditional \( A \rightarrow B \) is the tree containing \( A \rightarrow B \) at the root, connected to the parse tree for \( A \) and the parse tree for \( B \). The subformulas of a formula \( A \) are those formulas found in \( A \)'s parse tree. We let \( sf(A) \) be the set of all subformulas of \( A \). \( sf(p) = \{ p \} \), and \( sf(A \rightarrow B) = \{ A \rightarrow B \} \cup sf(A) \cup sf(B) \). To generalise, when \( X \) is a multiset of formulas, we will write \( sf(X) \) for the set of subformulas of each formula in \( X \).

Here is the parse tree for \( (p \rightarrow q) \rightarrow ((q \rightarrow r) \rightarrow p) \):

\[
\begin{array}{ccc}
\vdots & \vdots & \vdots \\
p & q & r \\
p \rightarrow q & q \rightarrow r & p \\
(\vdots) & (\vdots) & (\vdots) \\
(\vdots) & (\vdots) & (\vdots) \\
(p \rightarrow q) & (q \rightarrow r) & p
\end{array}
\]

So, \( sf((p \rightarrow q) \rightarrow ((q \rightarrow r) \rightarrow p)) = ((p \rightarrow q), \rightarrow ((q \rightarrow r) \rightarrow p), p \rightarrow q, p, q, (q \rightarrow r) \rightarrow p, q, q, r, r) \).

We may prove the following theorem.

Theorem 2.11 [the subformula theorem] Each normal proof from the premises \( X \) to the conclusion \( A \) contains only formulas in \( sf(X, A) \).

Notice that this is not the case for non-normal proofs. Consider the following circuitous proof from \( A \) to \( A \).

\[
\frac{\begin{array}{c} [A]^{(1)} \\vdash_{L1} L \\vdash_{E} E \end{array}}{A \rightarrow A \rightarrow_{E} A \rightarrow_{E} A}
\]

§2.3 • Normal Proofs
Here \( A \rightarrow A \) is in the proof, but it is not a subformula of the premise (\( A \)) or the conclusion (also \( A \)).

The subformula property for normal proofs goes some way to reassure us that a normal proof is direct. A normal proof from \( X \) to \( A \) cannot stray so far away from the premises and the conclusion so as to incorporate material outside \( X \) and \( A \).

**Proof:** To prove the subformula theorem, we need to look carefully at how proofs are constructed. If \( \pi \) is a normal proof, then it is constructed in exactly the same way as all proofs are, but the fact that the proof is normal gives us some useful information. By the definition of proofs, \( \pi \) either is a lone assumption, or \( \pi \) ends in an application of \( \rightarrow I \), or it ends in an application of \( \rightarrow E \). Assumptions are the basic building blocks of proofs. We will show that assumption-only proofs have the subformula property, and then, also show on the assumption that the proofs we have on had have the subformula property, then the normal proofs we construct from them also have the property. Then it will follow that all normal proofs have the subformula property, because all of the normal proofs can be generated in this way.

**Assumption** A sole assumption, considered as a proof, satisfies the subformula property. The assumption \( A \) is the only constituent of the proof and it is both a premise and the conclusion.

**Introduction** In the case of \( \rightarrow I \), \( \pi \) is constructed from another normal proof \( \pi' \) from \( X \) to \( B \), with the new step added on (and with the discharge of a number – possibly zero – of assumptions). \( \pi \) is a proof from \( X' \) to \( A \rightarrow B \), where \( X' \) is \( X \) with the deletion of some number of instances of \( A \). Since \( \pi' \) is normal, we may assume that every formula in \( \pi' \) is in \( sf(X, B) \). Notice that \( sf(X', A \rightarrow B) \) contains every element of \( sf(X, B) \), since \( X \) differs only from \( X' \) by the deletion of some instances of \( A \). So, every formula in \( \pi \) (namely, those formulas in \( \pi' \), together with \( A \rightarrow B) \) is in \( sf(X', A \rightarrow B) \) as desired.

**Elimination** In the case of \( \rightarrow E \), \( \pi \) is constructed out of *two* normal proofs: one (call it \( \pi_1 \)) to the conclusion of a conditional \( A \rightarrow B \) from premises \( X \), and the other (call it \( \pi_2 \)) to the conclusion of the antecedent of that conditional \( A \) from premises \( Y \). Both \( \pi_1 \) and \( \pi_2 \) are normal, so we may assume that each formula in \( \pi_1 \) is in \( sf(X, A \rightarrow B) \) and each formula in \( \pi_2 \) is in \( sf(Y, A) \). We wish to show that every formula in \( \pi \) is in \( sf(X, Y, B) \). This seems difficult (\( A \rightarrow B \) is in the proof—where can it be found inside \( X, Y \) or \( B \)?), but we also have some more information: \( \pi_1 \) cannot end in the *introduction* of the conditional \( A \rightarrow B \). So, \( \pi_1 \) is either the assumption \( A \rightarrow B \) itself (in which case \( Y = A \rightarrow B \), and clearly in this case each formula in \( \pi \) is in \( sf(X, A \rightarrow B, B) \)) or \( \pi_1 \) ends in a \( \rightarrow E \) step. But if \( \pi_1 \) ends in an \( \rightarrow E \) step, the major premise of that inference is a formula of the form \( C \rightarrow (A \rightarrow B) \). So \( \pi_1 \) contains the formula \( C \rightarrow (A \rightarrow B) \), so *whatever* list \( Y \) is, \( C \rightarrow (A \rightarrow B) \in sf(Y, A) \),
and so, $A \rightarrow B \in \text{sf}(Y)$. In this case too, every formula in $\pi$ is in $\text{sf}(X, Y, B)$, as desired.

This completes the proof of our theorem. Every normal proof is constructed from assumptions by introduction and elimination steps in this way. The subformula property is preserved through each step of the construction.

Normal proofs are useful to work with. Even though an argument might have very many proofs, it will have many fewer normal proofs, and we can exploit this fact.

**Example 2.12 [No Normal Proofs]** There is no normal proof from $p$ to $q$. There is no normal relevant proof from $p \rightarrow r$ to $p \rightarrow (q \rightarrow r)$.

**Proof:** Normal proofs from $p$ to $q$ (if there are any) contain only formulas in $\text{sf}(p, q)$: that is, they contain only $p$ and $q$. That means they contain no $\rightarrow I$ or $\rightarrow E$ steps, since they contain no conditionals at all. It follows that any such proof must consist solely of an assumption. As a result, the proof cannot have a premise $p$ that differs from the conclusion $q$. There is no normal proof from $p$ to $q$.

Consider the second example: If there is a normal proof of $p \rightarrow (q \rightarrow r)$, from $p \rightarrow r$, it must end in an $\rightarrow I$ step, from a normal (relevant) proof from $p \rightarrow r$ and $p$ to $q \rightarrow r$. Similarly, this proof must also end in an $\rightarrow I$ step, from a normal (relevant) proof from $p \rightarrow r$, $p$ and $q$ to $r$. Now, what normal relevant proofs can be found from $p \rightarrow r$, $p$ and $q$ to $r$? There are none! Any such proof would have to use $q$ as a premise somewhere, but since it is normal, it contains only subformulas of $p \rightarrow r$, $p$, $q$ and $r$—namely those formulas themselves. There is no formula involving $q$ other than $q$ itself on that list, so there is nowhere for $q$ to go. It cannot be used, so it will not be a premise in the proof. There is no normal relevant proof from the premises $p \rightarrow r$, $p$ and $q$ to the conclusion $r$.

These facts are interesting enough. It would be more productive, however, to show that there is no proof at all from $p$ to $q$, and no relevant proof from $p \rightarrow r$ to $p \rightarrow (q \rightarrow r)$. We can do this if we have some way of showing that if we have a proof for some argument, we have a normal proof for that argument.

So, we now work our way towards the following theorem:

**Theorem 2.13 [Normalisation Theorem]** A proof $\pi$ from $X$ to $A$ reduces in some number of steps to a normal proof $\pi'$ from $X'$ to $A$.

If $\pi$ is linear, so is $\pi'$, and $X = X'$. If $\pi$ is affine, so is $\pi'$, and $X'$ is a sub-multiset of $X$. If $\pi$ is relevant, then so is $\pi'$, and $X'$ covers the same ground as $X$, and is a super-multiset of $X$. If $\pi$ is standard, then so is $\pi'$, and $X'$ covers no more ground than $X$.
Notice how the premise multiset of the normal proof is related to the premise multiset of the original proof. If we allow duplicate discharge, then the premise multiset may contain formulas to a greater degree than in the original proof, but the normal proof will not contain any new premises. If we allow vacuous discharge, then the normal proof might contain fewer premises than the original proof.

The normalisation theorem mentions the notion of reduction, so let us first define it.

**Definition 2.14 [Reduction]** A proof $\pi$ reduces to $\pi'$ ($\pi \rightsquigarrow \pi'$) if some indirect pair in $\pi$ is eliminated in the usual way.

$$
\begin{array}{c}
[A]^{(i)} \\
\vdots \pi_1 \\
B \\
A \rightarrow B \quad \rightarrow_{l, i} \quad \vdots \pi_2 \\
A \quad \rightarrow_E \quad \vdots \pi_1 \\
\vdots \\
B \\
\vdots \\
C
\end{array}
$$

If there is no $\pi'$ such that $\pi \rightsquigarrow \pi'$, then $\pi'$ is normal. If $\pi_0 \rightsquigarrow \pi_2 \rightsquigarrow \cdots \rightsquigarrow \pi_n$ we write “$\pi_0 \rightsquigarrow_{n} \pi_n$” and we say that $\pi_0$ reduces to $\pi_n$ in a number of steps. We aim to show that for any proof $\pi$, there is some normal $\pi^*$ such that $\pi \rightsquigarrow_{k} \pi^*$.

The only difficult part in proving the normalisation theorem is showing that the process reduction can terminate in a normal proof. In the case where we do not allow duplicate discharge, there is no difficulty at all.

**Proof [Theorem 2.13: Linear and Affine Cases]**: If $\pi$ is a linear proof, or is an affine proof, then whenever you pick an indirect pair and normalise it, the result is a shorter proof. At most one copy of the proof $\pi_2$ for $A$ is inserted into the proof $\pi_1$. (Perhaps no substitution is made in the case of an affine proof, if a vacuous discharge was made.) Proofs have some finite size, so this process cannot go on indefinitely. Keep deleting indirect pairs until there are no pairs left to delete. The result is a normal proof to the conclusion $A$. The premises $X$ remain undisturbed, except in the affine case, where we may have lost premises along the way. (An assumption from $\pi_2$ might disappear if we did not need to make the substitution.) In this case, the premise multiset $X'$ from the normal proof is a sub-multiset of $X$, as desired.

If we allow duplicate discharge, however, we cannot be sure that in normalising we go from a larger to a smaller proof. The example on page 24 goes from a proof with 11 formulas to another proof with 11 formulas. The result is no smaller, so size is no guarantee that the process terminates.

To gain some understanding of the general process of transforming a non-normal proof into a normal one, we must find some other measure.
that decreases as normalisation progresses. If this measure has a least value then we can be sure that the process will stop. The appropriate measure in this case will not be too difficult to find. Let’s look at a part of the process of normalisation: the complexity of the formula that is normalised.

**Definition 2.15 [Complexity]** A formula’s complexity is the number of connectives in that formula. In this case, it is the number of instances of ‘$\rightarrow$’ in the formula.

The crucial features of complexity are that each formula has a finite complexity, and that the proper subformulas of a formula each have a lower complexity than the original formula. This means that complexity is a good measure for an induction, like the size of a proof.

Now, suppose we have a proof containing just one indirect pair, introducing and eliminating $A \rightarrow B$, and suppose that otherwise, $\pi_1$ (the proof of $B$ from $A$) and $\pi_2$ (the proof of $A$) are normal.

$$
\begin{array}{c}
|A| \rightarrow \\
\vdots \pi_1 \quad \vdots \pi_2 \\
\hline
\text{Before:} & B & \quad B \\
A \rightarrow B & A \rightarrow E
\end{array}
$$

Unfortunately, the new proof is not necessarily normal. The new proof is non-normal if $\pi_2$ ends in the introduction of $A$, while $\pi_1$ starts off with the elimination of $A$. Notice, however, that the non-normality of the new proof is, somehow, smaller. There is no non-normality with respect to $A \rightarrow B$, or any other formula that complex. The potential non-normality is with respect to a subformula $A$. This result would still hold if the proofs $\pi_1$ and $\pi_2$ weren’t normal themselves, but when they might have $\rightarrow I/\rightarrow E$ pairs for formulas less complex than $A \rightarrow B$. If $A \rightarrow B$ is the most complex detour formula in the original proof, then the new proof has a smaller most complex detour formula.

**Definition 2.16 [Non-normality]** The non-normality measure of a proof is a sequence $(c_1, c_2, \ldots, c_n)$ of numbers such that $c_i$ is the number of indirect pairs of formulas of complexity $i$. The sequence for a proof stops at the last non-zero value. Sequences are ordered with their last number as most significant. That is, $(c_1, \ldots, c_n) > (d_1, \ldots, d_m)$ if and only if $n > m$, or if $n = m$, when $c_n > d_n$, or if $c_n = d_n$, when $(c_1, \ldots, c_{n-1}) > (d_1, \ldots, d_{n-1})$.

Non-normality measures satisfy the finite descending chain condition. Starting at any particular measure, you cannot find any infinite descending chain of measures below it. There are infinitely many measures smaller than $(0, 1)$ (in this case, $(0), (1), (2), \ldots$). However, to form a descending sequence from $(0, 1)$ you must choose one of these as your next measure. Say you choose $(500)$. From that, you have only finitely
many (500, in this case) steps until \( \emptyset \). This generalises. From the sequence \( \langle c_1, \ldots, c_n \rangle \), you lower \( c_n \) until it gets to zero. Then you look at the index for \( n-1 \), which might have grown enormously. Nonetheless, it is some finite number, and now you must reduce this value. And so on, until you reach the last quantity, and from there, the empty sequence \( \emptyset \). Here is an example sequence using this ordering \( \langle 3, 2, 30 \rangle \rightarrow \langle 2, 8, 23 \rangle \rightarrow \langle 1, 47, 15 \rangle \rightarrow \langle 138, 478 \rangle \cdots \rightarrow \langle 1, 3088 \rangle \rightarrow \langle 314159 \rangle \rightarrow \cdots \rightarrow \langle 1 \rangle \rightarrow \emptyset \).

**Lemma 2.17** [Non-normality Reduction] Any a proof with an indirect pair reduces in one step to some proof with a lower measure of non-normality.

**Proof:** Choose a detour formula in \( \pi \) of greatest complexity (say \( n \)), such that its proof contains no other detour formulas of complexity \( n \). Normalise that proof. The result is a proof \( \pi' \) with fewer detour formulas of complexity \( n \) (and perhaps many more of \( n-1 \), etc.). So, it has a lower non-normality measure.

Now we have a proof of our normalisation theorem.

**Proof** [of Theorem 2.13: relevant and standard case]: Start with \( \pi \), a proof that isn’t normal, and use Lemma 2.17 to choose a proof \( \pi' \) with a lower measure of non-normality. If \( \pi' \) is normal, we’re done. If it isn’t, continue the process. There is no infinite descending chain of non-normality measures, so this process will stop at some point, and the result is a normal proof.

Every proof may be transformed into a normal proof. If there is a linear proof from \( X \) to \( A \) then there is a normal linear proof from \( X \) to \( A \). Linear proofs are satisfying and strict in this manner. If we allow vacuous discharge or duplicate discharge, matters are not so straightforward. For example, there is a non-normal standard proof from \( p, q \) to \( p \):

\[
\frac{p \rightarrow (p \rightarrow q) \quad [p]^{(1)} \rightarrow E}{p \rightarrow q} \quad \frac{q \rightarrow p \quad q}{p} \rightarrow E
\]

but there is no normal standard proof from exactly these premises to the same conclusion, since any normal proof from atomic premises to an atomic conclusion must be an assumption alone. We have a normal proof from \( p \) to \( p \) (it is very short!), but there is no normal proof from \( p \) to \( p \) that involves \( q \) as an extra premise.

Similarly, there is a relevant proof from \( p \rightarrow (p \rightarrow q) \), \( p \) to \( q \), but it is non-normal:

\[
\frac{p \rightarrow (p \rightarrow q) \quad [p]^{(1)} \rightarrow E}{p \rightarrow q} \quad \frac{q \rightarrow p \quad q}{p} \rightarrow E
\]

\[
\frac{p \rightarrow q \quad p}{q} \rightarrow E
\]
There is no normal relevant proof from \( p \to (p \to q) \), \( p \) to \( q \). Any normal relevant proof from \( p \to (p \to q) \) and \( p \) to \( q \) must use \( \to E \) to deduce \( p \to q \), and then the only other possible move is either \( \to I \) (in which case we return to \( p \to (p \to q) \) none the wiser) or we perform another \( \to E \) with another assumption \( p \) to deduce \( q \), and we are done. Alas, we have claimed two undischarged assumptions of \( p \). In the non-linear cases, the transformation from a non-normal to a normal proof does damage to the number of times a premise is used.

2.4 | STRONG NORMALISATION

It is very tempting to view normalisation as a way of eliminating redundancies and making explicit the structure of a proof. However, if that is the case, then it should be the case that the process of normalisation cannot give us two distinct “answers” for the structure of the one proof. Can two different reduction sequences for a single proof result in different normal proofs? To investigate this, we need one more notion of reduction.

DEFINITION 2.18 [PARALLEL REDUCTION] A proof \( \pi \) parallel reduces to \( \pi' \) if some number of indirect pairs in \( \pi \) are eliminated in parallel. We write “\( \pi \parallel \pi' \).”

For example, consider the proof with the following two detour formulas marked:

\[
A \to (A \to B) \quad [A]^{(1)}
\]

\[
\begin{align*}
A & \to B \quad \to E \\
B & \quad \to E
\end{align*}
\]

\[
A \to B \quad \to E
\]

\[
A \to A \quad \to I
\]

\[
A \to A \quad \to E
\]

To process them we can take them in any order. Eliminating the \( A \to B \), we have

\[
\begin{align*}
[A]^{(2)} & \\
A & \quad \to I
\end{align*}
\]

\[
\begin{align*}
A \to (A \to B) & \\
A & \quad \to E
\end{align*}
\]

\[
\begin{align*}
A \to B & \\
B & \quad \to E
\end{align*}
\]

which now has two copies of the \( A \to A \) to be reduced. However, these copies do not overlap in scope (they cannot, as they are duplicated in the place of assumptions discharged in an eliminated \( \to I \) rule) so they can be processed together. The result is the proof

\[
A \to (A \to B) \quad A
\]

\[
\begin{align*}
A & \quad \to E \\
B & \quad \to E
\end{align*}
\]

This passage is the hardest part of Chapter 2. Feel free to skip over the proofs of theorems in this section, until page 36 on first reading.
You can check that if you had processed the formulas to be eliminated in the other order, the result would have been the same.

**Lemma 2.19 [Diamond Property for \( \rightsquigarrow \)]** If \( \pi \rightsquigarrow \pi_1 \) and \( \pi \rightsquigarrow \pi_2 \) then there is some \( \pi' \) where \( \pi_1 \rightsquigarrow \pi' \) and \( \pi_2 \rightsquigarrow \pi' \).

**Proof:** Take the detour formulas in the proof \( \pi \) that are eliminated in either the move to \( \pi_1 \) or the move to \( \pi_2 \). ‘Colour’ them in \( \pi \), and transform the proof to \( \pi_1 \). Some of the coloured formulas may remain. Do the same in the move from \( \pi \) to \( \pi_2 \). The result are two proofs \( \pi_1 \) and \( \pi_2 \) in which some formulas may be coloured. The proof \( \pi' \) is found by parallel reducing either collection of formulas in \( \pi_1 \) or \( \pi_2 \).

**Theorem 2.20 [Only One Normal Form]** Any sequence of reduction steps from a proof \( \pi \) that terminates, terminates in a unique normal proof \( \pi^* \).

**Proof:** Suppose that \( \pi \rightsquigarrow_\pi \pi' \), and \( \pi \rightsquigarrow_\pi \pi'' \). It follows that we have two reduction sequences

\[
\pi \rightsquigarrow \pi_1 \rightsquigarrow \pi_2 \rightsquigarrow \cdots \rightsquigarrow \pi_n \rightsquigarrow \pi' \\
\pi \rightsquigarrow \pi_1'' \rightsquigarrow \pi_2'' \rightsquigarrow \cdots \rightsquigarrow \pi_m \rightsquigarrow \pi''
\]

By the diamond property, we have a \( \pi_1,1 \) where \( \pi_1 \rightsquigarrow \pi_{1,1} \) and \( \pi_1'' \rightsquigarrow \pi_{1,1} \). Then \( \pi_1'' \rightsquigarrow \pi_{1,1} \) and \( \pi_1'' \rightsquigarrow \pi_{2,1} \) so by the diamond property there is some \( \pi_2,1 \) where \( \pi_2'' \rightsquigarrow \pi_{2,1} \) and \( \pi_{1,1} \rightsquigarrow \pi_{2,1} \). Continue in this vein, guided by the picture below:

\[
\pi \rightsquigarrow \pi_1 \rightsquigarrow \pi_2 \rightsquigarrow \cdots \rightsquigarrow \pi_n \\
\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \\
\pi_1'' \rightsquigarrow_\pi \pi_{1,1} \rightsquigarrow_\pi \pi_{1,2} \rightsquigarrow_\pi \cdots \rightsquigarrow_\pi \pi_{1,n} \\
\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \\
\pi_2'' \rightsquigarrow_\pi \pi_{2,1} \rightsquigarrow_\pi \pi_{2,2} \rightsquigarrow_\pi \cdots \rightsquigarrow_\pi \pi_{2,n} \\
\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \\
\vdots \vdots \vdots \vdots \vdots \vdots \\
\downarrow \downarrow \downarrow \downarrow \downarrow \downarrow \\
\pi_m'' \rightsquigarrow_\pi \pi_{m,1} \rightsquigarrow_\pi \pi_{m,2} \rightsquigarrow_\pi \cdots \rightsquigarrow_\pi \pi^*
\]

to find the desired proof \( \pi^* \). So, if \( \pi_n' \) and \( \pi_n'' \) are normal they must be identical.

So, sequences of reductions from \( \pi \) cannot terminate in two different proofs. However, does every reduction process terminate?

**Definition 2.21 [Strongly Normalising]** A proof \( \pi \) is strongly normalising (under a reduction relation \( \rightsquigarrow \)) if and only if there is no infinite reduction sequence starting from \( \pi \).
We will prove that every proof is strongly normalising under the relation \( \leadsto \) of deleting detour formulas. To assist in talking about this, we need to make a few more definitions. First, the \textit{reduction tree}.

\textbf{Definition 2.22 [Reduction Tree]} The reduction tree (under \( \leadsto \)) of a proof \( \pi \) is the tree whose branches are the reduction sequences on the relation \( \leadsto \). So, from the root \( \pi \) we reach any proof accessible in one \( \leadsto \) step from \( \pi \). From each \( \pi' \) where \( \pi \leadsto \pi' \), we branch similarly. Each node has only finitely many successors as there are only finitely many detour formulas in a proof. For each proof \( \pi \), \( \nu(\pi) \) is the size of its reduction tree.

\textbf{Lemma 2.23 [The Size of Reduction Trees]} The reduction tree of any strongly normalising proof is finite. It follows that not only is every reduction path finite, but there is a longest reduction path.

\textit{Proof:} This is a corollary of König's Lemma, which states that every tree in which the number of immediate descendants of a node is finite (it is finitely branching), and in which every branch is finitely long, is itself finite. It follows that any strongly normalising proof not only has only finite reduction paths, it also has a longest reduction path. \hfill \Box

Now to prove that every proof is strongly normalising. To do this, we define a new property that proofs can have: of being \textit{red}. It will turn out that all \textit{red} proofs are strongly normalising. It will also turn out that all proofs are \textit{red}.

\textbf{Definition 2.24 [Red Proofs]} We define a new predicate ‘\textit{red}’ applying to proofs in the following way.

\begin{itemize}
  \item A proof of an atomic formula is \textit{red} if and only if it is strongly normalising.
  \item A proof \( \pi \) of an implication formula \( A \rightarrow B \) is \textit{red} if and only if whenever \( \pi' \) is a \textit{red} proof of type \( A \), then the proof

\begin{prooftree}
\UnaryInf \AxiomC{\pi'} \ IonicPage{\pi'} \AxiomC{A} \RightLabel{B} \RightLabel{E}
\end{prooftree}

\end{itemize}

is a \textit{red} proof of type \( B \).

We will have cause to talk often of the proof found by extending a proof \( \pi \) of \( A \rightarrow B \) and a proof \( \pi' \) of \( A \) to form the proof of \( B \) by adding an \( \rightarrow E \) step. We will write \( ' (\pi \pi') ' \) to denote this proof. If you like, you can think of it as the application of the proof \( \pi \) to the proof \( \pi' \).

Now, our aim will be twofold: to show that every \textit{red} proof is strongly normalising, and to show that every proof is \textit{red}. We start by proving the following crucial lemma:

\textbf{Lemma 2.25 [Properties of Red Proofs]} For any proof \( \pi \), the following three conditions hold:
\( c_1 \) If \( \pi \) is \textcolor{red}{\text{red}} then \( \pi \) is strongly normalisable.

\( c_2 \) If \( \pi \) is \textcolor{red}{\text{red}} and \( \pi \) reduces to \( \pi' \) in one step, then \( \pi' \) is \textcolor{red}{\text{red}} too.

\( c_3 \) If \( \pi \) is a proof not ending in \( \to \), and whenever we eliminate one indirect pair in \( \pi \) we have a \textcolor{red}{\text{red}} proof, then \( \pi \) is \textcolor{red}{\text{red}} too.

Proof: We prove this result by induction on the formula proved by \( \pi \). We start with proofs of atomic formulas.

\( c_1 \) Any \textcolor{red}{\text{red}} proof of an atomic formula is strongly normalising, by the definition of \textcolor{red}{\text{red}}.

\( c_2 \) If \( \pi \) is strongly normalising, then so is any proof to which \( \pi \) reduces.

\( c_3 \) \( \pi \) does not end in \( \to \) as it is a proof of an atomic formula. If whenever \( \pi \Rightarrow_1 \pi' \) and \( \pi' \) is \textcolor{red}{\text{red}}, since \( \pi' \) is a proof of an atomic formula, it is strongly normalising. Since any reduction path through \( \pi \) must travel through one such proof \( \pi' \), each such path through \( \pi \) terminates. So, \( \pi \) is \textcolor{red}{\text{red}}.

Now we prove the results for a proof \( \pi \) of \( A \to B \), under the assumption that \( c_1 \), \( c_2 \) and \( c_3 \) they hold for proofs of \( A \) and proofs of \( B \). We can then conclude that they hold of all proofs, by induction on the complexity of the formula proved.

\( c_1 \) If \( \pi \) is a \textcolor{red}{\text{red}} proof of \( A \to B \), consider the proof

\[
\sigma: \quad \frac{\pi}{A \to B \quad A} \quad B
\]

The assumption \( A \) is a normal proof of its conclusion \( A \) not ending in \( \to \), so \( c_3 \) applies and it is \textcolor{red}{\text{red}}. So, by the definition of \textcolor{red}{\text{red}} proofs of implication formulas, \( \sigma \) is a \textcolor{red}{\text{red}} proof of \( B \). Condition \( c_1 \) tells us that \textcolor{red}{\text{red}} proofs of \( B \) are strongly normalising, so any reduction sequence for \( \sigma \) must terminate. It follows that any reduction sequence for \( \pi \) must terminate too, since if we had a non-terminating reduction sequence for \( \pi \), we could apply the same reductions to the proof \( \sigma \). But since \( \sigma \) is strongly normalising, this cannot happen. It follows that \( \pi \) is strongly normalising too.

\( c_2 \) Suppose that \( \pi \) reduces in one step to a proof \( \pi' \). Given that \( \pi \) is \textcolor{red}{\text{red}}, we wish to show that \( \pi' \) is \textcolor{red}{\text{red}} too. Since \( \pi' \) is a proof of \( A \to B \), we want to show that for any \textcolor{red}{\text{red}} proof \( \pi'' \) of \( A \), the proof \( (\pi' \pi'') \) is \textcolor{red}{\text{red}}. But this proof is \textcolor{red}{\text{red}} since the \textcolor{red}{\text{red}} proof \( (\pi \pi'') \) reduces to \( (\pi' \pi'') \) in one step (by reducing \( \pi \) to \( \pi' \)), and \( c_2 \) applies to proofs of \( B \).

\( c_3 \) Suppose that \( \pi \) does not end in \( \to \), and suppose that all of the proofs reached from \( \pi \) in one step are \textcolor{red}{\text{red}}. Let \( \sigma \) be a \textcolor{red}{\text{red}} proof
of $A$. We wish to show that the proof $(\pi \sigma)$ is \textcolor{red}{red}. By $c_1$ for the
formula $A$, we know that $\sigma$ is strongly normalising. So, we may
reason by induction on the length of the longest reduction path
for $\sigma$. If $\sigma$ is normal (with path of length 0), then $(\pi \sigma)$ reduces in
one step only to $(\pi' \sigma)$, with $\pi'$ one step from $\pi$. But $\pi'$ is \textcolor{red}{red}
so $(\pi' \sigma)$ is too.

On the other hand, suppose $\sigma$ is not yet normal, but the result
holds for all $\sigma'$ with shorter reduction paths than $\sigma$. So, suppose
$\tau$ reduces to $(\pi \sigma')$ with $\sigma'$ one step from $\sigma$. $\sigma'$ is \textcolor{red}{red}
by the induction hypothesis $c_2$ for $A$, and $\sigma'$ has a shorter reduction
path, so the induction hypothesis for $\sigma'$ tells us that $(\pi \sigma')$ is \textcolor{red}{red}.

There is no other possibility for reduction as $\pi$ does not end in
$\rightarrow I$, so reductions must occur wholly in $\pi$ or wholly in $\sigma$, and not
in the last step of $(\pi \sigma)$.

This completes the proof by induction. The conditions $c_1$, $c_2$ and $c_3$
hold of every proof. 

Now we prove one more crucial lemma.

**Lemma 2.26 [\textcolor{red}{red} proofs ending in $\rightarrow I$]** If for each \textcolor{red}{red} proof $\sigma$ of $A$,
the proof

$$
\begin{array}{c}
\vdots \quad \sigma \\
\pi(\sigma) : \\
\vdots \quad \pi \\
\vdots \quad \sigma \\
\vdots \quad A \\
\vdots \quad B \\
\end{array}
$$

is \textcolor{red}{red}, then so is the proof

$$
\begin{array}{c}
\vdots \quad \pi \\
\vdots \quad \sigma \\
\vdots \quad \tau : \\
\vdots \quad B \\
\vdots \quad A \rightarrow B \\
\vdots \quad \rightarrow I \\
\end{array}
$$

**Proof:** We show that the $(\tau \sigma)$ is \textcolor{red}{red} whenever $\sigma$ is \textcolor{red}{red}. This will
suffice to show that the proof $\pi$ is \textcolor{red}{red}, by the definition of the predicate
\textcolor{red}{‘red’} for proofs of $A \rightarrow B$. We will show that every proof resulting
from $(\tau \sigma)$ in one step is \textcolor{red}{red}, and we will reason by induction on the
sum of the sizes of the reduction trees of $\pi$ and $\sigma$. There are three cases:

- $(\tau \sigma) \rightsquigarrow \pi(\sigma)$. In this case, $\pi(\sigma)$ is \textcolor{red}{red} by the hypothesis of the
  proof.
- $(\tau \sigma) \rightsquigarrow (\pi' \sigma)$. In this case the sum of the size of the reduction
trees of $\tau'$ and $\sigma$ is smaller, and we may appeal to the induction
hypothesis.
- $(\tau \sigma) \rightsquigarrow (\tau \sigma')$. In this case the sum of the size of the reduction
trees is $\tau$ and $\sigma'$ smaller, and we may appeal to the induction
hypothesis.
**Theorem 2.27 [All Proofs Are Red]** Every proof \( \pi \) is \textbf{red}.

**Lemma 2.28 [Red Proofs by Induction]** Given proof \( \pi \) with assumptions \( A_1, \ldots, A_n \) and any \textbf{red} proofs \( \sigma_1, \ldots, \sigma_n \) of the respective formulas \( A_1, \ldots, A_n \), it follows that the proof \( \pi(\sigma_1, \ldots, \sigma_n) \) in which each assumption \( A_i \) is replaced by the proof \( \sigma_i \) is \textbf{red}.

**Proof:** We prove this by induction on the construction of the proof.

- If \( \pi \) is an assumption \( A_1 \), the claim is a tautology (if \( \sigma_1 \) is \textbf{red}, then \( \sigma_1 \) is \textbf{red}).
- If \( \pi \) ends in \( \rightarrow E \), and is \( (\pi_1, \pi_2) \), then by the induction hypothesis \( \pi_1(\sigma_1, \ldots, \sigma_n) \) and \( \pi_2(\sigma_1, \ldots, \sigma_n) \) are \textbf{red}. Since \( \pi_1(\sigma_1, \ldots, \sigma_n) \) has type \( A \rightarrow B \), the definition of \textbf{red}ness tells us that when ever it is applied to a \textbf{red} proof the result is also \textbf{red}. So, the proof \( (\pi_1(\sigma_1, \ldots, \sigma_n), \pi_2(\sigma_1, \ldots, \sigma_n)) \) is \textbf{red}, but this is simply \( \pi(\sigma_1, \ldots, \sigma_n) \).
- If \( \pi \) ends in an application of \( \rightarrow I \). This case is dealt with by Lemma 2.26: if \( \pi \) is a proof of \( A \rightarrow B \) ending in \( \rightarrow E \), then we may assume that \( \pi' \), the proof of \( B \) from \( A \) inside \( \pi \) is \textbf{red}, so by Lemma 2.26, the result \( \pi \) is \textbf{red} too.

It follows that every proof is \textbf{red}.\[ \square \]

It follows also that every proof is strongly normalising, since all \textbf{red} proofs are strongly normalising.

### 2.5 | Proofs and \( \lambda \)-Terms

It is very tempting to think of proofs as \textit{processes} or \textit{functions} that convert the information presented in the premises into the information in the conclusion. This is doubly tempting when you look at the notation for implication. In \( \rightarrow E \) we apply something which converts \( A \) to \( B \) (a function from \( A \) to \( B \)) to something which delivers you \( A \) (from premises) into something which delivers you \( B \). In \( \rightarrow I \) if we can produce \( B \) (when supplied with \( A \), at least in the presence of other resources—the other premises) then we can (in the context of the other resources at least) convert \( A \)'s into \( B \)'s at will.

Let’s make this talk a little more precise, by making \textit{explicit} this kind of \textit{function}-talk. It will give us a new vocabulary to talk of proofs.

We start with simple notation to talk about functions. The idea is straightforward. Consider numbers, and addition. If you have a number, you can add 2 to it, and the result is another number. If you like, if \( x \) is a number then

\[
x + 2
\]

is another number. Now, suppose we don’t want to talk about a particular number, like \( 5 + 2 \) or \( 7 + 2 \) or \( x + 2 \) for any choice of \( x \), but we want to talk about the \textit{operation} or of adding two. There is a sense in
which just writing “$x + 2$” should be enough to tell someone what we mean. It is relatively clear that we are treating the “$x$” as a marker for the input of the function, and “$x + 2$” is the output. The function is the output as it varies for different values of the input. Sometimes leaving the variables there is not so useful. Consider the subtraction

$x - y$

You can think of this as the function that takes the input value $x$ and takes away $y$. Or you can think of it as the function that takes the input value $y$ and subtracts it from $x$. Or you can think of it as the function that takes two input values $x$ and $y$, and takes the second away from the first. Which do we mean? When we apply this function to the input value $5$, what is the result? For this reason, we have a way of making explicit the different distinctions: it is the $\lambda$-notation, due to Alonzo Church [14]. The function that takes the input value $x$ and returns $x + 2$ is denoted

$$\lambda x. (x + 2)$$

The function taking the input value $y$ and subtracts it from $x$ is

$$\lambda y. (x - y)$$

The function that takes two inputs and subtracts the second from the first is

$$\lambda x. \lambda y. (x - y)$$

Notice how this function works. If you feed it the input $5$, you get the output $\lambda y. (5 - y)$. We can write application of a function to its input by way of juxtaposition. The result is that

$$(\lambda x. \lambda y. (x - y) \ 5)$$

evaluates to the result $\lambda y. (5 - y)$. This is the function that subtracts $y$ from $5$. When you feed this function the input $2$ (i.e., you evaluate $(\lambda y. (5 - y) \ 2)$) the result is $5 - 2$ — in other words, $3$. So, functions can have other functions as outputs.

Now, suppose you have a function $f$ that takes two inputs $y$ and $z$, and we wish to consider what happens when you apply $f$ to a pair where the first value is the repeated as the second value. (If $f$ is $\lambda x. \lambda y. (x - y)$ and the input value is a number, then the result should be $0$.) We can do this by applying $f$ to the value $x$ twice, to get $((f x) \ x)$. But this is not a function, it is the result of applying $f$ to $x$ and $x$. If you consider this as a function of $x$ you get

$$\lambda x. ((f x) \ x)$$

This is the function that takes $x$ and feeds it twice into $f$. But just as functions can create other functions as outputs, there is no reason not to make functions take other functions as inputs. The process here was completely general — we knew nothing specific about $f$ — so the function

$$\lambda y. \lambda x. ((y \ x) \ x)$$
takes an input \( y \), and returns the function \( \lambda x.((y x) x) \). This function takes an input \( x \), and then applies \( y \) to \( x \) and then applies the result to \( x \) again. When you feed it a function, it returns the diagonal of that function.

Now, sometimes this construction does not work. Suppose we feed our diagonal function \( \lambda y.\lambda x.((y x) x) \) an input that is not a function, or that is a function that does not expect two inputs? (That is, it is not a function that returns another function.) In that case, we may not get a sensible output. One response is to bite the bullet and say that everything is a function, and that we can apply anything to anything else. We won’t take that approach here, as something becomes very interesting if we consider what happens if we consider variables (the \( x \) and \( y \) in the expression \( \lambda y.\lambda x.((y x) x) \)) to be typed. We could consider \( y \) to only take inputs which are functions of the right kind. That is, \( y \) is a function that expects values of some kind (let’s say, of type \( A \)), and when given a value, returns a function. In fact, the function it returns has to be a function that expects values of the very same kind (also type \( A \)). The result is an object (perhaps a function) of some kind or other (say, type \( B \)). In other words, we can say that the variable \( y \) takes values of type \( A \rightarrow (A \rightarrow B) \). Then we expect the variable \( x \) to take values of type \( A \). We’ll write these facts as follows:

\[
\begin{align*}
\text{y} &: A \rightarrow (A \rightarrow B) \quad \text{x} &: A
\end{align*}
\]

Now, we may put these two things together, to say derive the type of the result of applying the function \( y \) to the input value \( x \).

\[
\begin{align*}
\text{y} &: A \rightarrow (A \rightarrow B) \quad \text{x} &: A \\
(y x) &: A \rightarrow B
\end{align*}
\]

Applying the result to \( x \) again, we get

\[
\begin{align*}
\text{y} &: A \rightarrow (A \rightarrow B) \quad \text{x} &: A \\
(y x) &: A \rightarrow B \\
((y x) x) &: B
\end{align*}
\]

Then when we abstract away the particular choice of the input value \( x \), we have this

\[
\begin{align*}
\text{y} &: A \rightarrow (A \rightarrow B) \\
(x &: A) \\
(y x) &: A \rightarrow B \\
((y x) x) &: B
\end{align*}
\]

and abstracting away the choice of \( y \), we have

\[
\begin{align*}
\text{[y} &: A \rightarrow (A \rightarrow B)] \\
\text{x} &: A \\
(y x) &: A \rightarrow B \\
((y x) x) &: B
\end{align*}
\]

\[
\begin{align*}
\lambda x.((y x) x) &: A \rightarrow B
\end{align*}
\]

\[
\begin{align*}
\lambda y.\lambda x.((y x) x) &: (A \rightarrow (A \rightarrow B)) \rightarrow (A \rightarrow B)
\end{align*}
\]
so the diagonal function \( \lambda y. \lambda x. ((y \times x)) \) has type \((A \to (A \to B)) \to (A \to B)\). It takes functions of type \( A \to (A \to B) \) as input and returns an output of type \( A \to B \).

Does that process look like something you have already seen?

We may use these \( \lambda \)-terms to represent proofs. Here are the definitions. We will first think of formulas as **types**.

\[
\text{TYPE ::= ATOM | (TYPE \to TYPE)}
\]

Then, given the class of types, we can construct terms for each type.

**Definition 2.29 [Typed Simple \( \lambda \)-Terms]** The class of typed simple \( \lambda \)-terms is defined as follows:

- For each type \( A \), there is an infinite supply of variables \( x^A, y^A, z^A, w^A, x_1^A, x_2^A, \) etc.
- If \( M \) is a term of type \( A \to B \) and \( N \) is a term of type \( A \), then \((MN)\) is a term of type \( B \).
- If \( M \) is a term of type \( B \) then \( \lambda x^A.M \) is a term of type \( A \to B \).

These formation rules for types may be represented in ways familiar to those of us who care for proofs. See Figure 2.3.

\[
\begin{align*}
M &: A \to B & N &: A \\
(M \cdot N) &: B & [x &: A]^{(i)} & \vdash \vdash \\
\vdots & \vdots & M &: B & \vdash \vdash \\
\lambda x.M &: A \to B & \vdash \vdash \\
\end{align*}
\]

**Figure 2.3: Rules for \( \lambda \)-Terms**

Sometimes we write variables without superscripts, and leave the typing of the variable understood from the context. It is simpler to write \( \lambda y.\lambda x.((y \times x)) \) instead of \( \lambda y^{A \to (A \to B)}.\lambda x^{A}(y^{A \to (A \to B)}(x^A)x^A) \).

Not everything that looks like a typed \( \lambda \)-term actually is. Consider the term

\( \lambda x.(x \times x) \)

There is no such simple typed \( \lambda \)-term. Were there such a term, then \( x \) would have to both have type \( A \to B \) and type \( A \). But as things stand now, a variable can have only one type. Not every \( \lambda \)-term is a typed \( \lambda \)-term.

Now, it is clear that typed \( \lambda \)-terms stand in some interesting relationship to proofs. From any typed \( \lambda \)-term we can reconstruct a unique
proof. Take $\lambda x.\lambda y. (y \ x)$, where $y$ has type $p \rightarrow q$ and $x$ has type $p$. We can rewrite the unique formation pedigree of the term as a tree.

$$
\begin{array}{c}
[y : p \rightarrow q] \\
[x : p]
\end{array}
\quad
\begin{array}{c}
(y \ x) : q
\end{array}
\quad
\begin{array}{c}
\lambda y. (y \ x) : (p \rightarrow q) \rightarrow q
\end{array}
\quad
\begin{array}{c}
\lambda x.\lambda y. (y \ x) : p \rightarrow ((p \rightarrow q) \rightarrow q)
\end{array}
$$

and once we erase the terms, we have a proof of $p \rightarrow ((p \rightarrow q) \rightarrow q)$. The term is a compact, linear representation of the proof which is presented as a tree.

The mapping from terms to proofs is many-to-one. Each typed term constructs a single proof, but there are many different terms for the one proof. Consider the proofs

$$
\begin{array}{c}
p \rightarrow q
\end{array}
\quad
\begin{array}{c}
p \rightarrow (q \rightarrow r)
\end{array}
\quad
\begin{array}{c}
p
\end{array}
\quad
\begin{array}{c}
(q \rightarrow r)
\end{array}
$$

we can label them as follows

$$
\begin{array}{c}
x : p \rightarrow q
\end{array}
\quad
\begin{array}{c}
y : p
\end{array}
\quad
\begin{array}{c}
z : p \rightarrow (q \rightarrow r)
\end{array}
\quad
\begin{array}{c}
y : p
\end{array}
\quad
\begin{array}{c}
(x y) : q
\end{array}
\quad
\begin{array}{c}
(z y) : q \rightarrow r
\end{array}
$$

we could combine them into the proof

$$
\begin{array}{c}
z : p \rightarrow (q \rightarrow r)
\end{array}
\quad
\begin{array}{c}
y : p
\end{array}
\quad
\begin{array}{c}
x : p \rightarrow q
\end{array}
\quad
\begin{array}{c}
y : p
\end{array}
\quad
\begin{array}{c}
(z y) : q \rightarrow r
\end{array}
\quad
\begin{array}{c}
(x y) : q
\end{array}
\quad
\begin{array}{c}
(z y)(x y) : r
\end{array}
$$

but if we wished to discharge just one of the instances of $p$, we would have to have chosen a different term for one of the two subproofs. We could have chosen the variable $w$ for the first $p$, and used the following term:

$$
\begin{array}{c}
z : p \rightarrow (q \rightarrow r)
\end{array}
\quad
\begin{array}{c}
[w : p]
\end{array}
\quad
\begin{array}{c}
x : p \rightarrow q
\end{array}
\quad
\begin{array}{c}
y : p
\end{array}
\quad
\begin{array}{c}
(z w) : q \rightarrow r
\end{array}
\quad
\begin{array}{c}
(x y) : q
\end{array}
\quad
\begin{array}{c}
(z w)(x y) : r
\end{array}
\quad
\begin{array}{c}
\lambda w. (z w)(x y) : p \rightarrow r
\end{array}
$$

So, the choice of variables allows us a great deal of choice in the construction of a term for a proof. The choice of variables both does not matter (who cares if we replace $x^A$ by $y^A$) and does matter (when it comes to discharge an assumption, the formulas discharged are exactly those labelled by the particular free variable bound by $\lambda$ at that stage).

**Definition 2.30 [From Terms to Proofs and Back]** For every typed term $M$ (of type $A$), we find PROOF($M$) (of the formula $A$) as follows:

- PROOF($x^A$) is the identity proof $A$. 

---

**Natural Deduction for Conditionals** • **Chapter 2**
» If \( \text{proof}(M^A \rightarrow B) \) is the proof \( \pi_1 \) of \( A \rightarrow B \) and \( \text{proof}(N^A) \) is the proof \( \pi_2 \) of \( A \), then extend them with one \( \rightarrow E \) step into the proof \( \text{proof}(MN^B) \) of \( B \).

» If \( \text{proof}(M^B) \) is a proof \( \pi \) of \( B \) and \( x^A \) is a variable of type \( A \), then extend the proof \( \pi \) by discharging each premise in \( \pi \) of type \( A \) labelled with the variable \( x^A \). The result is the proof \( \text{proof}(\lambda x.M)^{A \rightarrow B} \) of type \( A \rightarrow B \).

Conversely, for any proof \( \pi \), we find the set \( \text{terms}(\pi) \) as follows:

» \( \text{terms}(A) \) is the set of variables of type \( A \). (Note that the term is an unbound variable, whose type is the only assumption in the proof.)

» If \( \pi_1 \) is a proof of \( A \rightarrow B \), and \( M \) (of type \( A \rightarrow B \)) is a member of \( \text{terms}(\pi_1) \), and \( N \) (of type \( A \)) is a member of \( \text{terms}(\pi_r) \), then \( (MN) \) (which is of type \( B \)) is a member of \( \text{terms}(\pi) \), where \( \pi \) is the proof found by extending \( \pi_1 \) and \( \pi_r \) by the \( \rightarrow E \) step. (Note that if the unbound variables in \( M \) have types corresponding to the assumptions in \( \pi_1 \) and those in \( N \) have types corresponding to the assumptions in \( \pi_r \), then the unbound variables in \( (MN) \) have types corresponding to the variables in \( \pi \).)

» Suppose \( \pi \) is a proof of \( B \), and we extend \( \pi \) into the proof \( \pi' \) by discharging some set (possibly empty) of instances of the formula \( A \), to derive \( A \rightarrow B \) using \( \rightarrow I \). Then, in \( M \) is a member of \( \text{terms}(\pi) \) for which a variable \( x \) labels all and only those assumptions \( A \) that are discharged in this \( \rightarrow I \) step, then \( \lambda x.M \) is a member of \( \text{terms}(\pi') \). (Notice that the free variables in \( \lambda x.M \) correspond to the remaining active assumptions in \( \pi' \).)

**Theorem 2.31 [Relating Proofs and Terms]** If \( M \in \text{terms}(\pi) \) then \( \pi = \text{proof}(M) \). Conversely, \( M' \in \text{terms}(\text{proof}(M)) \) if and only if \( M' \) is a relabelling of \( M \).

**Proof:** For the first part, we proceed by induction on the proof \( \pi \). If \( \pi \) is an atomic proof, then since \( \text{terms}(A) \) is the set of variables of type \( A \), and since \( \text{proof}(x^A) \) is the identity proof \( A \), we have the base case of the induction. If \( \pi \) is composed of two proofs, \( \pi_1 \) of \( A \rightarrow B \), and \( \pi_r \) of \( A \), joined by an \( \rightarrow E \) step, then \( M \) is in \( \text{terms}(\pi) \) if and only if \( M = (N_1N_2) \) where \( N_1 \in \text{terms}(\pi_1) \) and \( N_2 \in \text{terms}(\pi_r) \). But by the induction hypothesis, if \( N_1 \in \text{terms}(\pi_1) \) and \( N_2 \in \text{terms}(\pi_r) \), then \( \pi_1 = \text{proof}(N_1) \) and \( \pi_r = \text{proof}(N_2) \), and as a result, \( \pi = \text{proof}(M) \), as desired.

Finally, if \( \pi \) is a proof of \( B \), extended to the proof \( \pi' \) of \( A \rightarrow B \) by discharging some (possibly empty) set of instances of \( A \), then if \( M \) is in \( \text{terms}(\pi) \) if and only if \( M = \lambda x.N, N \in \text{terms}(\pi') \), and \( x \) labels those (and only those) instances of \( A \) discharged in \( \pi \). By the induction hypothesis, \( \pi' = \text{proof}(N) \). It follows that \( \pi = \text{proof}(\lambda x.N) \), since \( x \) labels all and only the formulas discharged in the step from \( \pi' \) to \( \pi \).
For the second part of the proof, if \( M' \in \text{terms}(\text{proof}(M)) \), then if \( M \) is a variable, \( \text{proof}(M) \) is an identity proof of some formula \( \Lambda \), and \( \text{terms}(\text{proof}(M)) \) is a variable with type \( \Lambda \), so the base case of our hypothesis is proved. Suppose the hypothesis holds for terms simpler than our term \( M \). If \( M \) is an application term \((N_1N_2)\), then \( \text{proof}(N_1N_2) \) ends in \( \rightarrow E \), and the two subproofs are \( \text{proof}(N_1) \) and \( (N_2) \) respectively. By hypothesis, \( \text{term}(\text{proof}(N_1)) \) is a relabelling of \( N_1 \) and \( \text{term}(\text{proof}(N_2)) \) is a relabelling of \( N_2 \), so \( \text{term}(\text{proof}(N_1N_2)) \) may only be relabelling of \( (N_1N_2) \) as well. Similarly, if \( M \) is an abstraction term \( \lambda x.N \), then \( \text{proof}(\lambda x.N) \) ends in \( \rightarrow I \) to prove some conditional \( A \rightarrow B \), and \( \text{proof}(N) \) is a proof of \( B \), in which some (possibly empty) collection of instances of \( A \) are about to be discharged. By hypothesis, \( \text{term}(\text{proof}(N)) \) is a relabelling of \( N \), so \( \text{term}(\text{proof}(\lambda x.N)) \) can only be a relabelling of \( \lambda x.N \).

The following theorem shows that the \( \lambda \)-terms of different kinds of proofs have different features.

**Theorem 2.32 [Discharge Conditions and \( \lambda \)-Terms]** \( M \) is a linear \( \lambda \)-term (a term of some linear proof) iff each \( \lambda \) expression in \( M \) binds exactly one variable. \( M \) is a relevant \( \lambda \)-term (a term of a relevant proof) iff each \( \lambda \) expression in \( M \) binds at least one variable. \( M \) is a an affine \( \lambda \)-term (a term of some affine proof) iff each \( \lambda \) expression binds at most one variable.

**Proof:** Check the definition of \( \text{proof}(M) \). If \( M \) satisfies the conditions on variable binding, \( \text{proof}(M) \) satisfies the corresponding discharge conditions. Conversely, if \( \pi \) satisfies a discharge condition, the terms in \( \text{term}(\pi) \) are the corresponding kinds of \( \lambda \)-term.

The most interesting connection between proofs and \( \lambda \)-terms is not simply this pair of mappings. It is the connection between normalisation and evaluation. We have seen how the application of a function, like \( \lambda x.(yx)x \) to an input like \( M \) is found by removing the lambda binder, and substituting the term \( M \) for each variable \( x \) that was bound by the binder. In this case, we get \( ((yM)M) \).

**Definition 2.33 [\( \beta \) Reduction]** The term \( \lambda x.MN \) is said to directly \( \beta \)-reduce to the term \( M[x := N] \) found by substituting the term \( N \) for each free occurrence of \( x \) in \( M \).

Furthermore, \( M \) \( \beta \)-reduces in one step to \( M' \) if and only if some subterm \( N \) inside \( M \) immediately \( \beta \)-reduces to \( N' \) and \( M' = M[N := N'] \). A term \( M \) is said to \( \beta \)-reduce to \( M^* \) if there is some chain \( M = M_1, \cdots, M_n = M^* \) where each \( M_i \) \( \beta \)-reduces in one step to \( M_{i+1} \).

Consider what this means for proofs. The term \( (\lambda x.MN) \) immediately \( \beta \)-reduces to \( M[x := N] \). Representing this transformation as a proof,
we have

\[
\begin{array}{c}
\frac{\[x : A\] \quad \pi_1 \quad \pi_f \quad \pi_f}{\lambda x. M : A \rightarrow B} & \Rightarrow^\beta \\
\frac{\pi_f \quad \pi_f}{N : A} & \frac{M : B}{M[x := N] : B}
\end{array}
\]

and \(\beta\)-reduction corresponds to normalisation. This fact leads immediately to the following theorem.

**Theorem 2.34 [Normalisation and \(\beta\)-Reduction]** A proof \(\text{proof}(N)\) is normal if and only if the term \(N\) does not \(\beta\)-reduce to another term. If \(N\) \(\beta\)-reduces to \(N'\) then a normalisation process sends \(\text{proof}(N)\) to \(\text{proof}(N')\).

This natural reading of normalisation as function application, and the easy way that we think of \((\lambda x. M N)\) as being identical to \(M[x := N]\) leads some to make the following claim:

If \(\pi\) and \(\pi'\) normalise to the same proof,

then \(\pi\) and \(\pi'\) are really the same proof.

We will discuss proposals for the identity of proofs in a later section.

## 2.6 | History

Gentzen’s technique for natural deduction is not the only way to represent this kind of reasoning, with introduction and elimination rules for connectives. Independently of Gentzen, the Polish logician, Stanisław Jaśkowski constructed a closely related, but different system for presenting proofs in a natural deduction style. In Jaśkowski’s system, a proof is a \textit{structured list} of formulas. Each formula in the list is either a supposition, or it follows from earlier formulas in the list by means of the rule of \textit{modus ponens} (conditional elimination), or it is proved by \textit{conditionalisation}. To prove something by conditionalisation you first make a supposition of the antecedent: at this point you start a box. The contents of a box constitute a proof, so if you want to use a formula from outside the box, you may repeat a formula into the inside. A conditionalisation step allows you to exit the box, discharging the supposition you made upon entry. Boxes can be nested, as follows:

| 1. \( A \rightarrow (A \rightarrow B) \) | Supposition |
| 2. \( A \) | Supposition |
| 3. \( A \rightarrow (A \rightarrow B) \) | 1, Repeat |
| 4. \( A \rightarrow B \) | 2, 3, Modus Ponens |
| 5. \( B \) | 2, 4, Modus Ponens |
| 6. \( A \rightarrow B \) | 2–5, Conditionalisation |
| 7. \((A \rightarrow (A \rightarrow B)) \rightarrow (A \rightarrow B)\) | 1–6, Conditionalisation |
This nesting of boxes, and repeating or reiteration of formulas to enter boxes, is the distinctive feature of Jaśkowski’s system. Notice that we could prove the formula \((A \rightarrow (A \rightarrow B)) \rightarrow (A \rightarrow B)\) without using a duplicate discharge. The formula \(A\) is used twice as a minor premise in a Modus Ponens inference (on line 4, and on line 5), and it is then discharged at line 6. In a Gentzen proof of the same formula, the assumption \(A\) would have to be made twice.

Jaśkowski proofs also straightforwardly incorporate the effects of a vacuous discharge in a Gentzen proof. We can prove \(A \rightarrow (B \rightarrow A)\) using the rules as they stand, without making any special plea for a vacuous discharge:

1. \(A\) \hspace{1cm} Supposition
2. \(B\) \hspace{1cm} Supposition
3. \(A\) \hspace{1cm} 1, Repeat
4. \(B \rightarrow A\) \hspace{1cm} 2–3, Conditionalisation
5. \(A \rightarrow (B \rightarrow A)\) \hspace{1cm} 1–4, Conditionalisation

The formula \(B\) is supposed, and it is not used in the proof that follows. The formula \(A\) on line 4 occurs after the formula \(B\) on line 3, in the subproof, but it is harder to see that it is inferred from that \(B\). Conditionalisation, in Jaśkowski’s system, colludes with reiteration to allow the effect of vacuous discharge. It appears that the “fine control” over inferential connections between formulas in proofs in a Gentzen proof is somewhat obscured in the linearisation of a Jaśkowski proof. The fact that one formula occurs after another says nothing about how that formula is inferentially connected to its forbear.

Jaśkowski’s account of proof was modified in presentation by Frederick Fitch (boxes become assumption lines to the left, and hence become somewhat simpler to draw and to typeset). Fitch’s natural deduction system gained quite some popularity in undergraduate education in logic in the 1960s and following decades in the United States [26]. Edward Lemmon’s text Beginning Logic [46] served a similar purpose in British logic education. Lemmon’s account of natural deduction is similar to this, except that it does without the need to reiterate by breaking the box.

```
1 (1) A \rightarrow (A \rightarrow B) \hspace{1cm} Assumption
2 (2) A \hspace{1cm} Assumption
1,2 (3) A \rightarrow B \hspace{1cm} 1, 2, Modus Ponens
1,2 (4) B \hspace{1cm} 2,3, Modus Ponens
1 (5) A \rightarrow B \hspace{1cm} 2, 4, Conditionalisation
(6) B \hspace{1cm} 1, 5, Conditionalisation
```

Now, line numbers are joined by assumption numbers: each formula is tagged with the line number of each assumption upon which that formula depends. The rules for the conditional are straightforward: If \(A \rightarrow B\) depends on the assumptions \(X\) and \(A\) depends on the assumptions \(Y\), then you can derive \(B\), depending on the assumptions \(X, Y\). (You should ask yourself if \(X, Y\) is the set union of the sets \(X\) and \(Y\), or the multiset union of the multisets \(X\) and \(Y\). For Lemmon, the assumption
collections are sets.) For conditionalisation, if \( B \) depends on \( X, A \), then you can derive \( A \rightarrow B \) on the basis of \( X \) alone. As you can see, vacuous discharge is harder to motivate, as the rules stand now. If we attempt to use the strategy of the Ja\'skowski proof, we are soon stuck:

\[
\begin{align*}
1 \quad (1) & \quad A & \text{Assumption} \\
2 \quad (2) & \quad B & \text{Assumption} \\
\vdots \quad (3) & \quad \vdots
\end{align*}
\]

There is no way to attach the assumption number “2” to the formula \( A \). The linear presentation is now explicitly detached from the inferential connections between formulas by way of the assumption numbers. Now the assumption numbers tell you all you need to know about the provenance of formulas. In Lemmon’s own system, you can prove the formula \( A \rightarrow (B \rightarrow A) \) but only, as it happens, by taking a detour through conjunction or some other connective.

\[
\begin{align*}
1 \quad (1) & \quad A & \text{Assumption} \\
2 \quad (2) & \quad B & \text{Assumption} \\
1,2 \quad (3) & \quad A \land B & 1,2, \text{Conjunction intro} \\
1,2 \quad (4) & \quad A & 3, \ \text{Conjunction elim} \\
1 \quad (5) & \quad B \rightarrow A & 2,4, \ \text{Conditionalisation} \\
(6) & \quad A \rightarrow (B \rightarrow A) & 1,5, \ \text{Conditionalisation}
\end{align*}
\]

This seems quite unsatisfactory, as it breaks the normalisation property. (The formula \( A \rightarrow (B \rightarrow A) \) is proved only by a non-normal proof—in this case, a proof in which a conjunction is introduced and then immediately eliminated.) Normalisation can be restored to Lemmon’s system, but at the cost of the introduction of a new rule, the rule of weakening, which says that if \( A \) depends on assumptions \( X \), then we can infer \( A \) depending on assumptions \( X \) together with another formula.

Notice that the lines in a Lemmon proof don’t just contain formulas (or formulas tagged a line number and information about how the formula was deduced). They are pairs, consisting of a formula, and the formulas upon which the formula depends. In a Gentzen proof this information is implicit in the structure of the proof. (The formulas upon which a formula depends in a Gentzen proof are the leaves in the tree above that formula that are undischarged at the moment that this formula is derived.) This feature of Lemmon’s system was not original to him. The idea of making completely explicit the assumptions upon which a formula depends had also occurred to Gentzen, and this insight is our topic for the next section.

For more information on the history of natural deduction, consult Jeffrey Pelletier’s article [59].

Linear, relevant and affine implication have a long history. Relevant implication burst on the scene through the work of Alan Anderson and Nuel Belnap in the 1960s and 1970s [1, 2], though it had precursors in the work of the Russian logician, I. E. Orlov in the 1920s [20, 56]. The idea of a proof in which conditionals could only be introduced if
the assumption for discharge was genuinely used is indeed one of the motivations for relevant implication in the Anderson–Belnap tradition. However, other motivating concerns played a role in the development of relevant logics. For other work on relevant logic, the work of Dunn [22, 24], Routley and Meyer [75], Read [66] and Mares [48] are all useful. Linear logic arose much more centrally out of proof-theoretical concerns in the work of the proof-theorist Jean-Yves Girard in the 1980s [32, 33]. A helpful introduction to linear logic is the text of Troelstra [82]. Affine logic is introduced in the tradition of linear logic as a variant on linear implication. Affine implication is quite close, however, to the implication in Łukasiewicz’s infinitely valued logic—which is slightly stronger, but shares the property of rejecting all contraction-related principles [68]. These logics are all substructural logics [21, 58, 69].

The definition of normality is due to Prawitz [62], though glimpses of the idea are present in Gentzen’s original work [28].

The λ-calculus is due to Alonzo Church [14], and the study of λ-calculi has found many different applications in logic, computer science, type theory and related fields [3, 38, 77]. The correspondence between formulas/proofs and types/terms is known as the Curry–Howard correspondence [16, 41].

2.7 | EXERCISES

Working through these exercises will help you understand the material. As with all logic exercises, if you want to deepen your understanding of these techniques, you should attempt the exercises until they are no longer difficult. So, attempt each of the different kinds of basic exercises, until you know you can do them. Then move on to the intermediate exercises, and so on. (The project exercises are not the kind of thing that can be completed in one sitting.)

BASIC EXERCISES

Q1 Which of the following formulas have proofs with no premises?

1 : p → (p → p)
2 : p → (q → q)
3 : ((p → p) → p) → p
4 : ((p → q) → p) → p
5 : ((q → q) → p) → p
6 : ((p → q) → q) → p
7 : p → (q → (q → p))
8 : (p → q) → (p → (p → q))
9 : ((q → p) → p) → ((p → q) → q)
10 : (p → q) → ((q → p) → (p → p))
11 : (p → q) → ((q → p) → (p → q))
12 : (p → q) → ((p → (q → r)) → (p → r))
13 : (q → p) → ((p → q) → ((q → p) → (p → q)))

Formula 4 is Peirce’s Law. It is a two-valued classical logic tautology.
For each formula that can be proved, find a proof that complies with the strictest discharge policy possible.

Q2 Annotate your proofs from Exercise 1 with \( \lambda \)-terms. Find a most general \( \lambda \)-term for each provable formula.

Q3 Construct a proof from \( q \to r \) to \( (q \to (p \to p)) \to (q \to r) \) using vacuous discharge. Then construct a proof of \( q \to (p \to p) \) (also using vacuous discharge). Combine the two proofs, using \( \to E \) to deduce \( q \to r \). Normalise the proof you find. Then annotate each proof with \( \lambda \)-terms, and explain the \( \beta \) reductions of the terms corresponding to the normalisation.

Then construct a proof from \( (p \to r) \to ((p \to r) \to q) \) to \( (p \to r) \to q \) using duplicate discharge. Then construct a proof from \( p \to (q \to r) \) and \( p \to q \) to \( p \to r \) (also using duplicate discharge). Combine the two proofs, using \( \to E \) to deduce \( q \). Normalise the proof you find. Then annotate each proof with \( \lambda \)-terms, and explain the \( \beta \) reductions of the terms corresponding to the normalisation.

Q4 Find types and proofs for each of the following terms.

\begin{enumerate}
\item \( \lambda x.\lambda y.x \)
\item \( \lambda x.\lambda y.\lambda z.((xz)(yz)) \)
\item \( \lambda x.\lambda y.\lambda z.(x(yz)) \)
\item \( \lambda x.\lambda y.(yx) \)
\item \( \lambda x.\lambda y.(yx)x \)
\end{enumerate}

Which of the proofs are linear, which are relevant and which are affine?

Q5 Show that there is no normal relevant proof of these formulas.

\begin{enumerate}
\item \( p \to (q \to p) \)
\item \( (p \to q) \to (p \to (r \to q)) \)
\item \( p \to (p \to p) \)
\end{enumerate}

Q6 Show that there is no normal affine proof of these formulas.

\begin{enumerate}
\item \( (p \to q) \to ((p \to (q \to r)) \to (p \to r)) \)
\item \( (p \to (p \to q)) \to (p \to q) \)
\end{enumerate}

Q7 Show that there is no normal proof of these formulas.

\begin{enumerate}
\item \( ((p \to q) \to p) \to p \)
\item \( ((p \to q) \to q) \to ((q \to p) \to p) \)
\end{enumerate}

Q8 Find a formula that can has both a relevant proof and an affine proof, but no linear proof.

INTERMEDIATE EXERCISES

Q9 Consider the following “truth tables.”
A GD₃ tautology is a formula that receives the value t in every GD₃ valuation. An Ł₃ tautology is a formula that receives the value t in every Ł₃ valuation. Show that every formula with a standard proof is a GD₃ tautology. Show that every formula with an affine proof is an Ł₃ tautology.

Q₁₀ Consider proofs that have paired steps of the form \( \rightarrow E / \rightarrow I \). That is, a conditional is eliminated only to be introduced again. The proof has a sub-proof of the form of this proof fragment:

\[
\begin{array}{c}
A \rightarrow B \quad [A]^{(1)}
\hline
B
\rightarrow E
\hline
A \rightarrow B
\rightarrow I, I
\end{array}
\]

These proofs contain redundancies too, but they may well be normal. Call a proof with a pair like this circuitous. Show that all circuitous proofs may be transformed into non-circuitous proofs with the same premises and conclusion.

Q₁₁ In Exercise 5 you showed that there is no normal relevant proof of \( p \rightarrow (p \rightarrow p) \). By normalisation, it follows that there is no relevant proof (normal or not) of \( p \rightarrow (p \rightarrow p) \). Use this fact to explain why it is more natural to consider relevant arguments with multisets of premises and not just sets of premises. (Hint: is the argument from \( p, p \rightarrow p \) to \( p \) relevantly valid?)

Q₁₂ You might think that “if . . . then . . .” is a slender foundation upon which to build an account of logical consequence. Remarkably, there is rather a lot that you can do with implication alone, as these next questions ask you to explore.

First, define \( A \checkmark B \) as follows: \( A \checkmark B \ := (A \rightarrow B) \rightarrow B \). In what way is “\( \checkmark \)” like disjunction? What usual features of disjunction are not had by \( \checkmark \)? (Pay attention to the behaviour of \( \checkmark \) with respect to different discharge policies for implication.)

Q₁₃ Provide introduction and elimination rules for \( \checkmark \) that do not involve the conditional connective \( \rightarrow \).

Q₁₄ Now consider negation. Given an atom \( p \), define the \( p \)-negation \( \neg_p A \) to be \( A \rightarrow p \). In what way is “\( \neg_p \)” like negation? What usual features of negation are not had by \( \neg_p \) defined in this way? (Pay attention to the behaviour of \( \neg \) with respect to different discharge policies for implication.)

Q₁₅ Provide introduction and elimination rules for \( \neg_p \) that do not involve the conditional connective \( \rightarrow \).
You have probably noticed that the inference from $\neg p \neg p A$ to $A$ is not, in general, valid. Define a new language $\text{cformula}$ inside $\text{formula}$ as follows:

$$\text{cformula ::= } \neg p \neg p \text{ATOM} \mid (\text{cformula} \to \text{cformula})$$

Show that $\neg p \neg p A \vdash A$ and $A \vdash \neg p \neg p A$ are valid when $A$ is a $\text{cformula}$.

Now define $A \hat{\land} B$ to be $\neg_p(A \to \neg_p B)$, and $A \hat{\lor} B$ to be $\neg p A \to B$. In what way are $A \hat{\land} B$ and $A \hat{\lor} B$ like conjunction and disjunction of $A$ and $B$ respectively? (Consider the difference between when $A$ and $B$ are $\text{formulas}$ and when they are $\text{cformulas}$.)

Show that if there is a normal relevant proof of $A \to B$ then there is an atom occurring in both $A$ and $B$.

Show that if we have two conditional connectives $\to_1$ and $\to_2$ defined using different discharge policies, then the conditionals collapse, in the sense that we can construct proofs from $A \to_1 B$ to $A \to_2 B$ and vice versa.

Explain the significance of the result of Exercise 19.

Add rules the obvious introduction rules for a conjunction connective $\otimes$ as follows:

$$\frac{A \quad B}{A \otimes B} \otimes I$$

Show that if we have the following two $\otimes E$ rules:

$$\frac{A \otimes B}{A} \otimes I_1 \quad \frac{A \otimes B}{B} \otimes I_2$$

we may simulate the behaviour of vacuous discharge. Show, then, that we may normalise proofs involving these rules (by showing how to eliminate all indirect pairs, including $\otimes I/\otimes E$ pairs).

**ADVANCED EXERCISES**

Another demonstration of the subformula property for normal proofs uses the notion of a track in a proof.

**Definition 2.35 [Track]** A sequence $A_0, \ldots, A_n$ of formula instances in the proof $\pi$ is a track of length $n + 1$ in the proof $\pi$ if and only if

- $A_0$ is a leaf in the proof tree.
- Each $A_{i+1}$ is immediately below $A_i$.
- For each $i < n$, $A_i$ is not a minor premise of an application of $\to E$.

A track whose terminus $A_n$ is the conclusion of the proof $\pi$ is said to be a track of order 0. If we have a track $t$ whose terminus $A_n$ is the minor premise of an application of $\to E$ whose conclusion is in a track of order $n$, we say that $t$ is a track of order $n + 1$.
The following annotated proof gives an example of tracks.

\[ \begin{align*}
\blacksquare A & \rightarrow (\blacksquare (D \rightarrow D) \rightarrow B) & \blacksquare \{A\}^{(2)} \rightarrow E \\
& \rightarrow (D \rightarrow D) \rightarrow B & \blacksquare \{D\}^{(1)} \rightarrow E \\
& \rightarrow B & \blacksquare \{E\} \rightarrow E \\
\blacksquare (B \rightarrow C)^{(2)} & \rightarrow (B \rightarrow C)^{(2)} \\
& \blacksquare C & \rightarrow E \\
& \blacksquare A \rightarrow C & \rightarrow E \\
& \blacksquare (B \rightarrow C)^{(2)} \rightarrow (A \rightarrow C)^{(2)} \\
\end{align*} \]

(Don’t let the fact that this proof has one track of each order 0, 1, 2 and 3 make you think that proofs can’t have more than one track of the same order. Look at this example —

\[ \begin{align*}
A & \rightarrow (B \rightarrow C) & A \\
& B \rightarrow C & B \\
& & C \\
\end{align*} \]

— it has two tracks of order 1.) The formulas labelled with \(\blacksquare\) form one track, starting with \(B \rightarrow C\) and ending at the conclusion of the proof. Since this track ends at the conclusion of the proof, it is a track of order 0. The track consisting of formulas starts at \(A \rightarrow ((D \rightarrow D) \rightarrow B)\) and ends at \(B\). It is a track of order 1, since its final formula is the minor premise in the \(\rightarrow E\) whose conclusion is \(C\), in the \(\blacksquare\) track of order 0. Similarly, the \(\blacksquare\) track is order 2 and the \(\blacksquare\) track has order 3.

For this exercise, prove the following lemma by induction on the construction of a proof.

**Lemma 2.36** In every proof, every formula is in one and only one track, and each track has one and only one order.

Then prove this lemma.

**Lemma 2.37** Let \(t : A_0, \ldots, A_n\) be a track in a normal proof. Then

a) The rules applied within the track consist of a sequence (possibly empty) of \([\rightarrow E]\) steps and then a sequence (possibly empty) of \([\rightarrow I]\) steps.

b) Every formula \(A_1\) in \(t\) is a subformula of \(A_0\) or of \(A_n\).

Now prove the subformula theorem, using these lemmas.

Q23 Consider the result of Exercise 19. Show how you might define a natural deduction system containing (say) both a linear and a standard conditional, in which there is no collapse. That is, construct a system of natural deduction proofs in which there are two conditional connectives: \(\rightarrow_1\) for linear conditionals, and \(\rightarrow_s\) for standard conditionals, such that whenever an argument is valid for a linear conditional, it is (in some appropriate sense) valid in the system you design (when \(\rightarrow\) is translated as \(\rightarrow_1\)) and whenever an argument is valid for a standard conditional, it is (in some appropriate sense) valid in the system you design (when \(\rightarrow\) is translated as \(\rightarrow_s\)). What mixed inferences (those using both \(\rightarrow_1\) and \(\rightarrow_s\)) are valid in your system?
Q24 Suppose we have a new discharge policy that is “stricter than linear.” The ordered discharge policy allows you to discharge only the rightmost assumption at any one time. It is best paired with a strict version of →E according to which the major premise (A → B) is on the left, and the minor premise (A) is on the right. What is the resulting logic like? Does it have the normalisation property?

Q25 Take the logic of Exercise 24, and extend it with another connective ←, with the rule ←E in which the major premise (B ← A) is on the right, and the minor premise (A) is on the left, and ←I, in which the leftmost assumption is discharged. Examine the connections between → and ←. Does normalisation work for these proofs? This is Lambek’s logic for syntactic types [44, 45, 53, 54].

Q26 Show that there is a way to be even stricter than the discharge policy of Exercise 24. What is the strictest discharge policy for →I, that will result in a system which normalises, provided that →E (in which the major premise is leftmost) is the only other rule for implication.

Q27 Consider the introduction rule for ⊗ given in Exercise 21. Construct an appropriate elimination rule for fusion which does not allow the simulation of vacuous (or duplicate) discharge, and for which proofs normalise.

Q28 Identify two proofs where one can be reduced to the other by way of the elimination of circuitous steps (see Exercise 10). Characterise the identities this provides among λ-terms. Can this kind if identification be maintained along with β-reduction?

PROJECT

Q29 Thoroughly and systematically explain and evaluate the considerations for choosing one discharge policy over another. This will involve looking at the different uses to which one might put a system of natural deduction, and then, relative to a use, what one might say in favour of a different policy.
SEQUENTS FOR CONJUNCTION AND DISJUNCTION

In this section we will look at a different way of thinking about deduction: Gentzen’s *sequent calculus*. The core idea is straightforward. We want to know what follows from what, so we will keep a track of facts of consequence: facts we will record in the following form:

\[ A \vdash B \]

One can read “\( A \vdash B \)” in a number of ways. You can say that \( B \) follows from \( A \), or that \( A \) entails \( B \), or that the argument from \( A \) to \( B \) is valid. The symbol used here is sometimes called the turnstile.

Once we have the notion of consequence, we can ask ourselves what properties consequence has. There are many different ways you could answer this question. The focus of this section will be a particular technique, originally due to Gerhard Gentzen. We can think of consequence—relative to a particular *language*—like this: when we want to know about the relation of consequence, we first consider each different kind of formula in the language. To make the discussion concrete, let’s consider a very simple language: the language of propositional logic with only two connectives, conjunction \( \land \) and disjunction \( \lor \). That is, we will now look at formulas expressed in the following grammar:

\[
\text{formula ::= atom | (formula } \land \text{ formula) | (formula } \lor \text{ formula)}
\]

To characterise consequence relations, we need to figure out how consequence works on the *atoms* of the language, and then how the addition of \( \land \) and \( \lor \) expands the repertoire of facts about consequence. To do this, we need to know when we can say \( A \vdash B \) when \( A \) is a conjunction, or when \( A \) is a disjunction, and when \( B \) is a conjunction, or when \( B \) is a disjunction. In other words, for each connective, we need to know when it is appropriate to infer from a formula featuring that connective, and when it is appropriate to infer to a formula featuring that connective. Another way of putting it is that we wish to know how a connective works on the left of the turnstile, and how it works on the right.

The answers for our language seem straightforward. For atomic formulas, \( p \) and \( q \), we have \( p \vdash q \) only if \( p \) and \( q \) are the same atom: so we have \( p \vdash p \) for each atom \( p \). For conjunction, we can say that if \( A \vdash B \) and \( A \vdash C \), then \( A \vdash B \land C \). That’s how we can infer to a conjunction. Inferring from a conjunction is also straightforward. We can say that \( A \land B \vdash C \) when \( A \vdash C \), or when \( B \vdash C \). For disjunction, we can reason similarly. We can say \( A \lor B \vdash C \) when \( A \vdash C \) and \( B \vdash C \).

“This scorning a turnstile wheel at her reverend helm, she sported there a tiller; and that tiller was in one mass, curiously carved from the long narrow lower jaw of her hereditary foe. The helmsman who steered by that tiller in a tempest, felt like the Tartar, when he holds back his fiery steed by clutching its jaw. A noble craft, but somehow a most melancholy! All noble things are touched with that.” — Herman Melville, *Moby Dick.*

This is a formal account of consequence. We look only at the form of propositions and not their content. For atomic propositions (those with no internal form) there is nothing upon which we could pin a claim to consequence. Thus, \( p \vdash q \) is never true, unless \( p \) and \( q \) are the same atom.
We can say $A \vdash B \lor C$ when $A \vdash B$, or when $A \vdash C$. This is inclusive disjunction, not exclusive disjunction.

You can think of these definitions as adding new material (in this case, conjunction and disjunction) to a pre-existing language. Think of the inferential repertoire of the basic language as settled (in our discussion this is very basic, just the atoms), and the connective rules are “definitional” extensions of the basic language. These thoughts are the raw materials for the development of an account of logical consequence.

3.1 | DERIVATIONS

Like natural deduction proofs, derivations involving sequents are trees. The structure is as before:

Where each position on the tree follows from those above it. In a tree, the order of the branches does not matter. These are two different ways to present the same tree:

In this case, the tree structure is at the one and the same time simpler and more complicated than the tree structure of natural deduction proofs. They are simpler, in that there is no discharge. They are more complicated, in that trees are not trees of formulas. They are trees consisting of sequents. As a result, we will call these structures derivations instead of proofs. The distinction is simple. For us, a proof is a structure in which the formulas are connected by inferential relations in a tree-like structure. A proof will go from some formulas to other formulas, via yet other formulas. Our structures involving sequents are quite different. The last sequent in a tree (the endsequent) is itself a statement of consequence, with its own antecedent and consequent (or premise and conclusion, if you prefer.) The tree derivation shows you why (or perhaps how) you can infer from the antecedent to the consequent. The rules for constructing sequent derivations are found in Figure 3.1.

Definition 3.1 [Simple Sequent Derivation] If the leaves of a tree are instances of the Id rule, and if its transitions from node to node are instances of the other rules in Figure 3.1, then the tree is said to be a simple sequent derivation.

We must read these rules completely literally. Do not presume any properties of conjunction or disjunction other than those that can be
We will see more on this in the next section.

Important to notice that these are not derivations of the commutativity of disjunction, and the second is for associativity. (It is demonstrated on the basis of the rules. We will take these rules as constituting the behaviour of the connectives $\land$ and $\lor$.

**Example 3.2** [**Example sequent derivations**] In this section, we will look at a few sequent derivations, demonstrating some simple properties of conjunction and disjunction, and the consequence relation.

The first derivations show some commutative and associative properties of conjunction and disjunction. Here is the conjunction case, with derivations to the effect that $p \land q \vdash q \land p$, and that $p \land (q \land r) \vdash (p \land q) \land r$.

Here are the cases for disjunction. The first derivation is for the commutativity of disjunction, and the second is for associativity. (It is important to notice that these are not derivations of the commutativity or associativity of conjunction or disjunction in general. They only show the commutativity and associativity of conjunction and disjunction of atomic formulas. These are not derivations of $A \land B \vdash B \land A$ (for example) since $A \vdash A$ is not an axiom if $A$ is a complex formula. We will see more on this in the next section.)

\[
\begin{align*}
& \vdash p \land p \\
& \vdash q \land q \\
& \vdash p \lor p \\
& \vdash q \lor q \\
& \vdash p \lor (q \lor r) \\
& \vdash q \lor (q \lor r) \\
& \vdash (p \lor q) \lor r \vdash p \lor (q \lor r) \\
& \vdash p \lor (q \lor r) \\
& \vdash q \lor (q \lor r) \\
& \vdash r \lor (q \lor r) \\
\end{align*}
\]
You can see that the disjunction derivations have the same structure as those for conjunction. You can convert any derivation into another (its dual) by swapping conjunction and disjunction, and swapping the left-hand side of the sequent with the right-hand side. Here are some more examples of duality between derivations. The first is the dual of the second, and the third is the dual of the fourth.

\[
\begin{align*}
\frac{p \vdash p \quad p \vdash p}{p \lor p \vdash p} & \quad \frac{p \vdash p \quad p \vdash p \quad p \vdash p}{p \land p \vdash p} \quad \frac{p \vdash p \quad p \land q \vdash p}{p \lor (p \land q) \vdash p} \quad \frac{p \vdash p \quad p \lor q \vdash p \quad p \lor q \vdash p}{p \land (p \lor q) \vdash p} \\
\end{align*}
\]

You can use derivations you have at hand, like these, as components of other derivations. One way to do this is to use the Cut rule.

\[
\begin{align*}
\frac{p \vdash p}{p \land q \vdash p} \quad \frac{p \vdash p}{p \lor q \vdash p} & \quad \frac{p \vdash p \quad p \vdash p}{p \lor (p \land q) \vdash p} \quad \frac{p \vdash p \quad p \vdash p \lor q}{p \land (p \lor q) \vdash p} \\
\end{align*}
\]

Notice, too, that each of these derivations we’ve seen so far move from less complex formulas at the top to more complex formulas, at the bottom. Reading from bottom to top, you can see the formulas decomposing into their constituent parts. This isn’t the case for all sequent derivations. Derivations that use the Cut rule can include new (more complex) material in the process of deduction. Here is an example:

\[
\begin{align*}
\frac{p \vdash p}{p \lor q \vdash p} \quad \frac{q \vdash q}{q \lor p \vdash p} \quad \frac{q \vdash q}{q \lor p \vdash q} \quad \frac{p \vdash p}{p \lor q \vdash q} \quad \frac{p \vdash p}{p \lor q \vdash q} \\
\end{align*}
\]

This derivation is a complicated way to deduce \( p \lor q \vdash p \lor q \), and it includes \( q \lor p \), which is not a subformula of any formula in the final sequent of the derivation. Reading from bottom to top, the Cut step can introduce new formulas into the derivation.

### 3.2 | Identity & Cut

The two distinctive rules in our proof system are \( \text{Id} \) and \( \text{Cut} \). These rules are not about any particular kind of formula—they are structural, governing the behaviour of derivations, no matter what the nature of the formulas flanking the turnstiles. In this section we will look at the distinctive behaviour of \( \text{Id} \) and of \( \text{Cut} \). We start with \( \text{Id} \).

**Identity**

This derivation of \( p \lor q \vdash p \lor q \) is a derivation of an identity (a sequent of the form \( A \vdash A \)). There is a more systematic way to show that
We can piece together these little derivations in order to derive any sequent of the form $A \vdash A$. For example, here is the start of derivation of $p \land (q \lor (r_1 \land r_2)) \vdash p \land (q \lor (r_1 \land r_2))$.

\[
\frac{p \vdash p}{p \land (q \lor (r_1 \land r_2)) \vdash p \land (q \lor (r_1 \land r_2))} \quad \frac{q \lor (r_1 \land r_2) \vdash q \lor (r_1 \land r_2)}{p \land (q \lor (r_1 \land r_2)) \vdash p \land (q \lor (r_1 \land r_2))}
\]

It's not a complete derivation yet, as one leaf $q \lor (r_1 \land r_2) \vdash q \lor (r_1 \land r_2)$ is not an axiom. However, we can add the derivation for it.

The derivation of $q \lor (r_1 \land r_2) \vdash q \lor (r_1 \land r_2)$ itself contains a smaller identity derivation, for $r_1 \land r_2 \vdash r_1 \land r_2$. The derivation displayed here uses shading to indicate the way the derivations are nested together. This result is general, and it is worth a theorem of its own.

**Theorem 3.3 [Identity Derivations]** For each formula $A$, the sequent $A \vdash A$ has a derivation. A derivation for $A \vdash A$ may be systematically constructed from the identity derivations for the subformulas of $A$.

**Proof:** We define $Id_A$, the identity derivation for $A$ by induction on the construction of $A$, as follows. $Id_p$ is the axiom $p \vdash p$. For complex formulas, we have

\[
Id_A \lor B : \frac{Id_A \lor B}{A \lor B \vdash A \lor B} \quad \frac{Id_B}{B \vdash A \lor B} \quad \frac{Id_A \land B}{A \land B \vdash A \land B} \quad \frac{Id_B}{A \land B \vdash A \land B} \quad \frac{Id_A}{A \lor B \vdash A \lor B} \quad \frac{Id_B}{A \land B \vdash A \land B}
\]

We say that $A \vdash A$ is derivable in the sequent system. If we think of $Id$ as a degenerate rule (a rule with no premise), then its generalisation, $Id_A$, is a derivable rule.

It might seem crazy to have a proof of identity, like $A \vdash A$ where $A$ is a complex formula. Why don’t we take $Id_A$ as an axiom? There are a few different reasons we might like to consider for taking $Id_A$ as derivable instead of one of the primitive axioms of the system.
These are part of a general story, to be explored throughout this book, of what it is to be a logical constant. These sorts of considerations have a long history [36].

THE SYSTEM IS SIMPLE: In an axiomatic theory, it is always preferable to minimise the number of primitive assumptions. Here, it’s clear that $Id_A$ is derivable, so there is no need for it to be an axiom. A system with fewer axioms is preferable to one with more, for the reason that we have reduced derivations to a smaller set of primitive notions.

THE SYSTEM IS SYSTEMATIC: In the system without $Id_A$ as an axiom, when we consider a sequent like $L \vdash R$ in order to know whether it is derived (in the absence of Cut, at least), we can ask two separate questions. We can consider $L$. If it is complex perhaps $L \vdash R$ is derivable by means of a left rule like $[\land L]$ or $[\lor L]$. On the other hand, if $R$ is complex, then perhaps the sequent is derivable by means of a right rule, like $[\land R]$ or $[\lor R]$. If both are primitive, then $L \vdash R$ is derivable by identity only. And that is it! You check the left, check the right, and there’s no other possibility. There is no other condition under which the sequent is derivable. In the presence of $Id_A$, one would have to check if $L = R$ as well as the other conditions.

THE SYSTEM PROVIDES A CONSTRAINT: In the absence of a general identity axiom, the burden on deriving identity is passed over to the connective rules. Allowing derivations of identity statements is a hurdle over which a connective rule might be able to jump, or over which it might fail. As we shall see later, this is provides a constraint we can use to sort out “good” definitions from “bad” ones. Given that the left and right rules for conjunction and disjunction tell you how the connectives are to be introduced, it would seem that the rules are defective (or at the very least, incomplete) if they don’t allow the derivation of each instance of $Id$. We will make much more of this when we consider other connectives. However, before we make more of the philosophical motivations and implications of this constraint, we will add another possible constraint on connective rules, this time to do with the other rule in our system, Cut.

CUT

Some of the nice properties of a sequent system are as a matter of fact, the nice features of derivations that are constructed without the Cut rule. Derivations constructed without Cut satisfy the subformula property.

THEOREM 3.4 [SUBFORMULA PROPERTY] If $\delta$ is a sequent derivation not containing Cut, then the formulas in $\delta$ are all subformulas of the formulas in the endsequent of $\delta$.

Proof: You can see this merely by looking at the rules. Each rule except for Cut has the subformula property.

A derivation is said to be Cut-free if it does not contain an instance of the Cut rule. Doing without Cut is good for some things, and bad for
others. In the system of proof we’re studying in this section, sequents have very many more proofs with Cut than without it.

**Example 3.5 [Derivations with or without Cut]** \( p \vdash p \lor q \) has only one Cut-free derivation, it has infinitely many derivations using Cut. You can see that there is only one Cut-free derivation with \( p \vdash p \lor q \) as the endsequent. The only possible last inference in such a derivation is \([\lor R] \), and the only possible premise for that inference is \( p \vdash p \). This completes that proof.

On the other hand, there are very many different last inferences in a derivation featuring Cut. The most trivial example is the derivation:

\[
\begin{array}{c}
p \vdash p \\
\hline
p \vdash p \lor q \\
\end{array}
\]

which contains the Cut-free derivation of \( p \vdash p \lor q \) inside it. We can nest the cuts with the identity sequent \( p \vdash p \) as deeply as we like.

\[
\begin{array}{c}
\vdash p \vdash p \\
\hline
p \vdash \vdash p \lor q \\
\end{array}
\]

\[
\begin{array}{c}
p \vdash p \lor q \\
\hline
p \vdash \vdash p \lor q \\
\end{array}
\]

\[
\begin{array}{c}
p \vdash p \lor q \\
\hline
p \vdash \vdash p \lor q \\
\end{array}
\]

\[
\begin{array}{c}
p \vdash p \lor q \\
\hline
p \vdash \vdash p \lor q \\
\end{array}
\]

However, we can construct quite different derivations of our sequent, and we involve different material in the derivation. For any formula \( A \) you wish to choose, we could implicate \( A \) (an “innocent bystander”) in the derivation as follows:

\[
\begin{array}{c}
q \vdash q \\
\hline
q \vdash q \land A \\
\hline
q \land A \vdash q \land A \\
\hline
q \land A \vdash p \lor q \\
\hline
p \lor (q \land A) \vdash p \lor q \\
\end{array}
\]

In this derivation the Cut formula \( p \lor (q \land A) \) is doing no genuine work. It is merely repeating the left formula \( p \) or the right formula \( q \).

So, using Cut makes the search for derivations rather difficult. There are very many more possible derivations of a sequent, and many more actual derivations. The search space is much more constrained if we are looking for Cut-free derivations instead. Constructing derivations, on the other hand, is easier if we are permitted to use Cut. We have very many more options for constructing a derivation, since we are able to pass through formulas “intermediate” between the desired antecedent and consequent.
Do we need to use Cut? Is there anything derivable with Cut that cannot be derived without? Take a derivation involving Cut, like this:

\[
\begin{align*}
\frac{p \vdash p}{p \land (q \land r) \vdash p} & \quad \frac{q \vdash q}{q \land r \vdash q} \quad \frac{q \land r \vdash q}{q \vdash q} \\
\qquad \quad \frac{q \land r \vdash q}{p \land (q \land r) \vdash q} & \quad \frac{q \land r \vdash q}{p \land q \vdash q} \quad \frac{q \land r \vdash q}{p \land q \vdash q} \\
\frac{p \land (q \land r) \vdash p \land q}{p \land q \vdash q} & \quad \frac{p \land q \vdash q}{p \land q \vdash q} \quad \frac{p \land q \vdash q}{p \land q \vdash q} \\
\frac{p \land (q \land r) \vdash q \lor r}{p \land (q \land r) \vdash q} & \quad \frac{p \land (q \land r) \vdash q \lor r}{p \land (q \land r) \vdash q} \\
\end{align*}
\]

This sequent \( p \land (q \land r) \vdash q \lor r \) did not have to be derived using Cut. We can eliminate the Cut-step from the derivation in a systematic way by showing that whenever we use a Cut in a derivation we could have either done without it, or used it earlier. For example in the last inference here, we did not need to leave the Cut until the last step. We could have Cut on the sequent \( p \land q \vdash q \) and left the inference to \( q \lor r \) until later:

\[
\begin{align*}
\frac{p \vdash p}{p \land (q \land r) \vdash p} & \quad \frac{q \vdash q}{q \land r \vdash q} \quad \frac{q \land r \vdash q}{q \vdash q} \\
\qquad \quad \frac{q \land r \vdash q}{p \land (q \land r) \vdash q} & \quad \frac{q \land r \vdash q}{p \land q \vdash q} \quad \frac{q \land r \vdash q}{p \land q \vdash q} \\
\frac{p \land (q \land r) \vdash p \land q}{p \land q \vdash q} & \quad \frac{p \land q \vdash q}{p \land q \vdash q} \quad \frac{p \land q \vdash q}{p \land q \vdash q} \\
\frac{p \land (q \land r) \vdash q \lor r}{p \land (q \land r) \vdash q} & \quad \frac{p \land (q \land r) \vdash q \lor r}{p \land (q \land r) \vdash q} \\
\end{align*}
\]

Now the Cut takes place on the conjunction \( p \land q \), which is introduced immediately before the application of the Cut. Notice that in this case we use the Cut to get us to \( p \land (q \land r) \vdash p \), which is one of the sequents already seen in the derivation! This derivation repeats itself. (Do not be deceived, however. It is not a general phenomenon among proofs involving Cut that they repeat themselves. The original proof did not repeat any sequents except for the axiom \( q \vdash q \).)

No, the interesting feature of this new proof is that before the Cut, the Cut formula is introduced on the right in the derivation of left sequent \( p \land (q \land r) \vdash p \land q \), and it is introduced on the left in the derivation of the right sequent \( p \land q \vdash q \).

Notice that in general, if we have a Cut applied to a conjunction which is introduced on both sides of the step, we have a shorter route to \( L \vdash R \). We can sidestep the move through \( A \land B \) to Cut on the formula \( A \), since we have \( L \vdash A \) and \( A \vdash R \).

\[
\begin{align*}
L \vdash A & \quad L \vdash B \quad A \vdash R \\
L \vdash A \land B & \quad A \land B \vdash R \\
L \vdash R & \quad \text{Cut}
\end{align*}
\]

In our example we do the same: We Cut with \( p \land (q \land r) \vdash q \) on the left and \( q \vdash q \) on the right, to get the first proof below in which the Cut moves further up the derivation. Clearly, however, this Cut is
redundant, as cutting on an identity sequent does nothing. We could eliminate that step, without cost.

\[
\begin{array}{c}
q \vdash q \\
\frac{q \land r \vdash q}{p \land (q \land r) \vdash q} \quad \frac{p \land (q \land r) \vdash q}{q \rightleftharpoons L_1} \\
\frac{q \rightleftharpoons L_2}{q \land r \vdash q} \quad \frac{q \rightleftharpoons L_2}{q \land r \vdash q} \\
\frac{q \rightleftharpoons L_2}{q \land r \vdash q} \\
\frac{p \land (q \land r) \vdash q}{q \lor r} \quad \frac{p \land (q \land r) \vdash q}{q \lor r} \\
\end{array}
\]

We have a Cut-free derivation of our concluding sequent.

As I hinted before, this technique is a general one. We may use exactly the same method to convert any derivation using Cut into a derivation without it. To do this, we will make explicit a number of the concepts we saw in this example.

**Definition 3.6 [Active and Passive Formulas]** The formulas L and R in each inference in Figure 3.1 are said to be passive in the inference (they “do nothing” in the step from top to bottom), while the other formulas are active.

A formula is active in a step in a derivation if that formula is either introduced or eliminated. The active formulas in the connective rules are the principal formula (the conjunction or disjunction introduced, below the line) or the constituents from which the principal formula is constructed. The active formulas in a Cut step are the two instances of the Cut-formula, present above the line, but absent below the line.

**Definition 3.7 [Depth of an Inference]** The depth of an inference in a derivation is the number of nodes in the sub-derivation of δ in which that inference is the last step, minus one. In other words, it is the number of sequents above the conclusion of that inference.

Now we can proceed to present the technique for eliminating Cuts from a derivation. First we show that Cuts may be moved upward. Then we show that this process will terminate in a Cut-free derivation. This first lemma is the bulk of the procedure for eliminating Cuts from derivations.

**Lemma 3.8 [Cut-Depth Reduction]** Given a derivation δ of $A \vdash C$, whose final inference is Cut, but which is otherwise Cut-free, and in which that inference has a depth of $n$, we can transform δ another derivation $δ'$ of $A \vdash C$ which is Cut-free, or in which each Cut step has a depth less than $n$.

**Proof:** Our derivation δ contains two subderivations: $δ_1$ ending in $A \vdash B$ and $δ_r$ ending in $B \vdash C$. These subderivations are Cut-free.

\[
\begin{array}{c}
\vdots δ_1 \\
A \vdash B \\
\vdots δ_r \\
B \vdash C \\
\hline
A \vdash C
\end{array}
\]
To find our new derivation, we look at the two instances of the Cut-formula B and its roles in the final inference in \( \delta_1 \) and in \( \delta_r \). We have the following two cases: either B is passive in one or other of these inferences, or it is not.

**Case 1: The Cut-formula is Passive in Either Inference**

Suppose that the formula B is passive in the last inference in \( \delta_1 \) or passive in the last inference in \( \delta_r \). For example, if \( \delta_1 \) ends in \( \land L_1 \), then we may push the Cut above it like this:

\[
\begin{align*}
\text{Before:} & \quad A_1 \vdash B \\
& \quad A_1 \land A_2 \vdash B \\
& \quad A_1 \land A_2 \vdash C \\
\text{After:} & \quad A_1 \vdash B \\
& \quad A_1 \vdash C \\
& \quad A_1 \land A_2 \vdash C
\end{align*}
\]

The resulting derivation has a Cut-depth lower by one. If, on the other hand, \( \delta_1 \) ends in \( \lor L \), we may push the Cut above that \( \lor L \) step. The result is a derivation in which we have duplicated the Cut, but we have reduced the Cut-depth more significantly, as the effect of \( \delta_1 \) is split between the two cuts.

\[
\begin{align*}
\text{Before:} & \quad A_1 \vdash B \\
& \quad A_2 \vdash B \\
& \quad A_1 \lor A_2 \vdash B \\
& \quad A_1 \lor A_2 \vdash C \\
\text{After:} & \quad A_1 \vdash B \\
& \quad A_2 \vdash B \\
& \quad A_1 \lor A_2 \vdash C
\end{align*}
\]

The other two ways in which the Cut formula could be passive are when \( \delta_2 \) ends in \( \lor R \) or \( \land R \). The technique for these is identical to the examples we have seen. The Cut passes over \( \lor R \) trivially, and it passes over \( \land R \) by splitting into two cuts. In every instance, the depth is reduced.

**Case 2: The Cut-formula is Active**

In the remaining case, the Cut-formula formula B may be assumed to be active in the last inference in both \( \delta_1 \) and in \( \delta_r \), because we have dealt with the case in which it is passive in either inference. What we do now depends on the form of the formula B. In each case, the structure of the formula B determines the final rule in both \( \delta_1 \) and \( \delta_r \).

**Case 2a: The Cut-formula is Atomic**

If the Cut-formula is an atom, then the only inference in which an atomic formula is active in the conclusion is \( \text{Id} \). In this case, the Cut is redundant.

\[
\begin{align*}
\text{Before:} & \quad p \vdash p \\
& \quad p \vdash p \\
& \quad p \vdash p \\
\text{After:} & \quad p \vdash p
\end{align*}
\]
case 2b: the Cut-formula is a conjunction. If the Cut-formula is a conjunction \( B_1 \land B_2 \), then the only inferences in which a conjunction is active in the conclusion are \( \land R \) and \( \land L \). Let us suppose that in the inference \( \land L \), we have inferred the sequent \( B_1 \land B_2 \vdash C \) from the premise sequent \( B_1 \vdash C \). In this case, it is clear that we could have Cut on \( B_1 \) instead of the conjunction \( B_1 \land B_2 \), and the Cut is shallower. The choice for \( \land L_2 \) instead of \( \land L_1 \) involves choosing \( B_2 \) instead of \( B_1 \).

\[
\begin{array}{c}
A \vdash B_1 \\
A \vdash B_2 \\
A \vdash B_1 \land B_2 \\
A \vdash C
\end{array}
\quad
\begin{array}{c}
\vdash \delta^1_r \\
\vdash \delta^2_r \\
\vdash \delta^3_r \\
A \vdash C
\end{array}
\]

\[
\begin{array}{c}
A \vdash B_1 \land B_2 \\
B_1 \land B_2 \vdash C
\end{array}
\quad
\begin{array}{c}
A \vdash B_1 \\
B_1 \vdash C
\end{array}
\]

\[
A \vdash C
\]

\[
\begin{array}{c}
A \vdash B_1 \\
B_1 \vdash C
\end{array}
\quad
\begin{array}{c}
A \vdash C
\end{array}
\]

\[
\begin{array}{c}
A \vdash C
\end{array}
\]

In every case, then, we have traded in a derivation for a derivation either without Cut or with a shallower cut.

The process of reducing Cut-depth cannot continue indefinitely, since the starting Cut-depth of any derivation is finite. At some point we find a derivation of our sequent \( A \vdash C \) with a Cut-depth of zero: We find a derivation of \( A \vdash C \) without a cut. That is,

**Theorem 3.9 [Cut Elimination]** If a sequent is derivable with Cut, it is derivable without Cut.

**Proof:** Given a derivation of a sequent \( A \vdash C \), take a Cut with no Cuts above it. This Cut has some depth, say \( n \). Use the lemma to find a derivation with lower Cut-depth. Continue until there is no Cut remaining in this part of the derivation. (The depth of each Cut decreases, so this process cannot continue indefinitely.) Keep selecting cuts in the original derivation and eliminate them one-by-one. Since there are only finitely many cuts, this process terminates. The result is a Cut-free derivation.

This result has a number of fruitful consequences, which we will consider in the next section.
3.3 | CONSEQUENCES OF CUT ELIMINATION

Corollary 3.10 [Decidability for Simple Sequents] There is an algorithm for determining whether or not a simple sequent \( A \vdash B \) is valid.

Proof: To determine whether or not \( A \vdash B \) has a simple sequent derivation, look for the finitely many different sequents from which this sequent may be derived. Repeat the process until you find atomic sequents. Atomic sequents of the form \( p \vdash p \) are derivable, and those of the form \( p \vdash q \) are not.

Here is an example:

Example 3.11 [Distribution is not Derivable] The sequent \( p \land (q \lor r) \vdash (p \land q) \lor r \) is not derivable.

Proof: Any Cut-free derivation of \( p \land (q \lor r) \vdash (p \land q) \lor r \) must end in either a \( \land L \) step or a \( \lor R \) step. Consider the two cases:

Case 1: the derivation ends with \( \land L \): Then we infer our sequent from either \( p \vdash (p \land q) \lor r \), or from \( q \lor r \vdash (p \land q) \lor r \). Neither of these are derivable. As you can see, \( p \vdash (p \land q) \lor r \) is derivable only, using \( \lor R \) from either \( p \vdash p \land q \) or from \( p \vdash r \). The latter is not derivable (it is not an axiom, and it cannot be inferred from anywhere) and the former is derivable only when \( p \vdash q \) is — and it isn’t. Similarly, \( q \lor r \vdash (p \land q) \lor r \) is derivable only when \( q \vdash (p \land q) \lor r \) is derivable, and this is only derivable when either \( q \vdash p \land q \) or when \( q \vdash r \) are derivable, and as before, neither of these are derivable either.

Case 2: the derivation ends with \( \lor R \): Then we infer our sequent from either \( p \land (q \lor r) \vdash p \land q \) or from \( p \land (q \lor r) \vdash r \). By analogous reasoning, (more precisely, by dual reasoning) neither of these sequents are derivable. So, \( p \land (q \lor r) \vdash (p \land q) \lor r \) has no Cut-free derivation, and by Theorem 3.9 it has no derivation at all.

Searching for derivations in this naïve manner is not as efficient as we can be: we don’t need to search for all possible derivations of a sequent if we know about some of the special properties of the rules of the system. For example, consider the sequent \( A \lor B \vdash C \land D \) (where \( A, B, C \) and \( D \) are possibly complex statements). This is derivable in two ways (a) from \( A \vdash C \land D \) and \( B \vdash C \land D \) by \( \lor L \) or (b) from \( A \lor B \vdash C \) and \( A \lor B \vdash D \) by \( \land R \). Instead of searching both of these possibilities, we may notice that either choice would be enough to search for a derivation, since the rules \( \lor L \) and \( \land R \) ‘lose no information’ in an important sense.

Definition 3.12 [Invertibility] A sequent rule of the form

\[
\begin{array}{c}
S_1 \ldots S_n \\
\hline S
\end{array}
\]

is invertible if and only if whenever the sequent \( S \) is derivable, so are the sequents \( S_1, \ldots, S_n \).
**Theorem 3.13 [Invertible sequent rules]** The rules $\lor L$ and $\land R$ are invertible, but the rules $\lor R$ and $\land L$ are not.

**Proof:** Consider $\lor L$. If $A \lor B \vdash C$ is derivable, then since we have a derivation of $A \vdash A \lor B$ (by $\lor R$), a use of Cut shows us that $A \vdash C$ is derivable. Similarly, since we have a derivation of $B \vdash A \lor B$, the sequent $B \vdash C$ is derivable too. So, from the conclusion $A \lor B \vdash C$ of a $\lor L$ inference, we may derive the premises. The case for $\land R$ is completely analogous.

For $\land L$, on the other hand, we have a derivation of $p \land q \vdash p$, but no derivation of the premise $q \vdash p$, so this rule is not invertible. Similarly, $p \vdash q \lor p$ is derivable, but $p \vdash q$ is not.

It follows that when searching for a derivation of a sequent, instead of searching for all of the ways that a sequent may be derived, if it may be derived from an invertible rule we can look to the premises of that rule immediately, and consider those, without pausing to check the other sequents from which our target sequent is constructed.

**Example 3.14 [Derivation search]** The sequent $(p \land q) \lor (q \land r) \vdash (p \lor r) \land p$ is not derivable. By the invertibility of $\lor L$, it is derivable only if (a) $p \land q \vdash (p \lor r) \land p$ and (b) $q \land r \vdash (p \lor r) \land p$ are both derivable. Using the invertibility of $\land R$, the sequent (b) this is derivable only if (b$_1$) $q \land r \vdash p \lor r$ and (b$_2$) $q \land r \vdash p$ are both derivable. But (b$_2$) is not derivable because $q \vdash p$ and $r \vdash p$ are underivable.

The elimination of Cut is useful for more than just limiting the search for derivations. The fact that any derivable sequent has a Cut-free derivation has other consequences. One consequence is the fact of interpolation.

**Corollary 3.15 [Interpolation for simple sequents]** If $A \vdash B$ is derivable in the simple sequent system, then there is a formula $C$ containing only atoms present in both $A$ and $B$ such that $A \vdash C$ and $C \vdash B$ are derivable.

This result tells us that if the sequent $A \vdash B$ is derivable then that consequence “factors through” a statement in the vocabulary shared between $A$ and $B$. This means that the consequence $A \vdash B$ not only relies only upon the material in $A$ and $B$ and nothing else (that is due to the availability of a Cut-free derivation) but also in some sense the derivation ‘factors through’ the material in common between $A$ and $B$. The result is a straightforward consequence of the Cut-elimination theorem. A Cut-free derivation of $A \vdash B$ provides us with an interpolant.
Proof: We prove this by induction on the construction of the derivation of \( A \vdash B \). We keep track of the interpolant with these rules:

\[
\begin{align*}
p \vdash p & : id \\
A \vdash C & : \land L_1 \\
B \vdash C & : \land L_2 \\
\vdash C_1 & : \land R \\
\vdash C_2 & : \land R \\
A \land B \vdash C & : \land R
\end{align*}
\]

We show by induction on the length of the derivation that if we have a derivation of \( L \vdash C \) and \( C \vdash R \) then \( L \vdash C \) and \( C \vdash R \) and the atoms in \( C \) present in both \( L \) and in \( R \). These properties are satisfied by the atomic sequent \( p \vdash p, \) and it is straightforward to verify them for each of the rules. 

---

**Example 3.16 [A Derivation with an Interpolant]** Take the sequent \( p \land (q \lor (r_1 \land r_2)) \vdash (q \lor r_1) \land (p \lor r_2) \). We may annotate a Cut-free derivation of it as follows:

\[
\begin{align*}
p \vdash q & : q \\
q \vdash r & : q \lor r \\
r_1 \vdash r_1 & : r_1 \\
q \lor (r_1 \land r_2) \vdash q \lor r_1 & : \lor L \\
p \vdash p & : p \\
p \lor (q \lor r_2) \vdash p \lor r_2 & : \lor L
\end{align*}
\]

Notice that the interpolant \( (q \lor r_1) \land p \) does not contain \( r_2 \), even though \( r_2 \) is present in both the antecedent and the consequent of the sequent. This tells us that \( r_2 \) is doing no ‘work’ in this derivation. Since we have

\[
p \land (q \lor (r_1 \land r_2)) \vdash (q \lor r_1) \land p, \quad (q \lor r_1) \land p \vdash (q \lor r_1) \land (p \lor r_2)
\]

We can replace the \( r_2 \) in either derivation with another statement – say \( r_3 \) – preserving the structure of each derivation. We get the more general fact:

\[
p \land (q \lor (r_1 \land r_2)) \vdash (q \lor r_1) \land (p \lor r_3)
\]

More consequences of Cut-elimination and the admissibility of the identity rules \( Id_A \) will be considered as the book goes on. Exercises 8–14 ask you to consider different possible connective rules, some of which will admit of Cut-elimination and \( Id \)-admissibility when added, and others of which which will not. In Chapter ?? we will look at reasons why this might help us demarcate definitions of a kind of properly logical concept from those which are not logical in that sense.
The idea of taking the essence of conjunction and disjunction to be expressed in these sequent rules is to take conjunction and disjunction to form what is known as a lattice. A lattice is an ordered structure in which we have for every pair of objects a greatest lower bound and a least upper bound. A greatest lower bound of \( x \) and \( y \) is something below both \( x \) and \( y \) but which is greatest among such things. A least upper bound of \( x \) and \( y \) is something above both \( x \) and \( y \) but which is the least among such things. Among statements, taking \( \vdash \) to be the ordering, \( A \land B \) is the greatest lower bound of \( A \) and \( B \) (since \( A \land B \vdash A \) and \( A \land B \vdash B \), and if \( C \vdash A \) and \( C \vdash B \) then \( C \vdash A \land B \)) and \( A \lor B \) is their least upper bound (for dual reasons).

Lattices are wonderful structures, which may be applied in many different ways, not only to logic, but in many other domains as well. Davey and Priestley’s *Introduction to Lattices and Order* [19] is an excellent way into the literature on lattices. The concept of a lattice dates to the late 19th Century in the work of Charles S. Peirce and Ernst Schröder, who independently generalised Boole’s algebra of propositional logic. Richard Dedekind’s work on ‘ideals’ in algebraic number theory was an independent mathematical motivation for the concept. Work in the area found a focus in the groundbreaking series of papers by Garrett Birkhoff, culminating in a the book *Lattice Theory* [9]. For more of the history, and for a comprehensive state of play for lattice theory and its many applications, George Grätzer’s 1978 *General Lattice Theory* [34], and especially its 2003 Second Edition [35] is a good port of call.

We will not study much algebra in this book. However, algebraic techniques find a very natural home in the study of logical systems. Helena Rasiowa’s 1974 *An Algebraic Approach to Non-Classical Logics* [65] was the first look at lattices and other structures as models of a wide range of different systems. For a good guide to why this technique is important, and what it can do, you cannot go past J. Michael Dunn and Gary Hardegree’s *Algebraic Methods in Philosophical Logic* [23].

The idea of studying derivations consisting of sequents, rather than proofs from premises to conclusions, is entirely due to Gentzen, in his groundbreaking work in proof theory. His motivation was to extend his results on normalisation from what we called the standard natural deduction system to classical logic as well as intuitionistic logic [28, 29]. To do this, it was fruitful to step back from proofs from premises \( X \) to a conclusion \( A \) to consider statements of the form ‘\( X \vdash A \)’, making explicit at each step on which premises \( X \) the conclusion \( A \) depends. Then as we will see in the next chapter, normalisation ‘corresponds’ in some sense to the elimination of *Cuts* in a derivation. One of Gentzen’s great insights was that sequents could be generalised to the form \( X \vdash Y \) to provide a uniform treatment of traditional Boolean classical logic. We will make much of this connection in the next chapter.
Gentzen’s own logic wasn’t lattice logic, but traditional classical logic (in which the distribution of conjunction over disjunction—that is, \( A \land (B \lor C) \vdash (A \land B) \lor (A \land C) \)—is valid) and intuitionistic logic. I have chosen to start with simple sequents for lattice logic for two reasons. First, it makes the procedure for the elimination of Cuts much more simple. There are fewer cases to consider and the essential shape of the argument is laid bare with fewer inessential details. Second, once we see the technique applied again and again, it will hopefully reinforce the thought that it is very general indeed. Sequents were introduced as a way of looking at an underlying proof structure. As a pluralist, I take it that there is more than one sort of underlying proof structure to examine, and so, sequents may take more than one sort of shape. Much work has been done recently on why Gentzen chose the rules he did for his sequent calculi. I have found papers by Jan von Plato \([60, 61]\) most helpful. Gentzen’s papers are available in his collected works \([30]\), and a biography of Gentzen, whose life was cut short in the Second World War, has recently been written \([49, 50]\).

### 3.5 | EXERCISES

#### BASIC EXERCISES

**Q1** Find a derivation for \( p \vdash p \land (p \lor q) \) and a derivation for \( p \lor (p \land q) \vdash p \). Then find a Cut-free derivation for \( p \lor (p \land q) \vdash p \land (p \lor q) \) and compare it with the derivation you get by joining the two original derivations with a Cut.

**Q2** Show that there is no Cut-free derivation of the following sequents

1. \( p \lor (q \land r) \vdash p \land (q \lor r) \)
2. \( p \land (q \lor r) \vdash (p \land q) \lor r \)
3. \( p \land (q \lor (p \land r)) \vdash (p \land q) \lor (p \land r) \)

**Q3** Suppose that there is a derivation of \( A \vdash B \). Let \( C(A) \) be a formula containing \( A \) as a subformula, and let \( C(B) \) be that formula with the subformula \( A \) replaced by \( B \). Show that there is a derivation of \( C(A) \vdash C(B) \). Furthermore, show that a derivation of \( C(A) \vdash C(B) \) may be systematically constructed from the derivation of \( A \vdash B \) together with the context \( C(\_\_) \) (the shape of the formula \( C(A) \) with a ‘hole’ in the place of the subformula \( A \)).

**Q4** Find a derivation of \( p \land (q \land r) \vdash (p \land q) \land r \). Find a derivation of \( (p \land q) \land r \vdash p \land (q \land r) \). Put these two derivations together, with a Cut, to show that \( p \land (q \land r) \vdash p \land (q \land r) \). Then eliminate the cuts from this derivation. What do you get?

**Q5** Do the same thing with derivations of \( p \vdash (p \land q) \lor p \) and \( (p \land q) \lor p \vdash p \). What is the result when you eliminate this cut?

**Q6** Show that (1) \( A \vdash B \land C \) is derivable if and only if \( A \vdash B \) and \( A \vdash C \) is derivable, and that (2) \( A \lor B \vdash C \) is derivable if and only if \( A \vdash C \) and
B ⊨ C are derivable. Finally, (3) when is A ∨ B ⊨ C ∧ D derivable, in terms of the derivability relations between A, B, C and D.

Q7 Under what conditions do we have a derivation of A ⊨ B when A contains only propositional atoms and disjunctions and B contains only propositional atoms and conjunctions.

Q8 Expand the system with the following rules for the propositional constants ⊥ and T.

A ⊨ T \text{ [TR]} \quad ⊥ \vdash A \quad \text{[L]}

Show that Cut is eliminable from the new system. (You can think of ⊥ and T as zero-place connectives. In fact, there is a sense in which T is a zero-place conjunction and ⊥ is a zero-place disjunction. Can you see why?)

Q9 Show that simple sequents including T and ⊥ are decidable, following Corollary 3.10 and the results of the previous question.

Q10 Show that every formula composed of just T, ⊥, ∧ and ∨ is equivalent to either T or ⊥. (What does this result remind you of?)

Q11 Prove the interpolation theorem (Corollary 3.15) for derivations involving ∧, ∨, T and ⊥.

Q12 Expand the system with rules for a propositional connective with the following rules:

\[
\begin{align*}
A & \vdash R & A \text{ tonk } B \vdash R & \text{ tonk } L \\
\vdash L & \vdash B & \vdash L & \vdash A \text{ tonk } B
\end{align*}
\]

What new things can you derive using tonk? Can you derive A tonk B ⊨ A tonk B? Is Cut eliminable for formulas involving tonk?

Q13 Expand the system with rules for a propositional connective with the following rules:

\[
\begin{align*}
A & \vdash R & A \text{ honk } B \vdash R & \text{ honk } L \\
\vdash L & \vdash A & \vdash L & \vdash A \text{ honk } B
\end{align*}
\]

What new things can you derive using honk? Can you derive A honk B ⊨ A honk B? Is Cut eliminable for formulas involving honk?

Q14 Expand the system with rules for a propositional connective with the following rules:

\[
\begin{align*}
A & \vdash R & B & \vdash R & \text{ plonk } L \\
\vdash A \text{ plonk } B & \vdash R & \vdash L & \vdash B & \text{ plonk } R
\end{align*}
\]

What new things can you derive using plonk? Can you derive A plonk B ⊨ A plonk B? Is Cut eliminable for formulas involving plonk?

INTERMEDIATE EXERCISES

Q15 Give a formal, recursive definition of the dual of a sequent, and the dual of a derivation, in such a way that the dual of the sequent p_1 ∧ (q_1 ∨ r_1) ⊨
\[(p_2 \lor q_2) \land r_2\] is the sequent \((p_2 \land q_2) \lor r_2 \vdash p_1 \lor (q_1 \land r_1)\). And then use this definition to prove the following theorem.

**Theorem 3.17 [Duality for Derivations]** A sequent \(A \vdash B\) is derivable if and only if its dual \((A \vdash B)^d\) is derivable. Furthermore, the dual of the derivation of \(A \vdash B\) is a derivation of the dual of \(A \vdash B\).

Q16 Even though the distribution sequent \(p \land (q \lor r) \vdash (p \land q) \lor r\) is not derivable (Example 3.11), some sequents of the form \(A \land (B \lor C) \vdash (A \lor B) \land C\) are derivable. Give an independent characterisation of the triples \(\langle A, B, C \rangle\) such that \(A \land (B \lor C) \vdash (A \lor B) \land C\) is derivable.

Q17 Prove the invertibility result of Theorem 3.13 without appealing to the Cut rule or to Cut-elimination. (Hint: if a sequent \(A \lor B \vdash C\) has a derivation \(\delta\), consider the instances of \(A \lor B\) ‘leading to’ the instance of \(A \lor B\) in the conclusion. How does \(A \lor B\) appear first in the derivation? Can you change the derivation in such a way as to make it derive \(A \vdash C\)? Or to derive \(B \vdash C\) instead? Prove this, and a similar result for \(\land L\)).

**Advanced Exercises**

Q18 Define a notion of reduction for simple sequent derivations parallel to the definition of reduction of natural deduction proofs in Chapter 2. Show that it is strongly normalising and that each derivation reduces to a unique Cut-free derivation.

Q19 Define terms corresponding to simple sequent derivations, in an analogy to the way that \(\lambda\)-terms correspond to natural deduction proofs for conditional formulas. For example, we may annotate each derivation with terms in the following way:

\[
p \vdash_x p \quad \frac{L \vdash_A A \vdash_g R}{L \vdash_{\text{tag}} R \quad \text{Cut}} \]

\[
\frac{A \vdash R \quad A \land B \vdash_{l[f]} R}{A \land B \vdash_{\land l_1} R} \quad \frac{B \vdash R \quad A \land B \vdash_{r[f]} R}{A \land B \vdash_{\land l_2} R} \quad \frac{L \vdash_f A \quad L \vdash_g B}{L \vdash_{f\|g} A \land B \quad \land R}
\]

where \(x\) is an atomic term (of type \(p\)), \(f\) and \(g\) are terms, \(l[\ ]\) and \(r[\ ]\) are one-place term constructors and \(\|\) is a two-place term constructor (of a kind of parallel composition), and \(o\) is a two-place term constructor (of serial composition). Define similar term constructors for the disjunction rules.

Then reducing a Cut will correspond to simplifying terms by eliminating serial composition. A Cut in which \(A \land B\) is active will take the following form of reduction:

\[(f\|g) \circ l[h] \text{ reduces to } f \circ h \quad (f\|g) \circ r[h] \text{ reduces to } g \circ h\]

Fill out all the other reduction rules for every other kind of step in the Cut-elimination argument.

Do these terms correspond to anything like computation? Do they have any other interpretation?
PROJECTS

Q20 Provide sequent formulations for logics intermediate between simple sequent logic and the logic of distributive lattices (in which \( p \land (q \lor r) \vdash (p \land q) \lor r \)). Characterise which logics intermediate between lattice logic (the logic of simple sequents) and distributive lattice logic have sequent presentations, and which do not. (This requires making explicit what counts as a logic and what counts as a sequent presentation of a logic.)
PROOFS & DERIVATIONS: TREES

The goal in this chapter is to collect together what we have learned so far into a more coherent picture. We will begin to see how natural deduction and sequent systems can be related. It seems clear that there are connections, as the normalisation theorem and the proof of the redundancy of \textit{Cut} have a similar flavour. Both result in a subformula property, and both are proved in similar ways. In this section we will see how close this connection turns out to be.

So, to connect sequent systems and natural deduction, think of a derivation of the sequent $\Gamma \vdash \Phi$ as giving instructions for how one might conclude $\Phi$ from $\Gamma$. We can think of it as providing rules to construct a proof from $\Gamma$ to $\Phi$. A proof from $\Gamma$ to $\Phi$ cannot be the same thing as a derivation: as these proofs contain sequents, not just formulas. A proof from $\Gamma$ to $\Phi$ leads from $\Gamma$ to $\Phi$, going \textit{via} other formulas on the way. A derivation, on the other hand, can be thought of as a collection of different \textit{slices} of a proof. Thinking of the rules in a sequent system, then, perhaps we can understand them as telling us about the existence (and perhaps the construction) of natural deduction proofs. For example, the step from $L \vdash \Delta$ and $L \vdash \Phi$ to $L \vdash \Delta \land \Phi$ might be seen as saying that if we have a proof from $L$ to $\Delta$ and another proof from $L$ to $\Phi$ then these may (in some way or other) be combined into a proof from $L$ to $\Delta \land \Phi$.

The story is not completely straightforward, for we have different vocabularies for derivations and proofs. By the end of this chapter we will put them together and have our first look at the logic of conjunction, disjunction, implication and negation. For now, let us focus on implication alone. Natural deduction proofs in this vocabulary can have many assumptions but always only one conclusion. This means that a natural way of connecting these arguments with sequents is to use sequents of the form $X \vdash A$ where $X$ is a collection — a multiset — and $A$ is a single formula. So this is where we will start.

4.1 | SEQUENTS FOR LINEAR CONDITIONALS

In this section we will examine \textit{linear} natural deduction, and sequent rules appropriate for it. We need rules for conditionals in a \textit{sequent} context. That is, we want rules that say when it is appropriate to introduce a conditional on the left of a sequent, and when it is appropriate to introduce one on the right. The rule for conditionals on the right seems...
The rule can be motivated like this: If \( \pi \) is a proof from \( A \) to \( B \) (with other premises \( X \), too) then we can extend it into a proof from \( X \) to \( A \to B \) by discharging the premise \( A \). We use only linear discharge, so we read this rule quite literally. \( X, A \) is the multiset containing one more instance of \( A \) than \( X \) does. We delete that one instance of \( A \) from \( X, A \), and we have the premise multiset \( X \), from which we can deduce \( A \to B \), discharging just that instance of \( A \).

The rule \( \to L \) for conditionals on the left, on the other hand, is not as straightforward as the right rule \( \to R \). Just as with our Gentzen system for \( \land \) and \( \lor \), we want a rule that introduces our connective in the left of the sequent. This means we are after a rule that indicates when it is appropriate to infer something from a conditional formula. The canonical case of inferring something from a conditional formula is by *modus ponens*. The sequent

\[
A \to B, A \vdash B
\]

should be derivable. However, this is surely not the only context in which we may introduce \( A \to B \) into the left of a sequent. We may want to infer from \( A \to B \) when the minor premise \( A \) is not an assumption of our proof, but is itself deduced from some other premise set. That is, we at least want to endorse this step:

\[
X \vdash A \\
\hline
A \to B, X \vdash B
\]

If we have a proof from \( A \) on the basis of \( X \) then adding to this proof a new assumption of \( A \to B \) will lead us to \( B \), when we add the extra step of \( \to L \). This is straightforward enough. However, we may not only think that the \( A \) has been derived from other material — we may also think that the conclusion \( B \) has already been used as a premise in another proof. It would be a shame to have to use a Cut to deduce what follows from \( B \). In other words, we should endorse this inference:

\[
X \vdash A \\
B, Y \vdash C \\
\hline
A \to B, X, Y \vdash C
\]

which tells us how we can infer from \( A \to B \). If we can infer to \( A \) and from \( B \), then adding the assumption of \( A \to B \) lets us connect the proofs. This is clearly very closely related to the *Cut* rule, but it satisfies the subformula property, as \( A \) and \( B \) remain present in the conclusion sequent. The *Cut* rule is as before, except with the modification for our new sequents. The *Cut* formula \( C \) is one of the
antecedents in the sequent $C, Y \vdash R$, and it is Cut out and replaced by whatever assumptions are required in the proof. This motivates the following four rules for derivations in this sequent calculus presented here in Figure 4.1.

$$
\begin{array}{c}
p \vdash p & \text{Id}
\end{array}
\quad
\begin{array}{c}
X \vdash C & C, Y \vdash R
\end{array}
\quad
\begin{array}{c}
\text{Cut}
\end{array}
\quad
\begin{array}{c}
X, Y \vdash R
\end{array}

\begin{array}{c}
X \vdash A
\end{array}
\quad
\begin{array}{c}
B, Y \vdash R
\end{array}
\quad
\begin{array}{c}
\rightarrow^L
\end{array}
\quad
\begin{array}{c}
X, A \vdash B
\end{array}
\quad
\begin{array}{c}
\rightarrow^R
\end{array}

\begin{array}{c}
A \rightarrow B, X, Y \vdash R
\end{array}

\begin{array}{c}
X \vdash A \rightarrow B
\end{array}

Figure 4.1: sequents for conditionals

Sequent derivations using these rules can be constructed in the usual way. A derivation is a tree, whose leaves are Id sequents, and whose transitions follow the rules. Here is an example.

$$
\begin{array}{c}
q \vdash q & r \vdash r
\end{array}
\quad
\begin{array}{c}
p \vdash p
\end{array}
\quad
\begin{array}{c}
q, q \rightarrow r \vdash r
\end{array}
\quad
\begin{array}{c}
\rightarrow^L
\end{array}
\quad
\begin{array}{c}
p \rightarrow q, q \rightarrow r, p \rightarrow r
\end{array}
\quad
\begin{array}{c}
\rightarrow^L
\end{array}
\quad
\begin{array}{c}
p \rightarrow q, q \rightarrow r \vdash p \rightarrow r
\end{array}
\quad
\begin{array}{c}
\rightarrow^R
\end{array}
\quad
\begin{array}{c}
p \rightarrow q \vdash (q \rightarrow r) \rightarrow (p \rightarrow r)
\end{array}
\quad
\begin{array}{c}
\rightarrow^R
\end{array}
$$

Derivations for linear conditionals have the same structure as the sequent derivations for conjunction and disjunction seen in the previous section: if we do not use the Cut rule, then derivations have the subformula property, as you can see by inspecting the rules $\rightarrow^L$ and $\rightarrow^R$: all formulas in sequents above the line are subformulas of those formulas occurring below the line.

**Lemma 4.1 [Subformula Property for Cut-free Derivations]** If $\delta$ is a derivation for sequents of linear conditionals, and $\delta$ does not use the rule Cut then $\delta$ has the subformula property.

Furthermore, as with simple sequents, even though Id is assumed only for atomic formulas, it is derivable for all formulas.

**Lemma 4.2 [Identity Derivations]** For each formula $A$, there is a derivation of $A \vdash A$.

**Proof:** We define it by induction on the structure of $A$. $\text{Id}(p)$ is the axiomatic sequent $p \vdash p$. Given $\text{Id}(A)$ and $\text{Id}(B)$, this is $\text{Id}(A \rightarrow B)$:

$$
\begin{array}{c}
\text{Id}(A) & \text{Id}(B)
\end{array}
\quad
\begin{array}{c}
A \vdash A & B \vdash B
\end{array}
\quad
\begin{array}{c}
A \rightarrow B, A \vdash B
\end{array}
\quad
\begin{array}{c}
\rightarrow^L
\end{array}
\quad
\begin{array}{c}
A \rightarrow B \vdash A \rightarrow B
\end{array}
\quad
\begin{array}{c}
\rightarrow^R
\end{array}
$$

§4.1 • sequents for linear conditionals

75
However, sequents for linear conditionals are more complex than our simple sequents from the last chapter. There are more positions for formulas to appear in a sequent, as many formulas may appear on the left hand side. In the rules in Figure 4.1, a formula appearing in the spots filled by p, A, B, A → B, or C are active, and the formulas in the other positions—those filled by X, Y and R—are passive.

FROM DERIVATIONS TO PROOFS AND BACK

As we’ve seen, these rules can be understood as “talking about” the natural deduction system. We can think of a derivation of the sequent X ⊢ A as a recipe for constructing a proof from X to A. We may define a mapping, giving us for each derivation δ of X ⊢ A a proof nd(δ) from X to A.

**Definition 4.3 [nd : derivations → proofs]** For any sequent derivation δ of X ⊢ A, there is a natural deduction proof nd(δ) from the premises X to the conclusion A. It is defined recursively by first choosing nd of an identity derivation, and then, given nd of simpler derivations, we define nd of a derivation extending those derivations by →L, →I, or Cut:

» If δ is an identity sequent p ⊢ p, then nd(δ) is the proof with the sole assumption p. This is a proof from p to p.

» If δ is a derivation

```
  δ'
```

then we already have the proof nd(δ') from X, A to B. The proof nd(δ), from X to A → B is the following:

```
X ⊢ A
B
A → B
```

» If δ is a derivation

```
δ_1
δ_2
```

then we already have the proofs nd(δ_1) from X to A and nd(δ_2) from B, Y to R. The proof nd(δ), from A → B, X, Y to R is the following:

```
X
```

```
A → B
A
```

```
B
```

```
Y
```

```
R
```

76 PROOFS & DERIVATIONS: TREES • CHAPTER 4
If \( \delta \) is a derivation

\[
\begin{array}{c}
\vdash \delta_3 \\
X \vdash C \\
\vdash \delta_4 \\
C, Y \vdash R
\end{array}
\]

\[
\frac{X, Y \vdash R}{\text{Cut}}
\]

then we already have the proofs \( nd(\delta_3) \) from \( X \) to \( C \) and \( nd(\delta_4) \) from \( C, Y \) to \( R \). The proof \( nd(\delta) \), from \( X, Y \) to \( R \) is the following:

\[
\begin{array}{c}
X \\
\vdash nd(\delta_3) \\
C \\
\vdash Y \\
\vdash nd(\delta_4) \\
R
\end{array}
\]

Using these rules, we may read a derivation as a *recipe* for constructing a proof. If you examine the instructions closely, you will see that we have in fact proved a stronger result, connecting normal proofs and Cut-free derivations.

**Theorem 4.4 [Normality and Cut-Freedom]** For any Cut-free derivation \( \delta \), the proof \( nd(\delta) \) is normal.

**Proof:** This can be seen in a close examination of the steps of construction. Prove it by induction on the recursive construction of \( \delta \). If \( \delta \) is an identity step, \( nd(\delta) \) is normal, so the induction hypothesis is satisfied. Notice that whenever an \( \rightarrow E \) step is added to the proof, the major premise is a new assumption in a proof with a different conclusion. Whenever an \( \rightarrow I \) step is added to the proof, the conclusion is added at the bottom of the proof, and hence, it cannot be a major premise of an \( \rightarrow E \) step, which is an assumption in that proof and not a conclusion.

The only way we could introduce an indirect pair in to \( nd(\delta) \) would be by the use of the *Cut* rule, so if \( \delta \) is Cut-free, then \( nd(\delta) \) is normal.

This figure of speech is not idle. Just as a dish can be constructed with more than one recipe, the one proof \( \pi \) may be \( nd(\delta) \) and \( nd(\delta') \) for two different recipes \( \delta \neq \delta' \).

Another way to understand this result is as follows: the connective rules of a sequent system introduce formulas involving that connective either on the *left* or the *right*. Looking at it from the point of view of a *proof*, that means that the new formula is either introduced as an *assumption* or as a *conclusion*. In this way, the new material in the proof is always built on top of the old material, and we never compose an introduction with an elimination in such a way as to have an indirect pair in a proof. The only way to introduce an indirect pair is by way of a *Cut* step.

This mapping from sequent derivations to proofs brings to light one difference between the systems as we have set up. As we have defined them, there is *no* derivation \( \delta \) such that \( nd(\delta) \) delivers the simple proof consisting of the sole assumption \( p \rightarrow q \). It would have to be a derivation of the sequent \( p \rightarrow q \vdash p \rightarrow q \), but the proof corresponding to this

\section*{§4.1 • Sequents for Linear Conditionals}
derivation is more complicated than the simple proof consisting of the assumption alone:

\[
\delta : \quad \frac{p \rightarrow q, p \vdash q}{p \rightarrow q \vdash p \rightarrow q}^R \quad \frac{p \rightarrow q \quad [p]^{(1)}}{q \rightarrow p}^E \\
\]

What are we to make of this? If we want there to be a derivation constructing the simple proof for the argument from \( p \rightarrow q \) to itself, an option is to extend the class of derivations somewhat:

\[
\delta' : \quad \frac{p \rightarrow q \vdash p \rightarrow q}{q \rightarrow p}^L \]

If \( \delta' \) is to be a derivation, we can expand the scope of the identity rule, to allow arbitrary formulas, instead of just atoms.

\[
\Lambda \vdash \Lambda \quad \text{Id}^+ \\
\]

This motivates the following distinction:

**Definition 4.5 [Liberal and strict derivations]** A strict derivation of a sequent is one in which \( \text{Id} \) is the only identity rule used. A liberal derivation is one in which \( \text{Id}^+ \) is permitted to be used.

**Lemma 4.6** If \( \delta \) is a liberal derivation, it may be extended into \( \delta^{st} \), a strict derivation of the same sequent. Conversely, a strict derivation is already a liberal derivation. A strict derivation featuring a sequent \( \Lambda \vdash \Lambda \) where \( \Lambda \) is complex, may be truncated into a liberal derivation by replacing the derivation of \( \Lambda \vdash \Lambda \) by an appeal to \( \text{Id}^+ \).

**Proof:** The results here are a matter of straightforward surgery on derivations. To transform \( \delta \) into \( \delta^{st} \), replace each appeal to \( \text{Id}^+ \) to justify \( \Lambda \vdash \Lambda \) with the identity derivation \( \text{Id}(\Lambda) \) to derive \( \Lambda \vdash \Lambda \).

Conversely, in a strict derivation \( \delta \), replace each derivation of an identity sequent \( \Lambda \vdash \Lambda \), below which there are no more identity sequents, with an appeal to \( \text{Id}^+ \) to find the smallest liberal derivation corresponding to \( \delta \).

From now on, we will focus on liberal derivations, with the understanding that we may convert liberal derivations into strict ones if the need or desire arises.

So, we have \( nd : \text{derivations} \rightarrow \text{proofs} \): This transformation also sends Cut-free derivations to normal proofs. This lends some support to the view that derivations without Cut and normal proofs are closely related, and that Cut elimination and normalisation are in some sense the same kind of process. Can we make this connection tighter? What about the reverse direction? Is there a map that takes proofs to derivations? There is, but the situation is somewhat more complicated. In the rest of this section we will see how to transform proofs into derivations, and
we will examine the way that normal proofs can be transformed into 
Cut-free derivations.

Here is one example of how the map \( nd \) is many-to-one. There are 
derivations \( \delta \neq \delta' \) such that \( nd(\delta) = nd(\delta') \). Take \( \delta \) and \( \delta' \).

\[
\begin{align*}
\delta : & \quad p \vdash p, q \vdash q \\ & \implies q \implies q, p \vdash q, p \vdash r \\ & \implies q \implies r, p \vdash q, p \vdash r \\
\delta' : & \quad q \vdash q, r \vdash r \\ & \implies q \implies q, r \vdash r \\ & \implies q \implies r, q \vdash r \\
& \implies q \implies r, q \vdash r \\
\end{align*}
\]

Applying \( nd \) to \( \delta \) and \( \delta' \), you generate the same proof \( \pi \):

\[
\begin{align*}
\pi : & \quad q \implies r \\
& \implies q \implies r \\
& \implies q \implies r
\end{align*}
\]

This means that there are at least two different ways to make the reverse
trip, from \( \pi \) to a derivation. The matter is more complicated than this.
There is another derivation \( \delta'' \), using a \( Cut \), such that \( nd(\delta'') = \pi \).

\[
\begin{align*}
\text{\textsf{Cut}}
\end{align*}
\]

So, even though \( nd \) sends \( Cut \)-free derivations to normal proofs, it also
sends some derivations with \( Cut \) to normal proofs. To understand what
is going on, we will consider two different ways to reverse the trip, to
go from a proof \( \pi \) to a (possibly liberal) derivation \( \delta \).

**Bottom-Up Construction of Derivations:** If \( \pi \) is a proof from \( X \) to
\( A \) then \( sq^b(\pi) \) is a derivation of the sequent \( X \vdash A \), defined as follows:

> If \( \pi \) is an assumption \( A \), then \( sq^b(\pi) \) is the derivation \( Id_A \).

> If \( \pi \) is a proof from \( X \) to \( A \implies B \), composed from a proof \( \pi' \) from
\( X, A \) to \( B \) by \( \text{\textsf{→}\text{\textsf{L}}} \), then take the derivation \( sq^b(\pi') \) of the sequent
\( X, A \vdash B \), and extend it with a \( \text{\textsf{→}\text{\textsf{R}}} \) step to conclude \( X \vdash A \implies B \).

\[
\begin{align*}
& \implies sq^b(\pi') \\
& X, A \vdash B \\
& X \vdash A \implies B \\
\end{align*}
\]

> If \( \pi \) is composed using \( \text{\textsf{→}\text{\textsf{E}}} \) from a proof \( \pi' \) from \( X \) to \( A \implies B \) and
another proof \( \pi'' \) from \( Y \) to \( A \), then take the derivations \( sq^b(\pi') \) of \( X \vdash A \implies B \) and \( sq^b(\pi'') \) of \( Y \vdash A \),

\[
\begin{align*}
& \implies sq^b(\pi'') \\
& Y \vdash A \\
& B \vdash B \\
& A \implies B, Y \vdash B \\
& X, Y \vdash B \\
\end{align*}
\]

These maps are ways to reverse
engineer. All we have is the finished
dish, and we are now figure out
how to put it together, piece by piece.
This definition constructs a derivation for each natural deduction proof, from the bottom to the top.

The first thing to notice about $sq^b$ is that it does not always generate a Cut-free derivation, even if the proof you start off with is normal. We always use a Cut in the translation of a $\rightarrow E$ step, whether or not the proof $\pi$ is normal. Let’s look at how this works in an example: we can construct $sq^b$ of the following normal proof:

\[
\begin{align*}
& \frac{p \rightarrow q}{\vdash [q \rightarrow r]} \quad \text{(1)} \\
& \frac{\vdash [q \rightarrow r]}{\vdash q} \rightarrow E \\
& \frac{\vdash q}{\vdash r} \rightarrow I,1 \\
& \frac{\vdash r}{\vdash p} \rightarrow I,1,2
\end{align*}
\]

We are going to construct a derivation of the sequent $p \rightarrow q \vdash (q \rightarrow r) \rightarrow (p \rightarrow r)$. We start by working back from the last step, using the definition. Since the last step of the proof is $\rightarrow I$, we use $\rightarrow R$, and have the following part of the derivation:

\[
\begin{align*}
\vdash \delta' & \\
\vdash p \rightarrow q, q \rightarrow r \vdash p \rightarrow r & \rightarrow R \\
\vdash p \rightarrow q \vdash (q \rightarrow r) \rightarrow (p \rightarrow r) & \rightarrow \rightarrow R
\end{align*}
\]

We have $\delta'$ of the form $\pi^b$ of.

\[
\begin{align*}
& \frac{p \rightarrow q}{\vdash [q \rightarrow r]} \quad \text{(1)} \\
& \frac{\vdash [q \rightarrow r]}{\vdash q} \rightarrow E \\
& \frac{\vdash q}{\vdash r} \rightarrow I,1 \\
& \frac{\vdash r}{\vdash p} \rightarrow I,1,2
\end{align*}
\]

The next step to deal with is another $\rightarrow I$, so we have

\[
\begin{align*}
\vdash \delta'' & \\
\vdash p \rightarrow q, q \rightarrow r, p \vdash r & \rightarrow R \\
\vdash p \rightarrow q, q \rightarrow r \vdash p \rightarrow r & \rightarrow R \\
\vdash p \rightarrow q \vdash (q \rightarrow r) \rightarrow (p \rightarrow r) & \rightarrow \rightarrow R
\end{align*}
\]

Now the next step is $\rightarrow E$, so the derivation will use $\rightarrow R$ and Cut.

\[
\begin{align*}
\vdash \delta''' & \\
\vdash p \rightarrow q, p \vdash q, r \vdash r & \rightarrow L \\
\vdash q \rightarrow r \vdash q \rightarrow r & \rightarrow L \\
\vdash q \rightarrow r, p \rightarrow q, p \vdash r & \rightarrow L,1 \\
\vdash q \rightarrow r, p \rightarrow q \vdash p \rightarrow r & \rightarrow \rightarrow R \\
\vdash q \rightarrow r \vdash (p \rightarrow q) \rightarrow (p \rightarrow r) & \rightarrow \rightarrow R
\end{align*}
\]
Finally, we process the last $\rightarrow E$ in the proof, to get another $\rightarrow L$ and $Cut$ step to complete the derivation.

$$
\begin{array}{c}
p \vdash p \quad q \vdash q \\
p \rightarrow q \vdash p \rightarrow q \quad p \rightarrow q, p \vdash q \\
p \rightarrow q, p \vdash \quad r \vdash r \\
n \rightarrow r \rightarrow L \\
p \rightarrow q, p \vdash r, p \rightarrow q, p \vdash r \\
p \rightarrow q, p \vdash \rightarrow R \\
p \rightarrow q, q \rightarrow r \vdash (p \rightarrow r) \\
p \rightarrow q, q \rightarrow r \rightarrow R \\
p \rightarrow q, q \rightarrow r \vdash (p \rightarrow r) \\
\end{array}
$$

This is a derivation to construct our original proof $\pi$. Notice that the proof’s $Cut$ steps are completely redundant. We applied $Cut$s to identity sequents, have no effect whatsoever. We can eliminate these, to get a much simpler cutfree derivation:

$$
\begin{array}{c}
p \vdash p \quad q \vdash q \\
p \rightarrow q, p \vdash r \quad r \vdash r \\
n \rightarrow L \\
p \rightarrow q, q \rightarrow r \vdash (p \rightarrow r) \\
p \rightarrow q, q \rightarrow r \rightarrow R \\
p \rightarrow q, q \rightarrow r \vdash (p \rightarrow r) \\
\end{array}
$$

You can check for yourself that when you apply $nd$ to this derivation, you construct the original proof. And this derivation does not use any $Cut$ steps.

So, we have transformed the proof $\pi$ into a derivation $\delta$, which contained $Cut$s, and in this case, we eliminated them. Is there a way to construct a $Cut$-free derivation in the first place? It turns out that there is. We need to construct the proof in a more subtle way than unravelling it from the bottom.

**Perimeter construction of derivations:** If we wish to generate a $Cut$-free derivation from a normal proof, the subtlety is in how we use $\rightarrow L$ to encode an $\rightarrow E$ step. We want the major premise to appear as an assumption in the natural deduction proof. That means we must defer the decoding of an $\rightarrow E$ step until the major premise is an undischarged assumption in the proof. Thankfully, we can always do this if the proof is normal.

**Lemma 4.7 [Normal Proof Structure]** Any normal proof, using the rules $\rightarrow I$ and $\rightarrow E$ alone, is either an assumption, or ends in an $\rightarrow I$ step, or contains an undischarged assumption that is the major premise of an $\rightarrow E$ step.

**Proof:** We show this by induction on the construction of the proof $\pi$. We want to show that the proof $\pi$ has the property of being either (a)
an assumption, (b) ends in an \( \rightarrow I \) step, or (c) contains an undischarged assumption that is the major premise of an \( \rightarrow E \) step. Consider how \( \pi \) is constructed.

- If \( \pi \) is an assumption, it qualifies under condition (a).
- So, suppose that \( \pi \) ends in \( \rightarrow I \). Then it qualifies under condition (b).
- Suppose that \( \pi \) ends in \( \rightarrow E \). Then \( \pi \) combines two proofs, \( \pi_1 \) ending in \( A \rightarrow B \) and \( \pi_2 \) ending in \( A \), and we compose these with an \( \rightarrow E \) step to deduce \( B \). Since \( \pi_1 \) and \( \pi_2 \) are normal, we may presume the induction hypothesis, and that either (a), (b) or (c) apply to each proof. Since the whole proof \( \pi \) is normal, we know that the proof \( \pi_1 \) cannot end in an \( \rightarrow I \) step. So, it must satisfy property (a) or property (c). If it is (c), then one of the undischarged assumptions in \( \pi_1 \) is the major premise of an \( \rightarrow E \) step, and it is undischarged in \( \pi \), and hence \( \pi \) satisfies property (c). If, on the other hand, \( \pi_1 \) satisfies property (a), then the formula \( A \rightarrow B \), the major premise of the \( \rightarrow E \) step concluding \( \pi \), is undischarged, and \( \pi \) also satisfies property (c).

Now we may define the different map \( sq^p \) ("p" for "perimeter") according to which we strip each \( \rightarrow I \) off the bottom of the proof \( \pi \), until we have no more to take, and then, instead of dealing with the \( \rightarrow E \) at the bottom of the proof, we deal with the the leftmost undischarged major premise of an \( \rightarrow E \) step, unless there is none.

**Definition 4.8 [\( sq^p \)]** If \( \pi \) is a proof from \( X \) to \( A \) then \( sq^p(\pi) \) is a derivation of the sequent \( X \vdash A \), defined as follows:

- If \( \pi \) is an assumption \( A \), then \( sq^p(\pi) \) is the identity derivation \( Id_A \).
- If \( \pi \) is a proof from \( X \) to \( A \rightarrow B \), composed from a proof \( \pi' \) from \( X,A \) to \( B \) by a conditional introduction, then take the derivation \( sq^p(\pi') \) of the sequent \( X,A \vdash B \), and extend it with a \( \rightarrow R \) step to conclude \( X \vdash A \rightarrow B \).

\[
\frac{\vdots \, sq^p(\pi')}{X,A \vdash B} \rightarrow R \\
\frac{\vdots \, X \vdash A \rightarrow B}{X \vdash A \rightarrow B}
\]

- If \( \pi \) is a proof ending in a conditional elimination, then if \( \pi \) contains an undischarged assumption that is the major premise of an \( \rightarrow E \) step, choose the leftmost one in the proof. The proof \( \pi \) will have the following form:

\[
\begin{array}{c}
\vdots \pi_3 \\
\vdots \pi_2 \\
C \rightarrow D \\
\end{array} \\
\begin{array}{c}
\vdots \pi_1 \\
D \rightarrow E \\
\end{array} \\
\begin{array}{c}
Z \\
C \\
\end{array}
\]

82 proofs & derivations: trees • chapter 4
Take the two proofs \( \pi_2 \) and \( \pi_3 \), and apply \( \text{sq}^p \) to them to find derivations \( \text{sq}^p(\pi_2) \) of \( Z \vdash C \) and \( \text{sq}^p(\pi_3) \) of \( Y, D \vdash A \). Compose these with an \( \rightarrow L \) step as follows:

\[
\begin{array}{c}
\vdash \text{sq}^p(\pi_2) \\
Z \vdash C \\
\vdash \text{sq}^p(\pi_3) \\
Y, D \vdash A \\
\hline
C \rightarrow D, Z, Y \vdash A \\
\end{array} \rightarrow L
\]

to complete the derivation for \( C \rightarrow D, Z, Y \vdash A \).

If, on the other hand, there is no major premise of an \( \rightarrow E \) step that is an undischarged assumption in \( \pi \) (in which case, \( \pi \) is not normal), use a \text{Cut} as in the last part of the definition of \( \text{sq}^b \) (the bottom-up translation) to split the proof at the final \( \rightarrow E \) step.

This transformation will send a normal proof into a \text{Cut}-free derivation, since a \text{Cut} is only used in the mapping when the source proof is not normal. We have proved the following result.

**Theorem 4.9** For each natural deduction proof \( \pi \) from \( X \) to \( A \), \( \text{sq}^p(\pi) \) is a derivation of the sequent \( X \vdash A \). Furthermore, if \( \pi \) is normal, \( \text{sq}^p(\pi) \) is \text{Cut}-free.

This construction has an important corollary for linear sequent derivations.

**Corollary 4.10** [\text{Cut is Redundant}] If \( \delta \) is a derivation of \( X \vdash A \), then there is a \text{Cut}-free derivation \( \delta' \) of \( X \vdash A \).

**Proof**: Given a proof \( \pi \), let \( \text{norm}(\pi) \) be the normalisation of \( \pi \). Then given \( \delta \), consider \( \text{sq}^p(\text{norm}(\text{nl}(\delta))) \). This is a \text{Cut} free derivation of \( X \vdash A \). The result is a \text{Cut}-free derivation.

This isn’t how we eliminated \text{Cut} with simple sequent derivations. This is a different way to prove the redundancy of \text{Cut}. Of course, we can prove the redundancy of \text{Cut} directly.

**Eliminating Cut**

The crucial steps in the proof of the elimination of \text{Cut} from linear sequent derivations are just as they were in the case of simple sequents. In any derivation in which the last step is \text{Cut}, and in which there are no other \text{Cuts}, we push this \text{Cut} upwards towards the leaves, or trade it in for a \text{Cut} on a simpler formula.

As before, the crucial distinction is whether the \text{Cut} formula is active in both sequents in \text{Cut} step, or passive in either one. Consider the case in which the \text{Cut} formula is active.
Cut formula active on both sides: In this case the derivation is as follows:

\[
\begin{array}{c}
\delta_1 \\
X, A \vdash B \\
\end{array}
\begin{array}{c}
\delta_2 \\
Y \vdash A \\
\end{array}
\begin{array}{c}
\delta_2' \\
B, Z \vdash C \\
\end{array}
\begin{array}{c}
X \vdash A \rightarrow B \\
\rightarrow^R \\
A \rightarrow B, Y, Z \vdash C \\
\rightarrow^L \\
X, Y, Z \vdash C \\
\end{array}
\]

The Cut on A → B may be traded in for two simpler cuts: one on A and the other on B.

\[
\begin{array}{c}
\delta_2 \\
Y \vdash A \\
\end{array}
\begin{array}{c}
\delta_1 \\
X, A \vdash B \\
\end{array}
\begin{array}{c}
\delta_2' \\
B, Z' \vdash C \\
\end{array}
\begin{array}{c}
\text{Cut} \\
X, Y \vdash B \\
\end{array}
\begin{array}{c}
\text{Cut} \\
X, Y, Z \vdash C \\
\end{array}
\]

Cut formula passive on one side: There are more cases to consider here, as there are more ways the Cut formula can be passive in a derivation. The Cut formula can be passive by occurring in X, Y, or R in either →L or →R:

\[
\begin{array}{c}
X \vdash A, B, Y \vdash R \\
\rightarrow^L \\
A \rightarrow B, X, Y \vdash R \\
\end{array}
\begin{array}{c}
X, A \vdash B \\
\rightarrow^R \\
X \vdash A \rightarrow B \\
\end{array}
\]

So, let’s mark all of the different places that a Cut formula could occur passively in each of these inferences. The inferences in Figure 4.2 mark the four different locations of a Cut formula with C.

\[
\begin{array}{c}
X', C \vdash A \\
\rightarrow^L \\
A \rightarrow B, X', C, Y \vdash R \\
\end{array}
\begin{array}{c}
X \vdash A, B, Y' \vdash C \\
\rightarrow^L \\
A \rightarrow B, X, Y' \vdash C \\
\end{array}
\begin{array}{c}
X \vdash A, B, Y' \vdash C \\
\rightarrow^L \\
A \rightarrow B, X, Y' \vdash C \\
\end{array}
\begin{array}{c}
X', C, A \vdash B \\
\rightarrow^R \\
X', C \vdash A \rightarrow B \\
\end{array}
\]

Figure 4.2: four positions for passive cut formulas

In each case we want to show that a Cut on the presented formula C occurring in the lower sequent could be pushed up to occur on the upper sequent instead. That is, that we could permute the Cut step and this inference.

Start with the first example. We want to swap the Cut and the →L step in this fragment of the derivation:

\[
\begin{array}{c}
\delta_2 \\
X', C \vdash A \\
\end{array}
\begin{array}{c}
\delta_2' \\
B, Y \vdash R \\
\end{array}
\begin{array}{c}
\delta_1 \\
Z \vdash C \\
\end{array}
\begin{array}{c}
\rightarrow^L \\
A \rightarrow B, X', C, Y \vdash R \\
\end{array}
\begin{array}{c}
\text{Cut} \\
A \rightarrow B, X', Z, Y \vdash R \\
\end{array}
\]

84 PROOFS & DERIVATIONS: TREES • CHAPTER 4
But the swap is easy to achieve. We do this:

\[
\begin{array}{c}
\vdash \gamma_1 \\
\vdash \gamma_2 \\
Z \vdash C \\
X', C \vdash A \\
\text{Cut} \\
X', Z \vdash A \\
\rightarrow L \\
A \rightarrow B, X', Z, Y \vdash R
\end{array}
\]

The crucial feature of the rule \( \rightarrow L \) that allows this swap is that it is closed under the substitution of formulas in passive position. We could replace the formula \( C \) by \( Z \) in the inference without disturbing it. The result is still an instance of \( \rightarrow L \). The case of the second position in \( \rightarrow L \) is similar. The \text{Cut} replaces the \( C \) in \( A \rightarrow B, X, Y', C \vdash R \) by \( Z \), and we could have just as easily deduced this sequent by cutting on the \( C \) in the premise sequent \( B, Y', C \vdash R \), and then inferring with \( \rightarrow L \).

The final case where the passive \text{Cut} formula is on the left of the sequent is in the \( \rightarrow R \) inference. We have

\[
\begin{array}{c}
\vdash \gamma_2 \\
\vdash \gamma_1 \\
X', C, A \vdash B \\
Z \vdash C \\
\text{Cut} \\
X', Z \vdash A \rightarrow B
\end{array}
\]

and again, we could replace the \( C \) in the \( \rightarrow R \) step by \( Z \) and still have an instance of the same rule. We permute the \text{Cut} and the \( \rightarrow R \) step to get

\[
\begin{array}{c}
\vdash \gamma_1 \\
\vdash \gamma_2 \\
Z \vdash C \\
X', C, A \vdash B \\
\text{Cut} \\
X', Z \vdash A \rightarrow B
\end{array}
\]

a proof of the same endsequent, in which the \text{Cut} is higher. The only other case is for an \( \rightarrow L \) step in which the \text{Cut} formula \( C \) is on the right of the turnstile. This is slightly more complicated. We have

\[
\begin{array}{c}
\vdash \gamma_1 \\
\vdash \gamma_2 \\
X \vdash A \\
B, Y \vdash C \\
\rightarrow L \\
A \rightarrow B, X, Y \vdash C \\
Z, C \vdash D \\
\text{Cut} \\
Z, A \rightarrow B, X, Y \vdash D
\end{array}
\]

In this case we can permute the \text{Cut} and the \( \rightarrow L \) step:

\[
\begin{array}{c}
\vdash \gamma_1 \\
\vdash \gamma_2 \\
X \vdash A \\
B, Y \vdash C \\
\rightarrow L \\
Z, B, Y \vdash D \\
Z, C \vdash D \\
\text{Cut} \\
Z, A \rightarrow B, X, Y \vdash D
\end{array}
\]

Here, we have taken the \( C \) in the step

\[
\begin{array}{c}
X \vdash A \\
B, Y \vdash C \\
A \rightarrow B, X, Y \vdash C
\end{array}
\]

\[\S 4.1 \cdot \text{SEQUENTS FOR LINEAR CONDITIONALS} \]
and **Cut** on it. In this case, it does not simply mean replacing the **C** by another formula, or even by a multiset of formulas. Instead, when you **Cut** with the sequent \( Z, C \vdash D \), it means that you replace the **C** by **D** and you add \( Z \) to the left side of the sequent. So, we make the following transformation in the \( \rightarrow L \) step:

\[
\frac{X \vdash A, B, Y \vdash C}{A \rightarrow B, X, Y \vdash C} \rightarrow L \\
\frac{X \vdash A, B, Y, Z \vdash D}{A \rightarrow B, X, Y, Z \vdash D} \rightarrow L
\]

The result is also a \( \rightarrow L \) step, in which the \( C \) is replaced by \( D \), together with the extra antecedent material \( Z \).

We can see the features of \( \rightarrow L \) and \( \rightarrow R \) rules that allow the permutation with **Cut**. They are **preserved under cuts on formulas in passive position**. If you **Cut** on a formula in passive position in the endsequent of the rule, then find the corresponding formula in the to subsequent of the rule, and **Cut** on it. The resulting inference is also an instance of the same rule. We have proved the following lemma:

**Lemma 4.11 [Cut-depth reduction]** Given a derivation \( \delta \) of \( X \vdash A \), whose final inference is **Cut**, which is otherwise **Cut**-free, and in which that inference has a depth of \( n \), we may construct another derivation of \( X \vdash C \) in which each **Cut** on \( C \) has a depth less than \( n \).

The only step for which the depth reduction might be in doubt is in the case where the **Cut** formula is active on both sides. Before, we have

\[
\frac{\begin{array}{l}
\delta_1 \\
X, A \vdash B
\end{array}}{X \vdash A \rightarrow B} \rightarrow R \quad \frac{\begin{array}{l}
\delta_2 \\
Y \vdash A, B, Z \vdash C
\end{array}}{A \rightarrow B, Y, Z \vdash C} \rightarrow L
\]

and the depth of the **Cut** is \( |\delta_1| + 1 + |\delta_2| + |\delta'_2| + 1 \). After pushing the **Cut** up we have:

\[
\frac{\begin{array}{l}
\delta_2 \\
Y \vdash A
\end{array}}{X, A \vdash B} \quad \frac{\begin{array}{l}
\delta_1 \\
X, A \vdash B
\end{array}}{Cut} \quad \frac{\begin{array}{l}
\delta'_2 \\
B, Z \vdash C
\end{array}}{X, Y, Z \vdash C}
\]

The depth of the first **Cut** is \( |\delta_2| + |\delta_1| \) (which is significantly shallower than depth of the previous cut), and the depth of the second is \( |\delta_2| + |\delta_1| + 1 + |\delta'_2| \) (which is shallower by one). So, we have another proof of the **Cut** elimination theorem, directly eliminating cuts in proofs by pushing them up until they disappear.

**Theorem 4.12 [Cut elimination for linear sequents]** If \( \delta \) is a derivation of \( X \vdash A \), using **Cuts**, it may be converted into a derivation \( \delta' \) of \( X \vdash A \) in which **Cut** is not used.
4.2 | STRUCTURAL RULES

What about non-linear proofs? If we allow vacuous discharge, or duplicate discharge, we must modify the rules of the sequent system in some manner. The most straightforward possibility is to change the rules for \( \rightarrow R \), as it is the rule \( \rightarrow I \) that varies in application when we use different policies for discharge. The most direct modification would be this:

\[
\frac{X \vdash B}{X - A \vdash A \rightarrow B}^{R^-}
\]

where \( X - A \) is a multiset found by deleting instances of \( A \) from \( X \). Its treatment depends on the discharge policy in place:

- In linear discharge, \( X - A \) is the multiset \( X \) with one instance of \( A \) deleted. (If \( X \) does not contain \( A \), there is no multiset \( X - A \).)
- In relevant discharge, \( X - A \) is a multiset \( X \) with one or more instances of \( A \) deleted. (If \( X \) contains more than one instance of \( A \), then there are different multisets which can count as \( X - A \): it is not a function of \( X \) and \( A \).)
- In affine discharge, \( X - A \) is a multiset \( X \) with at most one instance of \( A \) deleted. (Now, there is always a multiset \( X - A \) for any choice of \( X \) and \( A \). There are two choices, if \( X \) actually contains \( A \), delete it or not.)
- In standard discharge, \( X - A \) is a multiset \( X \) with any number (including zero) of instances of \( A \) deleted.

The following derivations give examples of the new rule.

In the first derivation, we discharge two instances of \( p \), so this is a relevant sequent derivation, but not a linear (or affine) derivation. In the second derivation, the last \( \rightarrow R \) step is linear, but the first is not: it discharges a nonexistent instance of \( r \).

These rules match our natural deduction system very well. However, they have undesirable properties. The rules for implication vary from system to system. However, the features of the system do not actually involve implication alone: they dictate the structural properties of deduction. Here are two examples. Allowing for vacuous discharge, if the argument from \( X \) to \( B \) is valid, so is the argument from \( X, A \) to \( B \).

\[
\begin{align*}
X & \\
\vdots & \\
B & \\
\frac{A \rightarrow B}{A}^{\rightarrow I} & \\
A & \\
\frac{B}{B} &
\end{align*}
\]
In other words, if we have a derivation for $X \vdash B$, then we also should have a derivation for $X, A \vdash B$. We do, if we go through $A \rightarrow B$ and a Cut.

\[
\begin{array}{c}
\frac{X \vdash B}{X \vdash A \rightarrow B} \quad \frac{A \vdash A \quad B \vdash B}{A \rightarrow B, A \vdash B} \quad \text{Cut}
\end{array}
\]

So clearly, if we allow vacuous discharge, the step from $X \vdash B$ to $X, A \vdash B$ is justified. Instead of requiring the dodgy move through $A \rightarrow B$, we may allow it the addition of an antecedent as a primitive rule.

\[
\frac{X \vdash B}{X, A \vdash B} \quad \text{K}
\]

With $K$ and $W$ we may use the old $\rightarrow R$ rule and ‘factor out’ the different behaviour of discharge:

\[
\begin{array}{c}
p \vdash p \quad q \vdash q \quad q \vdash q \quad q, r \vdash q \\
p \vdash p \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q) \vdash p \rightarrow q \\
q \rightarrow q \\q, r \vdash q \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q) \vdash p \rightarrow q \\
q \rightarrow q \\q, r \vdash q. \rightarrow R \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q) \vdash p \rightarrow q \\
q \rightarrow q \\q, r \vdash q. \rightarrow R \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q), p, p \vdash q \\p \rightarrow (p \rightarrow q) \vdash p \rightarrow q \\
q \rightarrow q \\q, r \vdash q. \rightarrow R
\end{array}
\]

The presence of structural rules does not interfere with the connection between proofs and derivations. However, we must do a little more work when reading sequent derivations as recipes for constructing proofs.

**Definition 4.13 [nd for derivations using structural rules]** Given a derivation $\delta$ of $X \vdash A$ using structural rules, we may construct a proof $nd(\delta)$, of the conclusion $A$, from the premises $X'$, where, if contraction is used, $X'$ may contain formulas in $X$ repeated more times, and if weakening is used, $X'$ may contain fewer formulas than in $X$. Furthermore, if $\delta$ is Cut-free, $nd(\delta)$ is normal.

The hedging on the premises is for the following reason. Given weakening, $p, q \vdash p$ is derivable without using Cut. There is no normal
proof using \( p \) and \( q \), to get \( p \). (If we are to use vacuous discharge, there must be a conditional as a result. But if the proof is normal, it has no formulas other than \( p \) and \( q \).) Given contraction, \( p \rightarrow (p \rightarrow q), p \vdash p \) is derivable, but there is no normal proof of \( p \) from \( p \rightarrow (p \rightarrow q) \) that uses only one instance of \( p \) as a premise.

So, how are we to construct \( nd(\delta) \) from \( \delta \)? We do it as we did on page 76, supplementing the definition to deal with the structural rules, and taking more care when formulas are discharged. We keep track not only of the proof constructed, but also which premises in the proof correspond to what formulas in the antecedent multiset.

» If \( \delta \) is an identity sequent \( p \vdash p \), then \( nd(\delta) \) is the proof with the sole assumption \( p \). This is a proof from \( p \) to \( p \). The assumption \( p \) matches the antecedent \( p \) in the sequent.

» If \( \delta \) is a derivation

\[
\begin{array}{c}
\vdash \delta' \\
X, A, \Delta \vdash B \\
\hline
X, A \vdash B
\end{array}
\]

then we already have a proof \( nd(\delta) \) of \( B \) and we know which premises of this proof match formulas in \( X \) and which match each instance of \( A \) in the sequent \( X, A, \Delta \vdash B \). The proof for \( X, A \vdash B \) is exactly the same, but now the premises matching either of the repeated \( A \)s now match the single displayed \( A \), and premises matching formulas in \( X \) still do so.

» If \( \delta \) is a derivation

\[
\begin{array}{c}
\vdash \delta' \\
X \vdash B \\
\hline
X, A \vdash B
\end{array}
\]

then we already have a proof \( nd(\delta) \) of \( B \) and we know which premises of this proof match formulas in \( X \) and which match each instance of \( A \) in the sequent \( X, A, \Delta \vdash B \). The proof for \( X, A \vdash B \) is exactly the same, where no premises match the introduced \( A \), and premises matching formulas in \( X \) still do so.

» If \( \delta \) is a derivation

\[
\begin{array}{c}
\vdash \delta' \\
X, A \vdash B \\
\hline
X \vdash A \rightarrow B
\end{array}
\]

then we already have the proof \( nd(\delta') \) from \( X, A \) to \( B \), and we know which premises of this proof match the formula \( A \) in the antecedent. The proof \( nd(\delta) \), from \( X \) to \( A \rightarrow B \) is the following:

\[
\begin{array}{c}
X, [A]^{(i)} \\
\vdash \text{nd}(\delta') \\
B \\
\hline
A \rightarrow B
\end{array}
\]
where we discharge all and only those premises matching the distinguished \( A \) in \( X, A \). The remaining premises match the same formulas in \( X \) they did before.

» If \( \delta \) is a derivation

\[
\begin{array}{c}
\vdash \delta_1 \\
\vdash \delta_2 \\
X \vdash A, B, Y \vdash R
\end{array}
\]
\[
\frac{A \rightarrow B, X, Y \vdash R}{A \rightarrow B, X, Y \vdash R} \rightarrow L
\]

then we already have the proofs \( nd(\delta_1) \) from \( X \) to \( A \) and \( nd(\delta_2) \) from \( B, Y \) to \( R \). The proof \( nd(\delta) \), from \( A \rightarrow B, X, Y \) to \( R \) is the following:

\[
\begin{array}{c}
\vdash \delta_1 \\
X \\
A \rightarrow B \\
A \\
B \\
\vdash \delta_2 \\
Y \\
R
\end{array}
\]

Y

However, the proof of \( B \) from \( A \rightarrow B \) and \( nd(\delta_1) \) is inserted at all of the premises \( B \) matching the formula \( B \) displayed in the antecedent of the sequent \( B, Y \vdash R \). The introduced premise \( A \rightarrow B \) matches the sole \( A \rightarrow B \) displayed in the sequent \( A \rightarrow B, X, Y \vdash R \). The premises in each copy of \( X \) match their mates in the displayed \( X \) in the sequent according to the matching of \( nd(\delta_1) \), and the premises appearing in \( Y \) match their mates in \( Y \) according to the matching of \( nd(\delta_2) \).

» If \( \delta \) is a derivation

\[
\begin{array}{c}
\vdash \delta_3 \\
\vdash \delta_4 \\
X \vdash C, Y \vdash R
\end{array}
\]
\[
\frac{X, Y \vdash R}{X, Y \vdash R} \text{Cut}
\]

then we already have the proofs \( nd(\delta_3) \) from \( X \) to \( C \) and \( nd(\delta_4) \) from \( C, Y \) to \( R \). The proof \( nd(\delta) \), from \( X, Y \) to \( R \) is the following:

\[
\begin{array}{c}
\vdash \delta_3 \\
X \\
\vdash \delta_4 \\
C \\
\vdash \delta_4 \\
Y \\
\vdash \delta_4 \\
R
\end{array}
\]

where in the same way, the proof \( nd(\delta_3) \) of \( C \) is plugged in at all of the points at which a premise \( C \) matches the \( C \) in \( C, Y \vdash R \). The matches among premises and the antecedent in \( X, Y \vdash R \) are defined in the obvious way.

As before, we can see, by inspecting the way that the recipe \( \delta \) is used in \( nd \), if \( \delta \) \( \text{Cut-free} \), then the proof \( nd(\delta) \) is normal.

The construction of \( \text{sq}^b \) and \( \text{sq}^p \) is actually less difficult than the definition of \( nd \). For in this case, it does not matter of finding how to
interpret the structural rules in the natural deduction proof—where structural rules are only used when it comes to discharging premises—it is a matter instead of inserting structural rules where needed. And here, we know exactly where the structural rules are needed: they are needed exactly when premises are discharged. Here is the definition of $sq^b$.

**BOTTOM-UP CONSTRUCTION OF DERIVATIONS:** If $\pi$ is a proof from $X$ to $A$ then $sq^b(\pi)$ is a derivation of the sequent $X \vdash A$, defined as follows:

- If $\pi$ is an assumption $A$, then $sq^b(\pi)$ is the derivation $Id_A$.
- If $\pi$ is a proof from $X$ to $A \rightarrow B$, composed from a proof $\pi'$ from $X'$ to $B$ by $\rightarrow I$, in which some number of premises $A$ are discharged, then take the derivation $sq^b(\pi')$ of the sequent $X' \vdash B$, then if it was no instances of $A$ discharged, then $X' = X$ and we weaken in one instance of $A$ to discharge it, as follows:

$$
\begin{array}{c}
\vdash sq^b(\pi') \\
X \vdash B \\
\hline
X, A \vdash B \\
\hline
\bar{X} \vdash A \rightarrow B
\end{array}
$$

If, on the other hand, more than one instance of $A$ is discharged, then we use a number of contraction steps to reduce them to one instance to discharge:

$$
\begin{array}{c}
\vdash \vdash \\
X, A, \ldots, A \vdash B \\
\hline
X, A \vdash B \\
\hline
\bar{X} \vdash A \rightarrow B
\end{array}
$$

Finally, if it was one and only one instance of $A$ to discharge, we proceed as before.

- The step for $\rightarrow E$ is unchanged.

The definition for $sq^p$ takes exactly the same form. Lemma 4.7 concerning the structure of normal proofs holds for normal proofs with vacuous or duplicate discharge, so we may peel off major premises in $\rightarrow E$ steps in normal proofs as before.

It follows, as before, that sequent derivations with $Cut$ can be transformed into derivations without it. Simply take the same route as before. For any derivation $\delta$ of $X \vdash A$, $sq^p(norm(nd(\delta)))$ is a $Cut$-free derivation of $X' \vdash A$ for some close variant $X'$ of the multiset of premises. Adding vacuous premises with weakening steps or contracting multiple instances of premises with contractions, will suffice to finish off the derivation to lead to $X \vdash A$.

So, $Cut$ adds nothing to the stock of derivable sequents. In fact, we may eliminate $Cut$ directly, just as before. The structural rules
do not interfere with the elimination of Cut, though contraction does make the elimination of Cut more difficult. The first thing to note is that every formula occurring in a structural rule is passive. We may commute cuts above structural rules. In the case of the weakening rule, the weakened-in formula appears only in the endsequent. If the Cut is made on the weakened-in formula, it disappears, and is replaced by further instances of weakening, like this:

\[
\begin{array}{c}
\vdash \delta_1 \\
X \vdash A \\
\vdash \delta_2 \\
Y, A \vdash B \\
\hline
X, Y \vdash B
\end{array}
\]

In this case, the effect of the Cut step is achieved without any Cuts at all. The new derivation is clearly simpler, in that the derivation \(\vdash \delta_1\) is rendered unnecessary, and the number of cuts decreases. If the Cut formula is not the weakened in formula, then the Cut permutes trivially with the weakening step:

\[
\begin{array}{c}
\vdash \delta_1 \\
X \vdash A \\
\vdash \delta_2 \\
Y, A \vdash B \\
\hline
X, Y \vdash B
\end{array}
\]

In the case contraction formula matters are not so simple. If the contracted formula is the Cut formula, it occurs once in the endsequent but twice in the topsequent. This means that if this formula is the Cut formula, when the Cut is pushed upwards it duplicates.

\[
\begin{array}{c}
\vdash \delta_1 \\
X \vdash A \\
\vdash \delta_2 \\
Y, A, A \vdash B \\
\hline
X, Y \vdash B
\end{array}
\]

In this case, the new proof is not less complex than the old one. The depth of the second Cut in the new proof \((2|\delta_1| + |\delta_2| + 1)\) is greater than in the old one \((|\delta_1| + |\delta_2|)\). The old proof of Cut elimination no longer works in the presence of contraction. There are a number of options one might take here. Gentzen’s own approach is to prove the elimination of multiple applications of cut.

\[
\begin{array}{c}
\vdash \delta_1 \\
X \vdash A \\
\vdash \delta_2 \\
Y, A, A \vdash B \\
\hline
X, X, Y \vdash B
\end{array}
\]

PROOFS & DERIVATIONS: TREES • CHAPTER 4
Another option is to eliminate contraction as one of the rules of the system (retaining its effect by rewriting the connective rules) [25]. In our treatment of the elimination of Cut we will not take either of these approaches. We will be more subtle in the formulation of the inductive argument, following a proof due to Haskell Curry [15, page 250], in which we trace the occurrences of the Cut formula in the derivation back to the points (if any) at which the formulas become active. We show how Cuts at these points—on simpler formulas—suffice for the derivation of the endsequent without further Cuts. Here is an example fragment of a derivation. The darkly shaded formula occurrences are those active formulas from which the instances of the Cut formula $C \rightarrow D$ arise:

The Cut on $C \rightarrow D$ in the old derivation may be reconstructed in terms of Cuts on the simpler formulas $C$ and $D$, deferring the structural rules after the Cuts.

We won’t go through the detail of the Cut elimination argument here: suffice to say that this technique for dealing with the interaction of contraction and Cut works, and is the only subtlety in the argument. The technique of Theorem 4.12 works, with this emendation. Instead of working through the detail here, we will defer its presentation for Section 4.5 for when we have more connectives and rules to deal with.

### 4.3 CONJUNCTION AND DISJUNCTION

Let’s add conjunction and disjunction to our vocabulary. We have a number of options for the rules. One straightforward option would be to use natural deduction, and use the traditional rules. For example, the
rules for conjunction in Gentzen’s natural deduction are

\[
\begin{align*}
\frac{A \quad B}{A \land B} & \quad ^{\text{\&I}} & \frac{A \land B}{A} & \quad ^{\text{\&E}} & \frac{A \land B}{B} & \quad ^{\text{\&E}}
\end{align*}
\]

Notice that the structural rule of weakening is implicit in these rules:

\[
\frac{A \quad B}{A \land B} \quad ^{\text{\&I}}
\]

\[
\frac{A \land B}{A} \quad ^{\text{\&E}}
\]

So, if we wish to add conjunction to a logic in which we don’t have weakening, we must modify the rules. If we view these rules as sequents, it is easy to see what has happened:

\[
\begin{array}{c}
X \vdash A \\
Y \vdash B \\
X, Y \vdash A \land B \\
\hline
X, A \land B \vdash R
\end{array}
\]

The effect of weakening is then found using a Cut.

\[
\begin{array}{c}
A \vdash A \\
B \vdash B \\
A, B \vdash A \land B \\
\hline
A \land B \vdash A
\end{array}
\]

The \&R? rule combines two contexts (X and Y) whereas the \&L? does not combine two contexts—it merely infers from A \land B to A within the one context. The context ‘combination’ structure (the ‘comma’ in the sequents, or the structure of premises in a proof) is modelled using conjunction using the \&R? rule but the structure is ignored by the \&L? rule. It turns out that there are two kinds of conjunction (and disjunction).

The rules in Figure 4.3 are said to be additive. These rules do not exploit premise combination in the definition of the connectives. (The “X,” in the conjunction left and disjunction left rules is merely a passive ‘bystander’ indicating that the rules for conjunction may apply regardless of the context.) These rules define conjunction and disjunction, regardless of the presence or absence of structural rules.

\[
\begin{array}{c}
X, A \vdash R \\
\hline
X, A \land B \vdash R
\end{array}
\]

\[
\begin{array}{c}
X, A \vdash R \\
\hline
X, B \land A \vdash R
\end{array}
\]

\[
\begin{array}{c}
X \vdash A \\
X \vdash B \\
\hline
X \vdash A \land B
\end{array}
\]

\[
\begin{array}{c}
X, A \vdash R \\
X, B \vdash R \\
\hline
X, A \lor B \vdash R
\end{array}
\]

\[
\begin{array}{c}
X \vdash A \\
\hline
X \vdash A \lor B
\end{array}
\]

\[
\begin{array}{c}
X \vdash A \\
\hline
X \vdash B \lor A
\end{array}
\]

Figure 4.3: ADDITIVE CONJUNCTION AND DISJUNCTION RULES

These rules are the generalisation of the lattice rules for conjunction seen in the previous section. Every sequent derivation in the old system
is a proof here, in which there is only one formula in the antecedent multiset. We may prove many new things, given the interaction of implication and the lattice connectives:

\[
\begin{align*}
  p ⊨ p & \quad q ⊨ q \\
  p \rightarrow q, p ⊨ q & \rightarrow^L \\
  p \rightarrow q_1, p ⊨ q & \rightarrow^L \\
  (p \rightarrow q) \land (p \rightarrow r), p ⊨ q & \land^L \\
  (p \rightarrow q) \land (p \rightarrow r), p ⊨ r & \land^L \\
  (p \rightarrow q) \land (p \rightarrow r), p ⊨ q \land r & \land^R \\
  (p \rightarrow q) \land (p \rightarrow r), p ⊨ (q \land r) & \rightarrow^R
\end{align*}
\]

Just as with the sequents with pairs of formulas, we cannot derive the sequent \( p \land (q \lor r) \vdash (p \land q) \lor (p \land r) \)—at least, we cannot without any structural rules. It is easy to see that there is no Cut-free derivation of the sequent. There is no Cut-free derivation using only sequents with single formulas in the antecedent (we saw this in the previous section) and a Cut-free derivation of a sequent in \( \land \) and \( \lor \) containing no commas, will itself contain no commas (the additive conjunction and disjunction rules do not introduce commas into a derivation of a conclusion if the conclusion does not already contain them). So, there is no Cut-free derivation of distribution. As we will see later, this means that there is no derivation at all.

But the situation changes in the presence of the structural rules. (See Figure 4.4.) This sequent is derivable with the use of both weakening and contraction, but not without them.

\[
\begin{align*}
  A & ⊨ A \\
  B & ⊨ B & K \\
  A, B & ⊨ A & K \\
  A, B & ⊨ B & K \\
  A, B & ⊨ A \land B & \land^R \\
  A, B & ⊨ (A \land B) \lor (A \land C) & \lor^R_1 \\
  A, B & ⊨ (A \land B) \lor (A \land C) & \lor^R_2 \\
  A, C & ⊨ A \land C & \land^R \\
  A, C & ⊨ A \land C & \land^R \\
  A, C & ⊨ A \land C & \land^R \\
  A, C & ⊨ A \land C & \land^R \\
  A, C & ⊨ A \land C & \land^R \\
  A, A \land (B \lor C), A \land (B \lor C) & ⊨ (A \land B) \lor (A \land C) & \land^L_2 \\
  A \land (B \lor C), A \land (B \lor C) & ⊨ (A \land B) \lor (A \land C) & \land^L_1 \\
  A \land (B \lor C) & ⊨ (A \land B) \lor (A \land C) & W
\end{align*}
\]

Figure 4.4: DISTRIBUTION OF \( \land \) OVER \( \lor \), USING \( K \) AND \( W \).

Without using weakening, there is no way to derive \( A, B ⊨ A \land B \) using the additive rules for conjunction. If we think of the conjunction of \( A \) and \( B \) as the thing derivable from both \( A \) and \( B \), then this seems to define a different connective.

The fact that without weakening we cannot derive \( A, B ⊨ A \land B \) motivates a different pair of conjunction rules. After all, \( A, B ⊨ C \) means we can derive \( C \) from \( A \) and \( B \) together. These are the so-called

§4.3 • CONJUNCTION AND DISJUNCTION
**Multiplicative rules** for conjunction. We use a different symbol (the tensor: \( \otimes \)) for conjunction defined with the multiplicative rules, because in certain proof-theoretic contexts (in the absence of either contraction or weakening), they differ.

\[
X, A, B \vdash C \\
\vdash A \quad \vdash B \\
X, A \otimes B \vdash C \\
\vdash X, Y \vdash A \otimes B
\]

**Figure 4.5: Multiplicative Conjunction Rules.**

The multiplicative conjunction rules clearly make the conjunction connective \( \otimes \) correspond tightly with the comma of premise combination. \( A \otimes B \) is a way of saying in a single formula what \( A, B \) together do as premises. So, for example, \( (A \rightarrow B) \otimes A \) is a single formula from which we can derive \( B \):

\[
\begin{align*}
A \vdash A \\
B \vdash B \\
\vdash A \rightarrow B, A \vdash B \\
\vdash (A \rightarrow B) \otimes A \vdash B
\end{align*}
\]

As another example of the difference between \( \land \) and \( \otimes \), you can derive the distribution of \( \otimes \) over \( \lor \) (see Figure 4.6) in the absence of any structural rules. Notice that the derivation is much simpler than the case for additive conjunction \( \land \).

\[
\begin{align*}
A \vdash A \\
B \vdash B \\
A, B \vdash A \otimes B \\
\vdash A \land C \vdash A \otimes C \\
A, B \vdash (A \otimes B) \lor (A \otimes C) \\
A \land (B \lor C) \vdash (A \otimes B) \lor (A \otimes C)
\end{align*}
\]

**Figure 4.6: Distribution of \( \otimes \) over \( \lor \).**

What can we say about the connection between \( \land \) and \( \otimes \), in the presence of structural rules? Consider the following two derivations:

\[
\begin{align*}
A \vdash A \\
B \vdash B \\
A, B \vdash A \land B \\
\vdash A \land B, A \vdash A \otimes B \\
\vdash A \land B \vdash A \otimes B
\end{align*}
\]

The first shows that in the presence of weakening, a multiplicative conjunction \( \otimes \) entails the additive conjunction \( \land \). The second shows
that in the presence of contraction, the converse holds. So, if we have both weakening and contraction, additive and multiplicative conjunction are equivalent in the sense that whenever one is used as a premise, the other could have done instead, and whenever one features as a conclusion, the other could have done just as well.

Does this mean that \(\&\) and \(\land\) are completely equivalent in meaning? It depends on what we need for equivalence. Though \(A \land B\) and \(A \otimes B\) are interderivable, it does not follow that they can be derived in the same way. \(A \land B \vdash A \land B\) has \(Id_{A \land B}\) as a derivation. That’s not a derivation of \(A \otimes B \vdash A \land B\). What does this mean for the role of proof in a theory of meaning? It’s not our place to look at that in this chapter, but merely to raise it for later in the discussion.

### 4.4 | NEGATION

You can get some of the features of negation by defining it in terms of conditionals. If we pick a particular atomic proposition (call it \(f\) for the moment) then \(A \rightarrow f\) behaves somewhat like the negation of \(A\). For example, we can derive \(A \vdash (A \rightarrow f) \rightarrow f\), \((A \lor B) \rightarrow f \vdash (A \rightarrow f) \land (B \rightarrow f)\), and vice versa, \((A \rightarrow f) \land (B \rightarrow f) \vdash (A \lor B) \rightarrow f\). Here is one example:

\[
\begin{align*}
A \vdash A & \quad f \vdash f \\
A \rightarrow f, A \vdash f & \rightarrow^L \\
B \vdash B & \quad f \vdash f \\
B \rightarrow f, B \vdash f & \rightarrow^L \\
(A \rightarrow f) \land (B \rightarrow f), A \vdash f & \land^L_1 \\
(A \rightarrow f) \land (B \rightarrow f), B \vdash f & \land^L_2 \\
(A \rightarrow f) \land (B \rightarrow f), A \lor B \vdash f & \lor \\
(A \rightarrow f) \land (B \rightarrow f) \rightarrow f & \rightarrow^R
\end{align*}
\]

Notice that no special properties of \(f\), and no structural rules are required for this derivation to work. The proposition \(f\) is completely arbitrary. Now look at the rules for implication in this special case of implying \(f\):

\[
\begin{align*}
X \vdash A & \quad f, Y \vdash R \\
A \rightarrow f, X, Y \vdash R & \rightarrow^L \\
X, A \vdash f & \rightarrow^L \\
X \vdash A \rightarrow f & \rightarrow^R
\end{align*}
\]

If we want to do this without appealing to the proposition \(f\), we could consider what happens if \(f\) goes away. Write \(A \rightarrow f\) as \(\neg A\), and consider first \(\rightarrow R\). If we erase \(f\), we get

\[
\begin{align*}
X, A & \vdash \\
X \vdash \neg A & \rightarrow^R
\end{align*}
\]

Now the topsequent has an empty right-hand side. What might this mean? One possible interpretation is that \(X, A \vdash \) is a refutation of \(A\) in the context of \(X\). This could be similar to a proof from \(X\) and \(A\) to a contradiction, except that we have no particular contradiction in mind. To derive \(X, A \vdash \) is to refute \(A\) (if we are prepared to keep \(X\)). In other

\[\text{Something to think about: why have I not introduced } \otimes, \text{ a multiplicative disjunction as a mate for } \lor? \text{ There is such a thing, but it is not expressible using the sequents we have seen so far. Before peeking ahead to the next chapter, ask yourself this: why not?}\]
words, from $X$ we can derive $\neg A$, the negation of $A$. If we keep with this line of investigation, consider the rule $\to L$. First, notice the right premise sequent $f, Y \vdash R$. A special case of this is $f \vdash$, and we can take this sequent as a given: if a refutation of a statement is a deduction from it to $f$, then $f$ is self-refuting. So, if we take $Y$ and $R$ to be empty, $A \to f$ to be $\neg A$ and $f \vdash$ to be given, then this is what is left of $\to L$.

$$
\begin{align*}
X & \vdash A \\
\frac{}{X, \neg A \vdash} & \neg L
\end{align*}
$$

If we can deduce $A$ from $X$, then $\neg A$ is refuted (given $X$). This seems an eminently reasonable thing to mean by ‘not’. And, these are Gentzen’s rules for negation. With them, we can prove many of the usual properties of negation, even in the absence of other structural rules. The proof of distribution of negation over conjunction (one of the de Morgan laws) simplifies in the following way:

$$
\begin{align*}
A & \vdash A \\
\frac{}{\neg A, A \vdash} & \neg L \\
\frac{}{\neg A \land \neg B, A \vdash} & \land L_1 \\
\frac{}{\neg A \land \neg B, A \vdash} & \land L_2 \\
\frac{}{\neg A \land \neg B, A \lor B \vdash} & \lor L \\
\frac{}{\neg A \land \neg B \vdash (A \lor B)} & \neg R
\end{align*}
$$

We may show that $A$ entails its double negation $\neg \neg A$

$$
\begin{align*}
A & \vdash A \\
\frac{}{A, \neg A \vdash} & \neg R
\end{align*}
$$

but we cannot prove the converse. There is no proof of $p$ from $\neg \neg p$. Similarly, there is no derivation of $\neg (p \land q) \vdash \neg p \lor \neg q$, using all of the structural rules considered so far. What of the other property of negation, that contradictions imply everything? We can get this far:

$$
\begin{align*}
A & \vdash A \\
\frac{}{A, \neg A \vdash} & \neg L \\
\frac{}{A \otimes \neg A \vdash} & \otimes L
\end{align*}
$$

(using contraction, we can derive $A \land \neg A \vdash$ too) but we must stop there in the absence of more rules. To get from here to $A \otimes \neg A \vdash B$, we must somehow add $B$ into the conclusion. But the $B$ is not there! How can we do this? We can come close by adding $B$ to the left by means of a weakening move:

$$
\begin{align*}
A & \vdash A \\
\frac{}{A, \neg A \vdash} & \neg L \\
\frac{}{A, \neg A, B \vdash} & \neg K \\
\frac{}{A, \neg A \vdash \neg B} & \neg R
\end{align*}
$$
This shows us that a contradiction entails any negation. But to show that a contradiction entails anything we need a little more. We can do this by means of a structural rule operating on the right-hand side of a sequent. Now that we have sequents with empty right-hand sides, we may perhaps add things in that position, just as we can add things on the left by means of a weakening on the right. The rule of right weakening is just what is required to derive \( A, \neg A \vdash B \).

\[
\frac{X \vdash} {X \vdash B} \text{KR}
\]

The result is a sequent system for intuitionistic logic. Intuitionistic logic arises out of the program of intuitionism in mathematics due to L. E. J. Brouwer [13, 47]. The entire family of rules is listed in Figure 4.7.

The sequents take the form \( X \vdash R \) where \( X \) is a multiset of formulas and \( R \) is either a single formula or is empty. We take a derivation of \( X \vdash A \) to record a proof of \( X \) from \( A \). Furthermore, a derivation of \( X \vdash \) records a refutation of \( X \). The system of intuitionistic logic is a stable, natural and useful account of logical consequence [17, 40, 73].

We have not presented the entire system of natural deduction in which these proofs may be found — yet.

---

**Identity and Cut**

- \( p \vdash p \text{ Id} \)
- \( X \vdash C \quad C, X' \vdash R \quad \text{Cut} \)

**Structural Rules**

- \( X, A, A \vdash R \quad \text{W} \)
- \( A \vdash R \quad \text{KL} \)
- \( X \vdash \text{KR} \)

**Conditional Rules**

- \( X \vdash A \quad B, X' \vdash R \quad \rightarrow L \)
- \( A \rightarrow B, X, X' \vdash R \quad X, A \vdash B \quad \rightarrow R \)

**Negation Rules**

- \( X \vdash A \quad \neg L \)
- \( X, A \vdash \neg R \)
- \( X, \neg A \vdash \)
- \( X \vdash \neg A \)

**Additive Conjunction Rules**

- \( X, A \vdash R \quad \text{\&}_1 \)
- \( X, A \land B \vdash R \quad \text{\&}_2 \)
- \( X, B \land A \vdash R \quad X \vdash A \quad X \vdash B \quad \text{\&}_R \)

**Additive Disjunction Rules**

- \( X, A \vdash R \quad X, B \vdash R \quad \lor \quad X \vdash A \quad X \vdash B \quad \lor \text{R}_1 \)
- \( X, A \lor B \vdash R \quad \lor \text{R}_2 \)

**Multiplicative Conjunction Rules**

- \( X, A, B \vdash R \quad \otimes \text{L} \)
- \( X, A \otimes B \vdash R \quad \otimes R \)
- \( X, Y \vdash A \otimes B \)

---

Figure 4.7: sequent rules for intuitionistic propositional logic

These rules do not just characterise intuitionistic logic: if we leave out some of the structural rules, we get weaker logical systems, such as relevant, linear, and affine logics. If we leave out some of the connective rules, we have fragments of these logics.

To do: a nice table of different systems should go here.
A case could be made for the claim that intuitionistic logic is the strongest and most natural logic you can motivate using inference rules on sequents of the form \( X \vdash R \). It is possible to go further and to add rules to ensure that the connectives behave as one would expect given the rules of classical logic: we can add the rule of double negation elimination

\[
\frac{X \vdash \neg
\neg A}{X \vdash A} \quad \text{DNE}
\]

which strengthens the system far enough to be able to derive all classical tautologies and to derive all classically valid sequents. However, the results are not particularly attractive on proof-theoretical considerations. For example, the rule DNE does not satisfy the subformula property: the concluding sequent \( X \vdash A \) is derived by way of the premise sequent involving negation, even when negation does not feature in \( X \) or in \( A \). This feature of the rule is exploited in the derivation of the sequent \( \vdash ((p \to q) \to p) \to p \), of Peirce’s Law, which is classically derivable but not derivable intuitionistically. This derivation uses negation liberally, despite the fact that the concluding sequent is negation-free.

\[
\frac{p \vdash p}{p, (p \to q) \vdash p \quad \text{KL}}
\]

\[
\frac{p \vdash ((p \to q) \to p) \to p}{p \vdash ((p \to q) \to p) \quad \text{\( \to \)R}}
\]

\[
\frac{\neg(((p \to q) \to p) \to p), p \vdash q}{\neg(((p \to q) \to p) \to p) \vdash q \quad \text{\( \to \)R}}
\]

\[
\frac{\neg(((p \to q) \to p) \to p) \vdash p \to q}{\neg(((p \to q) \to p) \vdash p \quad \text{\( \to \)L}}
\]

\[
\frac{\neg(((p \to q) \to p) \vdash p \to q), (p \to q) \to p \vdash p}{\neg(((p \to q) \to p) \to p) \vdash (p \to q) \to p \quad \text{\( \to \)R}}
\]

\[
\frac{\neg(((p \to q) \to p) \vdash p \to q), \neg(((p \to q) \to p) \to p) \vdash p}{\neg(((p \to q) \to p) \vdash p \quad \text{\( \to \)L}}
\]

\[
\frac{\neg(((p \to q) \to p) \vdash p \to q)}{\neg(((p \to q) \to p) \vdash p \quad \text{\( \neg \)R}}
\]

\[
\frac{\vdash \neg(((p \to q) \to p) \to p)}{\vdash ((p \to q) \to p) \to p \quad \text{DNE}}
\]

It seems clear that this is not a particularly simple proof of Peirce’s law. It violates the subformula property, by way of the detour through negation. Looking at the structure of the proof, it seems clear that the contraction step (marked \( \text{\( W \)R} \)) is crucial. We needed to duplicate the conditional for Peirce’s law so that the inference of \( \to \text{\( L \)R} \) would work. Using \( \to \text{\( L \)R} \) on an unduplicated Peirce conditional does not result in a derivable sequent. The options for deriving \( (p \to q) \to p \vdash p \) are grim:

\[
\frac{\vdash p \to q \quad p \vdash p}{(p \to q) \to p \vdash p \quad \text{\( \to \)L}}
\]

No such proof will work. We must go through negation to derive the sequent, unless we can find a way to mimic the behaviour of the \( \text{\( W \)R} \) step without passing the formulas over to the left side of the turnstile, using negation.
4.5 | CUT ELIMINATION

In this section we will go over the Cut elimination proof for the entire derivation system, with all of the rules in Figure 4.7 (and in subsystems, in which structural rules, or entire connectives are missing). However, now we have seen the Cut elimination argument twice already, and instead of going through it again with more connectives and more detail, the time has come to state some general principles underlying the argument, and then summarise how any sequent calculus satisfying these conditions will allow Cuts to be eliminated in the way we have seen. The general strategy the same as in Theorems 3.9 and 4.12: we take a derivation involving a Cut, above which there are no others.

\[ \vdots \delta_1 \quad \vdots \delta_2 \]
\[ X \vdash A \quad Y, A \vdash B \]
\[ \frac{}{X, Y \vdash B} \text{Cut} \]

We look at the Cut formula (in this example, A): if it is active in the concluding inferences of both \( \delta_1 \) and \( \delta_2 \) we trade the Cut in for a Cut on a smaller formula, or if one of \( \delta_1 \) or \( \delta_2 \) was an axiom, it disappears. If the Cut formula is passive in one or other of those concluding inferences, we push the Cut upwards. The added complexity for the argument in our new setting is twofold. First, there are many more rules to consider. To deal with this new complexity, we will abstract away from the particulars of each rule, to isolate a number of general properties required to make the argument work, and we will show that any sequent system satisfying these properties allows for the elimination of Cuts. Second, the presence of structural rules like \( W \) and \( K \) makes the argument for lifting Cuts upward more difficult, as we have already seen on page 93.

So, let us start with the conditions. The easiest are those where the Cut formula is active. We start with identity.

Definition 4.14 [The cut/axiom condition] A sequent system satisfies the Cut/axiom condition if and only if any instance of Cut in which one premise is an axiom can be replaced by a derivation not involving a Cut.

Our sequent system satisfies the Cut/axiom condition. Any Cut on an axiomatic sequent is a cut on an identity, and is one of these two cases:

\[ \vdots \delta_2 \]
\[ A \vdash A \]
\[ Y, A \vdash B \]
\[ \frac{}{Y, A \vdash B} \text{Cut} \]
\[ \vdots \delta_1 \]
\[ X \vdash A \]
\[ A \vdash A \]
\[ \frac{}{X \vdash A} \text{Cut} \]

and in either case, the Cut is redundant. Cuts on identities do not take us anywhere but back from where we came, so we can avoid the Cut by not taking the trip.

Now consider the case where the Cut formula \( A \) is introduced as a complex formula on both sides. Here are the conditions on active formulas.

I learned this technique from Nuel Belnap’s wonderful “Display Logic” [7], who adapted the argument from the presentation in Haskell Curry’s Foundations of Mathematical Logic [15].
Definition 4.15 [Single Active Formula] A formula is active in a rule if it is not passive. A system satisfies the single active formula condition if and only if each inference has only one active formula below the line.

For our system this can be checked by examining the rules one by one. This condition is important, as we want the Cut rule, to be operating on the single active formula, if the cut formula is active. If two formulas could be active at one time, then dealing with the cut on one could interfere with the processing of the other.

The crucial condition is this. Cuts on matching formulas may be traded in for Cuts on subformulas.

Definition 4.16 [Eliminability of Matching Active Formulas] A system satisfies the eliminability of matching active formulas condition if and only if instance of Cut in which the Cut formula is active in both inferences immediately before the Cut may be traded in for a Cut (or cuts) on subformulas of the Cut formula.

We have seen examples already. Here is another example, for multiplicative conjunction. Suppose our Cut step looks like this:

\[
\begin{array}{c}
\vdash \delta_1 & \vdash \delta_2 & \vdash \delta_3 \\
X \vdash A & Y \vdash B & Z, A, B \vdash C \\
\hline
X, Y \vdash A \otimes B & Z, A \otimes B \vdash C \\
\hline
X, Y, Z \vdash C
\end{array}
\]

The Cut on \(A \otimes B\) could be traded in for two Cuts, on \(A\) and on \(B\).

\[
\begin{array}{c}
\vdash \delta_1 & \vdash \delta_3 \\
X \vdash A & Z, A, B \vdash C \\
\hline
Y \vdash B & X, Z, B \vdash C \\
\hline
X, Y, Z \vdash C
\end{array}
\]

The other rules can be checked in the same way.

Next we have conditions on passive formulas in rules. Recall, passive formulas in an inference falling under a rule is every formula except for the major formulas in a connective rule (the formula with the connective introduced below the line and its ancestor formulas above the line), and the Cut formulas in a Cut step. Every other formula is a parameter. Parameters may appear both above and below the line. A passive class is a collection of instances of a formula in a proof. Two formulas are a part of the same passive class if they are represented by the same letter in a presentation of the rule (the instances of \(A\) in an inference of contraction, for example) or if they occur in the same place in a structure.

Definition 4.17 [Regularity Condition] A sequent system satisfies the regularity condition if whenever a Cut formula is passive in an inference immediately before the Cut step, then Cut may be permuted above that inference.
For example the segment

\[
\frac{Y, A \vdash B}{X \vdash A} \quad \frac{Y, A \vdash B \lor C}{Y, X \vdash B \lor C} \quad \lor R
\]

\[
\frac{Y, A \vdash B \lor C}{Y, X \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor R
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]

\[
\frac{X, Y \vdash B}{X, Y \vdash B} \quad \lor L
\]

\[
\frac{X, Y \vdash B \lor C}{X, Y \vdash B \lor C} \quad \lor L
\]
• cut/axiom
• single active formula
• eliminability of matching active formulas
• regularity
• position-alikeness of passive formulas
• non-proliferation of passive formulas

and in which the subformula property is well-founded, then if a sequent is derivable using Cut it is also derivable without it.

Proof: We perform an induction on the complexity of the Cut formula \( A \). The hypothesis is:

Cut on \( A \): If the premises of a Cut-rule in which an \( A \) is the Cut-formula are derivable, so is the conclusion.

We wish to show that if cut on \( A' \) holds for every subformula \( A' \) of \( A \), then cut on \( A \) holds too. (The induction works because being a subformula is well-founded). So, suppose we have derivations \( \delta \) and \( \delta' \) of \( X \vdash A \) and \( Y, A \vdash B \) respectively. If the Cut-formula \( A \) indicated in the concluding inferences of \( \delta \) and \( \delta' \) is active, then we may apply the eliminability of matching active formula condition and our induction hypothesis (or the cut/axiom condition, if the derivation started here) to eliminate the Cut.

If, on the other hand, \( A \) is passive in either \( \delta \) or \( \delta' \), we proceed as follows: Without loss of generality, suppose \( A \) is parametric in \( \delta \). Consider the class \( A \) of occurrences of \( A \) in \( \delta \) found by tracing up the derivation and selecting each passive instance of \( A \) congruent with the \( A \) in the conclusion of \( \delta \). By regularity and non-proliferation we commute the cut on \( A \) (with the other premise \( Y, A \vdash B \) past each inference in which an instance in \( A \) features. The result is a derivation in which there may be many more Cuts, but for each Cut on \( A \) introduced in the derivation, there are no passive instances of \( A \) in consequent position (by position alikeness). For each copy of \( \delta' \) introduced, we may form the set \( A' \) of instances of \( A \) congruent with the \( A \) in antecedent position in the cut inference. We commute the cut with each inference crossing the set \( A' \) to construct a derivation in which the cut on \( A \) occurs only on active instances of \( A \), and this case has already been covered.

4.6 | EXERCISES

BASIC EXERCISES

Q1 Find derivations of the following formulas and use as few structural rules as you can get away with.

1: \( p \otimes q \rightarrow p \)
2: \( (p \rightarrow q) \otimes (q \rightarrow r) \rightarrow (p \rightarrow r) \)
3: \( p \rightarrow (q \rightarrow p \otimes q) \)
4 : \( p \rightarrow (q \rightarrow p \land q) \)
5 : \( ((p \rightarrow q) \land p) \rightarrow q \)
6 : \( \neg(p \otimes \neg p) \)
7 : \( \neg(p \land \neg p) \)
8 : \( (p \rightarrow q) \rightarrow ((p \land r) \rightarrow (q \land r)) \)
9 : \( (p \rightarrow q) \rightarrow ((p \lor r) \rightarrow (q \lor r)) \)
10 : \( p \rightarrow \neg \neg p \)
11 : \( \neg \neg \neg p \rightarrow \neg p \)
12 : \( \neg(p \lor q) \rightarrow (\neg p \land \neg q) \)
13 : \( (\neg p \land \neg q) \rightarrow (p \lor q) \)
14 : \( (\neg p \lor \neg q) \rightarrow (p \land \neg q) \)
15 : \( (p \land (q \rightarrow r)) \rightarrow (q \rightarrow (p \land r)) \)
16 : \( (\neg p \lor \neg q) \rightarrow (p \rightarrow q) \)

Q2 Construct a derivation for \( p \rightarrow q, q \rightarrow r \vdash p \rightarrow r \), and a derivation for \( p \rightarrow r \vdash \neg r \rightarrow \neg p \). Join the two derivations with a Cut, and then eliminate Cut from that derivation.

Q3 Define the Double Negation Translation \( d(A) \) of formula \( A \) as follows:

\[
\begin{align*}
d(p) & = \neg\neg p \\
d(\neg A) & = \neg d(A) \\
d(A \land B) & = d(A) \land d(B) \\
d(A \lor B) & = \neg(\neg d(A) \land \neg d(B)) \\
d(A \rightarrow B) & = d(A) \rightarrow d(B)
\end{align*}
\]

What formulas are \( d((p \rightarrow q) \lor (q \rightarrow p)) \) and \( d(\neg\neg p \rightarrow p) \)? Show that these formulas have intuitionistic proofs by giving a sequent derivation for each.

INTERMEDIATE EXERCISES

Q4 Show that whenever the argument from \( X \) to \( A \) is valid in classical two-valued logic, then there is an intuitionistic derivation of \( d(X) \vdash d(A) \). (The multiset \( d(X) \) is the multiset of formulas \( d(A) \) for each \( A \) in \( X \).)

ADVANCED EXERCISES

Q5 Consider what sort of rules make sense in a sequent system with sequents of the form \( A \vdash X \), where \( A \) is a formula and \( X \) a multiset. What connectives make sense?
PROOFS & DERIVATIONS: CIRCUITS

We have seen how proofs from some collection of premises (X) to a conclusion (A) may be represented as a tree of statements, and they may be reasoned about using sequents (X ⊢ A) connecting an antecedent multiset with a consequent formula. When we came to intuitionist logic, the story expanded just a little bit: we allowed for refutations: sequents without a consequent formula (X ⊬ ).

In this chapter, our topic is a more radical change in the structure of proofs and of derivations. Instead of allowing for one or zero formulas in consequent position, we will see what happens when we treat consequents of sequents in exactly the same way as antecedents. This was another of Gentzen’s great ideas in proof theory, and it had radical consequences.

5.1 | SEQUENTS FOR CLASSICAL LOGIC

One of Gentzen’s great insights in the sequent calculus was that we could get the full power of classical logic by way of a small but profound change to the structure of sequents: We allow for more than one formula on both sides of the turnstile. For intuitionistic logic we already allow a single formula or none. We now allow for more. The rules are trivial modifications of the standard intuitionistic rule, except for this one change. The rules are listed in Figure 5.1. As usual, we use ‘p’ for atomic formulas, ‘A’, ‘B’, ‘C’ for arbitrary formulas, and ‘X’, ‘Y’, etc., for multisets (possibly empty) of formulas. There are no changes with the rules, other than natural ways for allowing extra formulas in consequent position. So, for example, the Cut rule is modified to allow extra formulas (in Y and X′) in the consequent to pass through from premises to conclusion in just the same way as they pass from premises to conclusion in the antecedent (in X and X′).

Recall from the last chapter the derivation of Peirce’s Law

\[ \vdash ((p \to q) \to p) \to p \]

using the sequent calculus for intuitionist logic, extended with the rule for double negation elimination. We can derive \( \neg \neg ((p \to q) \to p) \to p \), and then, derive Peirce’s Law by eliminating the double negation. This derivation allowed us to duplicate \( \neg ((p \to q) \to p) \to p \) twice in antecedent position, using W. In the expanded sequent calculus get the same effect without the dodge of using a negation. Instead of having to swing the formula onto the left (using negation) to duplicate it in a contraction step, we may keep it on the right of the turnstile to perform the duplication, since now there is room to do this. The negation laws
are eliminated, the $W$ step on the left changes into a $W$ step on the right, but the other rules are unchanged. Here is the new, simpler, derivation.

\[
\begin{align*}
p \vdash p & \quad KL \\
p, (p \rightarrow q) \rightarrow p \vdash p & \quad KR \\
p \vdash p, ((p \rightarrow q) \rightarrow p) \rightarrow p & \quad \rightarrow R \\
p \vdash p \rightarrow q, ((p \rightarrow q) \rightarrow p) \rightarrow p & \quad \rightarrow L \\
p \vdash ((p \rightarrow q) \rightarrow p) \rightarrow p & \quad \rightarrow R
\end{align*}
\]

So, we can do more with our sequent system with multiset/multiset sequents than we could with multiset/formula sequents. Let’s get a sense of exactly how much more we can do. The system of rules is elegant and completely symmetric between left and right. Derivations have mirror images in every respect. Here are two double negation sequents:

\[
\begin{align*}
\text{Structural Rules} & \\
\text{Conditional Rules} & \\
\text{Negation Rules} & \\
\text{Conjunction Rules} & \\
\text{Disjunction Rules} & \\
\text{Figure 5.1: classical propositional logic sequent rules (system I)}
\end{align*}
\]

You might think that this derivation is, in some sense, what we were ‘trying’ to do in the other derivation. But there, we had to be sneaky with negation to do what we wished. Instead of keeping the extra $(p \rightarrow q) \rightarrow p$ in the consequent of the sequent as we can here, we had to stash it (under a negation) in the antecedent.

Well, in almost every respect. The mirror image of $\land$ is $\lor$. Negation is its own mirror image. What about the conditional? $Q$: What is the mirror image of the conditional? $A$: It has none here, but we could define it: the difference connective $B \rightarrow A$, which stands to $B \land \neg A$ as $A \rightarrow B$ stands to $\neg A \lor B$.
Here are two derivations of de Morgan laws. Again, they are completely left–right symmetric pairs (with conjunction and disjunction exchanged, but negation kept fixed).

\[
\begin{align*}
A & \vdash A & \quad B & \vdash B \\
\neg A, A & \vdash \quad \neg B, B & \vdash \quad \neg L \\
\neg A \land \neg B, A & \vdash \quad \neg B \land \neg A, B & \vdash \quad \land L_1 \\
\neg A \lor \neg B, A & \vdash \quad \neg B \lor \neg A, B & \vdash \quad \lor L_2 \\
\neg A \land \neg B, A \lor B & \vdash \quad \neg (A \lor B) & \quad \neg R \\
\neg A \lor \neg B, A \land B & \vdash \quad \neg (A \land B) & \quad \land R
\end{align*}
\]

The sequent rules for classical logic share the ‘true–false’ duality implicit in the truth-table account of classical validity. But this leads on to an important question. Intuitionistic sequents, of the form \(X \vdash A\), record a proof from \(X\) to \(A\). What do classical sequents mean? Do they mean anything at all about proofs? A sequent of the form \(A, B \vdash C, D\) does not tell us that \(C\) and \(D\) \emph{both} follow from \(A\) and \(B\). (Then it could be replaced by the two sequents \(A, B \vdash C\) and \(A, B \vdash D\).) No, the sequent \(A, B \vdash C, D\) may be valid even when \(A, B \vdash \neg C\) and \(A, B \vdash \neg D\) are not valid. The combination of the conclusions is \emph{disjunctive} and not \emph{conjunction} when read ‘positively’. We can think of a sequent \(X \vdash Y\) as proclaiming that if \emph{each} member of \(X\) is true then \emph{some} member of \(Y\) is true. Or to put it ‘negatively’, it tells us that it would be \emph{inconsistent} to assert each member of \(X\) and to deny each member of \(Y\).

This leaves open the important question: is there any notion of \emph{proof} appropriate for structures like these, in which premises and conclusions are collected in exactly the same way? Whatever is suitable will differ from the tree-structured proofs we have already seen. But before we take a look at what might count as a \emph{proof} behind Gentzen’s sequents for classical logic, let’s verify that the sequent calculus we have seen indeed delivers us classical two-valued propositional logic. Along the way, we will get an insight into the duality behind classical sequents, and as a bonus, we will get a very different proof showing us that any sequent derivable with \emph{Cut} is also derivable without it.

5.2 | TRUTH TABLES AND CUT

We start with some familiar definitions:

**Definition 5.1 [Truth-Table Evaluations]** A function \(v\) from formulas to the set \{TRUE, FALSE\} of truth values is said to be a \emph{truth-table evaluation} if it respects the usual truth-table laws:

- \(v(\neg A) = \text{TRUE} \quad \text{iff} \quad v(A) = \text{FALSE}\)
- \(v(A \land B) = \text{TRUE} \quad \text{iff} \quad v(A) = \text{TRUE} \text{ and } v(B) = \text{TRUE}\)
- \(v(A \lor B) = \text{TRUE} \quad \text{iff} \quad v(A) = \text{TRUE} \text{ or } v(B) = \text{TRUE}\)
- \(v(A \rightarrow B) = \text{TRUE} \quad \text{iff} \quad v(A) = \text{FALSE} \text{ or } v(B) = \text{TRUE}\)

You can think of an evaluation as a row of a truth-table, if that helps.
In other words, a sequent \( X \vdash Y \) is truth-table valid if and only if there is no way to make each element of \( X \) true while making each element of \( Y \) false. Or, if you like, if we make each member of \( X \) true, we must also make some member of \( Y \) true. Or, to keep the story balanced, if we make each member of \( Y \) false, we make some member of \( X \) false too. Now, we will show two facts. Firstly, that if the sequent \( X \vdash Y \) is derivable (with or without Cut) then it is truth-table valid. Second, we will show that if the sequent \( X \vdash Y \) is not derivable (again, with or without Cut), then it is not truth-table valid. Can you see why this proves that anything derivable with Cut may be derived without it? Let’s prove these two results, and then apply them after that.

**Theorem 5.3 [Truth-table soundness of sequent rules]** If \( X \vdash Y \) is derivable (using Cut if you wish), then it is truth-table valid.

**Proof:** Axiom sequents (identities) are clearly truth-table valid, since any valuation assigning p true assigns p true. Consider now the rules: we must show that for any instance of any rule, if the premises of that rule are truth-table valid, then so is the conclusion. We will consider two examples, and leave the rest as an exercise. Consider the rule \( \rightarrow L \):

\[
\frac{X \vdash Y, A \quad B, X' \vdash Y'}{X, X', A \rightarrow B \vdash Y, Y'}_{\rightarrow L}
\]

Suppose that \( X \vdash Y, A \) and \( B, X' \vdash Y' \) are both truth-table valid, and that we have an evaluation \( v \) for which each member of \( X, X' \), and \( A \rightarrow B \) is true. We wish to show that some member of \( Y, Y' \) is true according to \( v \). Now, since \( A \rightarrow B \) is true according to \( v \), it follows that either \( A \) is false or \( B \) is true (again, according to \( v \)). If \( A \) is false, then by the truth-table validity of \( X \vdash Y, A \), it follows that one member (at least) of \( Y \) is true according to \( v \), since all of \( X \) is true. On the other hand, if \( B \) is true, then the truth-table validity of \( B, X' \vdash Y' \) tells us that one member (at least) of \( Y' \) is true. In either case, at least one member of \( Y, Y' \) is true according to \( v \), as we desired.

That was a connective rule. Now for a structural rule. Consider the rule \( KR \):

\[
\frac{X \vdash Y}{X \vdash A, Y}^{\text{KR}}
\]

Suppose that \( X \vdash Y \) is truth-table valid. Is \( X \vdash A, Y \)? Suppose we have an evaluation making each element of \( X \) true. By the truth-table validity of \( X \vdash Y \), some member of \( Y \) is true according to \( v \). It follows that some member of \( Y, A \) is true according to \( v \) too. The other rules are no more difficult to verify, so (after you verify the rest of the rules to your own satisfaction) you may declare this theorem proved.
So, derivations will never supply us with a sequent that is not truth-table valid. If we take truth-tables as our measure of correctness then the sequent system is sound. It never steps out of the bounds of truth-table correctness. This raises the opposite question: Does it provide us a derivation for all of the truth-table valid sequents? That is the next result.

**Theorem 5.4 [Truth-Table Completeness for Sequents]** If \( X \vdash Y \) is truth-table valid, then it is derivable. In fact, it has a derivation that does not use Cut.

To prove this, we will demonstrate the converse: that if a sequent \( X \vdash Y \) has no Cut-free derivation, then it is not truth-table valid—that is, there is some evaluation \( v \) assigning \textsc{true} to each member of \( X \), and \textsc{false} to each member of \( Y \). To do this, we will take a detour through another collection of sequent rules for classical logic, displayed in Figure 5.2.

**Figure 5.2:** Alternative sequent rules for classical logic (system 2)

<table>
<thead>
<tr>
<th>Identity and Cut</th>
<th>( X, A \vdash A, Y \ (\text{id}) )</th>
<th>( X \vdash Y, C )</th>
<th>( C, X \vdash Y )</th>
<th>( X \vdash Y ) (Cut)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conditional Rules</td>
<td>( X \vdash Y, A ) ( B, X \vdash Y )</td>
<td>( A \rightarrow B, X \vdash Y ) (( \rightarrow L ))</td>
<td>( X, A \vdash B, Y )</td>
<td>( X \vdash A \rightarrow B, Y ) (( \rightarrow R ))</td>
</tr>
<tr>
<td>Negation Rules</td>
<td>( X \vdash A, Y )</td>
<td>( X, \neg A \vdash Y ) (( \neg L ))</td>
<td>( X, A \vdash Y )</td>
<td>( X \vdash \neg A, Y ) (( \neg R ))</td>
</tr>
<tr>
<td>Conjunction Rules</td>
<td>( X, A, B \vdash Y )</td>
<td>( X, A \land B \vdash Y ) (( \land L ))</td>
<td>( X \vdash A, Y ) ( X \vdash B, Y )</td>
<td>( X \vdash A \land B, Y ) (( \land R ))</td>
</tr>
<tr>
<td>Disjunction Rules</td>
<td>( X, A \vdash Y ) ( X, B \vdash Y )</td>
<td>( X, A \lor B \vdash Y ) (( \lor L ))</td>
<td>( X \vdash A, B, Y )</td>
<td>( X \vdash A \lor B, Y ) (( \lor R ))</td>
</tr>
<tr>
<td>Structural Rules</td>
<td>( X, A, A \vdash Y )</td>
<td>( X, A \vdash A, Y ) (( WL ))</td>
<td>( X, A \vdash A, Y )</td>
<td>( X \vdash A ) (( WR ))</td>
</tr>
</tbody>
</table>

This is not to say that we should take truth-tables as our standard of correctness, of course. It’s just that this is the way terminology has come to be used.

**Lemma 5.5 [Equivalence Between Sequent Systems]** There is a derivation \( \delta \) for \( X \vdash Y \) in the sequent system in Figure 5.1 (System 1) if and only if there is a derivation \( \delta' \) for that sequent in the system in Figure 5.2 (System 2). In fact, \( \delta \) and \( \delta' \) differ only in the structural rules applied.

**Proof:** This is a matter of showing how rules in System 1 can be encoded by those in System 2, and vice versa. The major difference
between the two systems is that System 2 incorporates the weakening rule into the *Identity* sequent, and the other rules keep the passive formulas (those in X and Y) fixed from premise to conclusion. To transform a derivation δ from System 1 into a derivation from System 2, we first add instances of weakening to ensure that in each connective rule and (and *Cut*) that the passive formulas agree on each side. We make transformations like this:

\[
\frac{X \vdash Y, A \quad B, X' \vdash Y'}{X, X', A \rightarrow B \vdash Y, Y'} \Rightarrow \frac{X \vdash Y, A \quad B, X' \vdash Y'}{X, X', A \vdash B \vdash Y, Y', Y'} \Rightarrow X, X', A \vdash B \vdash Y, Y' \]

The premises and conclusions of this step in the derivation is exact as before, but now, the \(\Rightarrow\) step used is applied to sequents with the same passive formulas in each case. This works for every rule, since in every rule, we are free to inflate the collection of passive formulas at will. In this case, we could now replace the \(\Rightarrow\) step from System 1 with a \(\Rightarrow\) step from System 2, and do away with the contractions below the \(\Rightarrow\) step, which would not be needed, but before we do that, we need to deal with instances of weakening in the derivation. For those, we shift all instances of weakening to the top of the derivation, by commuting the instances with each rule, as follows:

\[
\frac{X \vdash Y, A \quad B, X' \vdash Y'}{X, X', A \rightarrow B \vdash Y, Y'} \Rightarrow \frac{X \vdash Y, A \quad B, X' \vdash Y'}{X, C \vdash Y, A \quad B, X', C \vdash Y'} \Rightarrow X, X', A \vdash B \vdash C \vdash Y, Y' \]

Again, since all rules allow the passive formulas to be expanded at will, this the result is still a derivation. Note, too, that since all rules have a single sequent at the bottom, any formula we add to one premise of the rule we add to the others, so rules have matched collections of side formulas if they were matched before the instances of weakening passed through.

So, the result is a derivation in which the premises of every connective rule have matching collections of passive formulas (and a series of contractions below them to reduce the size of those collections back to what they were if the rules have two premises), and the weakenings are now immediately under the axioms.

Now we move to System 2. We incorporate the instances of weakening into the *Identity* sequents, and eliminate the instances of contraction immediately below each connective rule, like this:

\[
\frac{X \vdash Y, A \quad B \vdash Y}{X, X, A \rightarrow B \vdash Y, Y} \Rightarrow \frac{X \vdash Y, A \quad B \vdash Y}{X, A \vdash B \vdash Y} \Rightarrow X, A \rightarrow B \vdash Y \]

The result is a derivation in which the connective rules are unchanged, but the structural rules have shifted.
To transform derivations from System 2 to System 1, we replace appeals to \( \text{Identity sequent } X, A \vdash A, Y \) to appeals to \( \text{Id}(A) \) combined with weakening to introduce \( X \) and \( Y \). The rules with single premises are unchanged from System 2 to System 1, and the rules with two premises are replaced by the corresponding rules from System 1 with enough instances of contraction, in this way:

\[
\begin{align*}
X \vdash Y, A & \quad B, X \vdash Y \\
\frac{}{A \rightarrow B, X \vdash Y} \quad \rightarrow_{L(2)}
\end{align*}
\]

becomes

\[
\begin{align*}
X \vdash Y, A & \quad B, X \vdash Y \\
\frac{A \rightarrow B, X, X \vdash Y, Y}{} \quad \rightarrow_{L(1)}
\end{align*}
\]

The result is a derivation in System 1, in which the connective rules are unmoved, but the structural rules have shifted.

If you think of structural rules as bookkeeping, we have kept the ‘essence’ of the derivation fixed from one system to the other but varied the bureaucracy. Regardless of the significance of this lemma for what it is for one derivation to be ‘essentially the same’ as another, we are now in a position to prove Theorem 5.4.

**Proof**: We prove the contrapositive: that if the sequent \( X \vdash Y \) has no Cut-free derivation, then it is not truth-table valid. To do this, we appeal to our Lemma, to show that if we have a derivation (without Cut) using the System 2 then we have a derivation (also without Cut) in our original system, System 1.

System 2 has some very interesting properties. Suppose we have a sequent \( X \vdash Y \), that has no Cut-free derivation in this system. Then we may reason in the following way:

**[Atomic Formulas]** Suppose \( X \vdash Y \) contains no complex formulas, and only atoms. Since it is un derivable, it is not an instance of the new \( \text{Id} \) rule. That is, it contains no formula common to both \( X \) and \( Y \). Therefore, counterexample evaluation \( v \): simply take each member of \( X \) to be TRUE and \( Y \) to be FALSE.

That deals with sequents of atoms. We now proceed by induction, with the hypothesis for a sequent \( X \vdash Y \) being that if it has no Cut-free derivation, it is truth-table invalid. We will show that if the hypothesis holds for simpler sequents than \( X \vdash Y \) then it holds for \( X \vdash Y \) too. What is a simpler sequent than \( X \vdash Y \)? Let’s say that the complexity of a sequent is the number of connectives (\( \land, \lor, \rightarrow, \neg \)) occurring in that sequent. So, we have shown that the hypothesis holds for sequents of complexity zero. Now we must deal with sequents of greater complexity: that is, those containing formulas with connectives. We will consider each negation and the conditional.

**[Negation]** Suppose that the sequent contains a negation formula \( \neg A \). If this formula occurs on the left, the sequent has the form \( X, \neg A \vdash Y \). It follows that if this is undervisible in our sequent system, then so is \( X \vdash A, Y \). If this were derivable, then we could...
derive our target sequent by \( \neg L \). But look! This is a simpler sequent. So we may appeal to the induction hypothesis to give us a counterexample evaluation \( v \), making each member of \( X \) **true** and \( A \) **false** and each member of \( Y \) **false**. Now since this evaluation makes \( A \) **false**, then it makes \( \neg A \) **true**. So, it is a counterexample for \( X, \neg A \vdash Y \) too.

If the negation formula occurs on the right instead, then the sequent has the form \( X \vdash \neg A, Y \). It follows that \( X, A \vdash Y \) is underivable (for otherwise, we could derive our sequent by \( \neg R \)). This is a simpler sequent, so it has a counterexample \( v \), making each member of \( X \), and \( A \) **true** and \( Y \) **false**. This is also a counterexample to \( X, \neg A \vdash Y \), since it makes \( \neg A \) **false**.

[**Conditional**] Suppose that our sequent contains a **conditional** formula \( A \rightarrow B \). If it occurs on the left, the sequent has the form \( A \rightarrow B, X \vdash Y \). If it is not derivable then, using the rules of System 2, we may conclude that either \( X \vdash Y, A \) is underivable, or \( B, X \vdash Y \) is underivable. (If they were both derivable, then we could use \( \rightarrow L \) to derive our target sequent \( A \rightarrow B, X \vdash Y \).) Both of these sequents are simpler than our original sequent, so we may apply the induction hypothesis. If \( X \vdash Y, A \) is underivable, we have an evaluation \( v \) making each member of \( X \) **true** and each member of \( Y \) **false**, together with \( A \) **false**. But look! This makes \( A \rightarrow B \) **false**, so \( v \) is a counterexample for \( A \rightarrow B \). Similarly, if \( B, X \vdash Y \) is underivable, we have a counterexample \( v \), making each member of \( X \) **true** and each member of \( Y \) **false**, together with making \( B \) **true**. So we are in luck in this case too! The evaluation \( v \) makes \( A \rightarrow B \) true, so it is a counterexample to our target sequent \( A \rightarrow B, X \vdash Y \). This sequent is truth-table invalid.

Suppose, on the other hand, that an implication formula is on the right hand side of the sequent. If \( X \vdash A \rightarrow B, Y \) is not derivable, then neither is \( X, A \vdash B, Y \), a simpler sequent. The induction hypothesis applies, and we have an evaluation \( v \) making the formulas in \( X \) **true**, the formulas in \( Y \) **false**, and \( A \) **true** and \( Y \) **false**. So, it makes \( A \rightarrow B \) **false**, and our evaluation is a counterexample to our target sequent \( X \vdash A \rightarrow B \). This sequent is truth-table invalid.

[**Conjunction and disjunction**] The cases for conjunction and disjunction are left as exercises. They pose no more complications than the cases we have seen.

So, the sequent rules, read **backwards** from bottom-to-top, can be understood as giving instructions for making a counterexample to a sequent. In the case of sequent rules with more than one premise, these instructions provide **alternatives** which can both be explored. If a sequent is underivable, these instructions may be followed to the end, and we finish with a counterexample to the sequent. If following the instructions does not meet with success, this means that all searches
have terminated with derivable sequents. So we may play this attempt backwards, and we have a derivation of the sequent. Our Theorem has a useful byproduct.

**Corollary 5.6 [Simple Admissibility of Cut]** If \( X \vdash Y \) has a derivation with Cut (in either System 1 or in System 2) it has a derivation without it (in either System 1 or in System 2).

**Proof:** If there is no Cut-free derivation of \( X \vdash Y \) (in either System 1 or in System 2), then by Theorem 5.4 there is an evaluation according to which each element of \( X \) is **true** and each element of \( Y \) is **false**. By Theorem 5.3 this means that there is no derivation of \( X \vdash Y \) in System 1, and by Lemma 5.5 there is no derivation in System 2 either.

This does not give us any way to eliminate Cut from a derivation by direct transformation. It just tells us that if we had a derivation of a sequent, using Cut then a search for a derivation without Cut would lead us to one, since there is a Cut-free derivation to be found. For this reason, a result like this is most often called an admissibility theorem for Cut, since adding Cut to the system does not enlarge the stock of derivable sequents: these sequents are closed under the Cut rule already, so it is admissible. The theorem does not tell us how we can eliminate the appeal to Cut in a derivation, so it is not a Cut-elimination theorem. For the elimination theorem (which will work along exactly the same lines we have seen already), we must wait for a few sections. Before that, we must attend to proofs for involving more than one conclusion.

### 5.3 Derivations Describing Circuits

In this section we will look at the kinds of proofs motivated by the two-sided classical sequent calculus. Our aim is to complete this square:

- Derivations of \( X \vdash A \) are to tree proofs from \( X \) to \( A \), as
- Derivations of \( X \vdash Y \) are to ???

Just what goes in that corner? If the parallel is to work, the structure is not a straightforward tree with premises at the top and conclusion at the bottom, as we have in proofs for a single conclusion \( A \). What other structure could it be?

We will look first, not at the case of classical logic. The structural rules of weakening and contraction complicate the picture. As with our first look at tree proofs, we will start with a logic without these structural rules—Jean-Yves Girard’s linear logic [32].

Our target sequent system does without the structural rules of contraction or weakening. However, sequents have multisets on the left and on the right. In this section we will work with the connectives \( \otimes \) and \( \oplus \) (multiplicative conjunction and disjunction respectively) and
The sequent rules are as follows: First, negation flips conclusion to premise, and vice versa.

\[
\frac{X \vdash A, Y}{X, \neg A \vdash Y} \quad \frac{X, A \vdash Y}{X \vdash \neg A, Y}
\]

Multiplicative conjunction mirrors the behaviour of premise combination. We may trade in the two premises \(A, B\) for the single premise \(A \otimes B\). On the other hand, if we have a derivation of \(A\) (from \(X\), and with \(Y\) as alternate conclusions) and a derivation of \(B\) (from \(X'\) and with \(Y'\) as alternate conclusions) then we may combine these derivations to form a derivation of \(A \otimes B\) from both collections of premises, and with both collections of alternative conclusions.

\[
\frac{X, A, B \vdash Y}{X, A \otimes B \vdash Y} \quad \frac{X, A \vdash Y}{X \vdash A, Y}
\]

The case for multiplicative disjunction is dual to the case for conjunction. We swap premise and conclusion, and replace \(\otimes\) with \(\oplus\).

\[
\frac{X, A \vdash Y \quad X', B \vdash Y'}{X, X', A \oplus B \vdash Y, Y'} \quad \frac{X \vdash A, B, Y}{X \vdash A \oplus B, Y}
\]

The \textit{Cut} rule is simple:

\[
X \vdash A, Y \quad X', A \vdash Y' \quad \text{Cut}
\]

\[
\frac{X \vdash A, Y \quad X', A \vdash Y'}{X, X' \vdash Y, Y'}
\]

The \textit{Cut} formula (here it is \(A\)) is left out, and all of the other material remains behind. Any use of the \textit{Cut} rule is eliminable, in the usual manner. Notice that this proof system has no conditional connective. Its loss is no great thing, as we could define \(A \rightarrow B\) to be \(\neg(A \otimes \neg B)\), or equivalently, as \(\neg A \oplus B\). So that is our sequent system for the moment.

Let's try to find a notion of \textit{proof} appropriate for the derivations in this sequent system. It is clear that the traditional many-premise single-conclusion structure does not fit neatly. The \textit{Cut} free derivation of \(\neg\neg A \vdash A\) is no simpler and no more complex than the \textit{Cut} free derivation of \(A \vdash \neg\neg A\).

\[
\frac{A \vdash \neg A}{\neg \neg A \vdash A} \quad \frac{\neg \neg A \vdash A}{A \vdash \neg \neg A}
\]

The natural deduction proof from \(A\) to \(\neg\neg A\) goes through a stage where we have two premises \(A\) and \(\neg A\) and has no active conclusion (or equivalently, it has the conclusion \(\bot\)).

\[
A \quad \neg \neg A \quad \frac{[\neg A]^{(1)}}{\neg \neg A} \quad \frac{\ast}{\neg \neg A} \quad \frac{\neg \neg A}{\neg \neg A}
\]
In this proof, the premise ¬A is then discharged or somehow otherwise converted to the conclusion ¬¬A. The usual natural deduction proofs from ¬¬A to A are either simpler (we have a primitive inference from ¬A to A) or more complicated. A proof that stands to the derivation of ¬¬A ⊢ A would require a stage at which there is no premise but two conclusions. We can get a hint of the desired “proof” by turning the proof for double negation introduction on its head:

\[
\begin{array}{c}
\neg \vdash \neg
\\
\vdash * \\
(1) \neg[neg] \vdash
\end{array}
\]

Let’s make it easier to read by turning the formulas and labels the right way around, and swap I labels with E labels:

\[
\begin{array}{c}
\neg E \vdash \neg A \\
\neg I \vdash * \\
(1) \neg[neg] \vdash A
\end{array}
\]

We are after a proof of double negation elimination at least as simple as this. However, constructing this will require hard work. Notice that not only does a proof have a different structure to the natural deduction proofs we have seen—there is downward branching, not upward—there is also the kind of “reverse discharge” at the bottom of the tree which seems difficult to interpret. Can we make out a story like this? Can we define proofs appropriate to linear logic?

To see what is involved in answering this question in the affirmative, we will think more broadly to see what might be appropriate in designing our proof system. Our starting point is the behaviour of each rule in the sequent system. Think of a derivation ending in X ⊢ Y as having constructed a proof π with the formulas in X as premises or inputs and the formulas in Y as conclusions, or outputs. We could think of a proof as having a shape somewhat reminiscent of the traditional proofs from many premises to a single conclusion:

\[
A_1 \quad A_2 \quad \cdots \quad A_n \\
B_1 \quad B_2 \quad \cdots \quad B_m
\]

However, chaining proofs together like this is notationally very difficult to depict. Consider the way in which the sequent rule Cut corresponds to the composition of proofs. In the single-formula-right sequent system, a Cut step like this:

\[
X \vdash C \quad A, C, B \vdash D
\]

\[
A, X, B \vdash D \quad \text{Cut}
\]
corresponds to the composition of the proofs

\[
\begin{array}{ccc}
X & A & C & B \\
\pi_1 & \pi_2 & \\
C & D & \\
\end{array}
\quad \text{to form} \quad
\begin{array}{ccc}
X & A & C & B \\
\pi_1 & \pi_2 & \\
D & \\
\end{array}
\]

In the case of proofs with multiple premises and multiple conclusions, this notation becomes difficult if not impossible. The \textit{Cut} rule has an instance like this:

\[
X \vdash D, C, E \quad A, C, B \vdash Y \\
\text{Cut} \quad A, X, B \vdash D, Y, E
\]

This should correspond to the composition of the proofs

\[
\begin{array}{ccc}
X & A & C & B \\
\pi_1 & \pi_2 & \\
D & C & E & Y
\end{array}
\]

If we are free to rearrange the order of the conclusions and premises, we could manage to represent the \textit{Cut}:

\[
\begin{array}{ccc}
X & A & C & B \\
\pi_1 & \pi_2 & \\
D & E & C & A & B
\end{array}
\]

But we cannot always rearrange the \textit{Cut} formula to be on the left of one proof and the right of the other. Say we want to \textit{Cut} with the conclusion \(E\) in the next step, and then \(D\) after that? What could we do?

It turns out that it is much more flexible to change our notation completely. Instead of representing proofs as consisting of characters on a page, ordered in a tree diagrams, think of proofs as taking inputs and outputs, where we represent the inputs and outputs as \textit{wires}. Wires can be rearranged willy-nilly—we are all familiar with the tangle of cables behind the stereo or under the computer desk—so we can exploit this to represent \textit{Cut} straightforwardly. In our pictures, then, formulas will \textit{label} wires. This change of representation will afford another insight: instead of thinking of the rules as labelling transitions between formulas in a proof, we will think of inference steps (instances of our rules) as \textit{nodes} with wires coming in and wires going out. Proofs are then \textit{circuits} composed of wirings of nodes. Figure 5.3 should give you the idea.
A proof $\pi$ for the sequent $X \vdash Y$ has premise or *input* wires for each formula in $X$, and conclusion or *output* wires for each formula in $Y$. Now think of the contribution of each rule to the development of inferences. The *Cut* rule is the simplest. Given two proofs, $\pi_1$ from $X$ to $A$, $Y$, and $\pi_2$ from $X'$, $A$ to $Y'$, we get a new proof by chaining them together. You can depict this by “plugging in” the $A$ output of $\pi_1$ into the $A$ input of $\pi_2$. The remaining material stays fixed. In fact, this picture still makes sense if the *Cut* wire $A$ occurs in the middle of the output wires of $\pi_1$ and in the middle of the input wires of $\pi_2$.

![Diagram of a circuit and chains of two circuits](image)

**Figure 5.3: A circuit, and chaining together two circuits**

So, we are free to tangle up our wires as much as we like. It is clear from this picture that the conclusion wire $A$ from the proof $\pi_1$ is used as a premise in the proof $\pi_2$. It is just as clear that *any* output wire in one proof may be used as an input wire in another proof, and we can always represent this fact diagrammatically. The situation is much improved compared with upward-and-downward branching *tree* notation.

Now consider the behaviour of the connective rules. For negation, the behaviour is simple. An application of a negation rule turns an output $A$ into an input $\neg A$ (this is $\neg L$), or an input $A$ into an output $\neg A$ (this is $\neg R$). So, we can think of these steps as plugging in new *nodes* in the circuit. A $\neg E$ node takes an input $A$ and input $\neg A$ (and has no outputs), while a $\neg I$ node has an output $A$ and and output $\neg A$ (and has no inputs).

In other words, these nodes may be represented in the following way:

![Negation nodes](image)

Regardless, I will try to make proofs as tangle free as possible, for ease of reading.

Draw for yourself the result of making two cuts, one after another, inferring from the sequents $X_1 \vdash A, Y_1$ and $X_2, A \vdash B, Y_2$ and $X_3, B \vdash Y_3$ to the sequent $X_1, X_2, X_3 \vdash Y_1, Y_2, Y_3$. You get two different derivations depending on whether you *Cut* on $A$ first or on $B$ first. Does the order of the cuts matter when these different derivations are represented as circuits?

Ignore, for the moment, the little dots. These features have a significance, to be revealed in good time.
and they can be added to existing proofs to provide the behaviour of the sequent rules $\neg L$ and $\neg R$.

$\frac{X \vdash A, Y}{X, \neg A \vdash Y} \quad \text{becomes} \quad \pi$

Here, a circuit for $X \vdash A, Y$ becomes, with the addition of a $[\neg E]$ node, a circuit for $X, \neg A \vdash Y$. Similarly,

$\frac{X, A \vdash Y}{X \vdash \neg A, Y} \quad \text{becomes} \quad \pi$

a circuit for the sequent $X, A \vdash Y$ becomes, with the addition of a $[\neg I]$ node, a circuit for $X \vdash \neg A, Y$. Notice how these rules (or nodes) are quite simple and local. They do not involve the discharge of assumptions (unlike the natural deduction rule $\neg I$ we have already seen). Instead, these rules look like straightforward transcriptions of the law of non-contradiction ($A$ and $\neg A$ form a dead-end—don’t assert both) and the law of the excluded middle (either $A$ or $\neg A$ is acceptable—don’t deny both).

For conjunction, the right rule indicates that if we have a proof $\pi$ with $A$ as one conclusion, and a proof $\pi'$ with $B$ as another conclusion, we can construct a proof by plugging in the $A$ and the $B$ conclusion wires into a new node with a single conclusion wire $A \otimes B$. This motivates a node $\otimes I$ with two inputs $A$ and $B$ an a single output $A \otimes B$. So, the node looks like this:

$\otimes I$

and it can be used to combine circuits in the manner of the $\otimes R$ sequent rule:
This is no different from the natural deduction rule

\[
\begin{array}{c}
A & B \\
\hline
A \otimes B
\end{array}
\]

except for the notational variance and the possibility that it might be employed in a context in which there are conclusions alongside \(A \land B\). The rule \(\otimes E\), on the other hand, is novel. This rule takes a single proof \(\pi\) with the two premises \(A\) and \(B\) and modifies it by wiring together the inputs \(A\) and \(B\) into a node which has a single input \(A \otimes B\). It follows that we have a node \(\otimes E\) with a single input \(A \otimes B\) and two outputs \(A\) and \(B\).

In this case the relevant node has one input and two outputs:

\[
\begin{array}{c}
A \otimes B \\
\hline
A & B
\end{array}
\]

This is not a mere variant of the rules \(\land E\) in traditional natural deduction. It is novel. It corresponds to another kind of natural deduction rule, in which premises are discharged, and the new premise \(A \otimes B\) is used in its place.

\[
\begin{array}{c}
[A, B] \\
\vdots \\
A \otimes B & C \\
\hline
C
\end{array}
\]

In the presence of the traditional structural rules, these two conjunction-elimination rules are equivalent, but without these rules, there is no way to get from one rule to the other.

The extent of the novelty of this rule becomes apparent when you see that the circuit for \(\otimes E\) also has one input and two outputs, and the two outputs are \(A\) and \(B\), if the input is \(A \otimes B\). The step for \(\oplus L\) takes two proofs: \(\pi_1\) with a premise \(A\) and \(\pi_2\) with a premise \(B\), and combines them into a proof with the single premise \(A \otimes B\). So the node for \(\otimes E\) looks identical. It has a single input wire (in this case, \(A \otimes B\)), and two
output wires, A and B

The same happens with the rule to introduce a disjunction. The sequent step $\oplus R$ converts the two conclusions $A$, $B$ into the one conclusion $A \oplus B$. So, if we have a proof $\pi$ with two conclusion wires $A$ and $B$, we can plug these into a $\oplus I$ node, which has two input wires $A$ and $B$ and a single output wire $A \oplus B$.

Notice that this looks just like the node for $\otimes I$. Yet $\otimes$ and $\oplus$ are very different connectives. The difference between the two nodes is not their shape (two inputs, one output), but due to the different ways that they are added to a circuit.

Let’s start with one example circuit, to show how it can be built up step-by-step. We start with an $\otimes I$ node

which gives us $p, q \vdash p \otimes q$, and then add an $\oplus E$

to give us \( p, q \oplus r \vdash p \otimes q, r \). Then we can add the nodes \( \otimes E \) and \( \oplus I \) to the front and back of the proof, to get

\[
\begin{array}{c}
p \\
p \otimes (q \oplus r) \\
\otimes E \\
q \oplus r \\
\oplus E \\
p \oplus q \\
(p \otimes q) \oplus r
\end{array}
\]

which is a circuit for \( p \otimes (q \oplus r) \vdash (p \otimes q) \oplus r \).

## 5.4 | CIRCUITS FROM DERIVATIONS

**Definition 5.7 [Inductively generated circuit]** A derivation of \( X \vdash Y \) constructs a circuit with input wires labelled with the formulas in \( X \) and output wires labelled with the formulas in \( Y \) in the manner we have seen in the previous section. We will call these circuits inductively generated.

Here is an example. This derivation:

\[
\begin{array}{c}
A \vdash A \\
B \vdash B \\
A, \neg A \vdash B, \neg B \vdash \\
A, B, \neg A \oplus \neg B \vdash \\
A, B \vdash \neg (\neg A \oplus \neg B) \\
A \otimes B \vdash \neg (\neg A \oplus \neg B)
\end{array}
\]

can be seen as constructing the following circuit:

\[
\begin{array}{c}
p \otimes q \\
\otimes E \\
q \\
\neg E \\
p \oplus q \\
\oplus E \\
\neg q \\
\neg E \\
\neg (\neg p \oplus \neg q)
\end{array}
\]

Just as with other natural deduction systems, this is representation of derivations is efficient, in that different derivations can represent the one and the same circuit. This derivation

\[
\begin{array}{c}
A \vdash A \\
B \vdash B \\
A, \neg A \vdash B, \neg B \vdash \\
A, B, \neg A \oplus \neg B \vdash \\
A \otimes B, \neg A \oplus \neg B \vdash \\
A \otimes B \vdash \neg (\neg A \oplus \neg B)
\end{array}
\]

defines exactly the same circuit. The map from derivations to circuits is many-to-one.
Notice that the inductive construction of proof circuits provides for a difference for ⊕ and ⊗ rules. The nodes ⊗I and ⊕E combine different proof circuits, and ⊗E and ⊕I attach to a single proof circuit. This means that ⊗E and ⊕I are parasitic. They do not constitute a proof by themselves. (There is no linear derivation that consists merely of the step ⊕R, or solely of ⊗L, since all axioms are of the form A ⊩ A.) This is unlike ⊕L and ⊗R which can make fine proofs on their own.

Not everything that you can make out of the basic nodes is a circuit corresponding to a derivation. Not every “circuit” (in the broad sense) is inductively generated.

![Diagram of proof circuits](image)

You can define ‘circuits’ for A ⊕ B ⊩ A ⊗ B or A ⊗ B ⊩ A ⊕ B, or even worse, ⊩, but there are no derivations for these sequents. What makes an assemblage of nodes a proof?

### 5.5 | CORRECT CIRCuits

When is a circuit inductively generated? There are different correctness criteria for circuits. Here are two:

The notion of a switching is due to Vincent Danos and Laurent Regnier [18], who applied it to give an elegant account of correctness for proofnets.

**Definition 5.8** (Switched nodes and switchings) The nodes ⊗E and ⊕I are said to be switched nodes: the two output wires of ⊗E are its switched wires and the two input wires of ⊕I are its switched wires. A switching of a switched node is found by breaking one (and one only) of its switched wires. A switching of a circuit is found by switching each of its switch nodes.

Consider this circuit for the associativity of ⊗.

![Diagram of associativity circuit](image)
It has four switchings, with two possibilities for each $\otimes E$ node. Here are two of them.

---

**Theorem 5.9** [Switching Criterion] A circuit is inductively generated if and only if each of its switchings is a tree.

We call the criterion of “every-switching-being-a-tree” the *switching criterion*. There are two ways to fail it. First, by having a switching that contains a loop. Second, by having a switching that contains two disconnected pieces.

**Proof:** The left-to-right direction is a straightforward check of each inductively generated circuit. The basic inductively generated circuits (the single wires) satisfy the switching criterion. Then show that for each of the rules, if the starting circuits satisfy the switching criterion, so do the result of applying the rules. This is a straightforward check.

The right-to-left direction is another thing entirely. We prove it by introducing a new criterion.

---

The new notion is the concept of *retractability* [18].

**Definition 5.10** [Retractions] A single-step retraction of a circuit [node with a single link to another node, both unswitched, retracts into a single unswitched node], and a [a switched node with its two links into a single unswitched node.]... A circuit $\pi'$ is a retraction of another circuit $\pi$ if there is a sequence $\pi = \pi_0, \pi_1, \ldots, \pi_{n-1}, \pi_n = \pi'$ of circuits such that $\pi_{i+1}$ is a retraction of $\pi_i$.

**Theorem 5.11** [Retraction Theorem] If a circuit satisfies the switching criterion then it retracts to a single node.

**Proof:** This is a difficult proof. Here is the structure: it will be explained in more detail in class. Look at the number of switched nodes in your circuit. If you have none, it’s straightforward to show that the circuit is retractable. If you have more than one, choose a switched node, and look at the subcircuit of the circuit which is strongly attached to the...
switched ports of that node (this is called *empire* of the node). This satisfies the switching criterion, as you can check. So it must either be retractable (in which case we can retract away to a single point, and then absorb this switched node and continue) or we cannot. If we cannot, then look at this subcircuit. It must contain a switched node (it would be retractable if it didn’t), which must also have an empire, which must be not retractable, and hence, must contain a switched node ... which is impossible.

**Theorem 5.12 [Conversion Theorem]** If a circuit is retractable, it can be inductively generated.

You can use the retraction process to convert a circuit into a derivation. Take a circuit, and replace each unswitched node by a derivation. The circuit on the left becomes the structure on the right:

In this diagram, the inhabitants of a (green) rectangle are derivations, whose concluding sequent mirrors exactly the arrows in and out of the box. The in-arrows are on the left, and the out-arrows are on the right. If we have two boxes joined by an arrow, we can merge the two boxes. The effect on the derivation is to Cut on the formula in the arrow. The result is in Figure 5.4. After absorbing the two remaining derivations, we get a structure with only one node remaining, the switched $\otimes E$ node. This is in Figure 5.5.

Now at last, the switched node $\otimes E$ has both output arrows linked to the one derivation. This means that we have a derivation of a sequent with both $A$ and $B$ on the left. We can complete the derivation with a $[\otimes L]$ step. The result is in Figure 5.6.
Figure 5.4: A retraction in progress: part 1

Figure 5.5: A retraction in progress: part 2
In general, this process gives us a weird derivation, in which every connective rule, except for $\otimes L$ and $\oplus R$ occurs at the top of the derivation, and the only other steps are Cut steps and the inferences $\otimes L$ and $\oplus R$, which correspond to switched nodes.

\[
\begin{array}{c}
\neg A \vdash \neg A \\
\neg A \vdash \neg B \\
\neg A \oplus B \vdash \neg A, \neg B \\
\end{array}
\]

Figure 5.6: A derivation for $A \otimes B \vdash \neg (\neg A \oplus B)$

Notice that there is no explicit conception of discharge in these circuits. Nonetheless, conditionals may be defined using the vocabulary we have at hand: $A \to B$ is $\neg A \oplus B$. If we consider what it would be to eliminate $\neg A \oplus B$, we see that the combination of a $\oplus E$ and a $\neg E$ node gives us the effect of $\to E$, with $A$ and $\neg A \oplus B$ coming in as inputs, and $B$ the only output.

For an introduction rule, we can see that if we have a proof $\pi$ with a premise $A$ and a conclusion $B$ (possibly among other premises $X$ and conclusions $Y$) we may plug the $A$ input wire into a $\neg I$ node to give us a new $\neg A$ concluding wire, and the two conclusions $\neg A$ and $B$ may be wired up with a $\oplus I$ node to give us the new conclusion $\neg A \oplus B$, or if you prefer, $A \to B$.
In this way, circuit rules give us the effect of discharge, without any mysterious ‘action at a distance’ like we have in traditional natural deduction. The discharge of premises at the top of a proof is explicitly connected to the site at which the discharge takes effect—an arrow links discharged premise with the introduced formula.

5.6 NORMAL CIRCUITS

Not every circuit is normal. In a natural deduction proof in the system for implication, we said that a proof was normal if there is no step introducing a conditional \( A \to B \) which then immediately serves as a major premise in a conditional elimination move. The definition for normality for circuits is completely parallel to this definition. A circuit is normal if and only if no wire for \( A \land B \) (or \( A \lor B \) or \( \neg A \)) is both the output of a \( \otimes I \) node (or a \( \otimes I \) or \( \neg I \) node) and an input of a \( \otimes E \) node (or \( \otimes E \) or \( \neg E \)). This is the role of the small dots on the boundary of the nodes: these mark the ‘active wire’ of a node, and a non-normal circuit has a wire that has this dot at both ends.

It is straightforward to show that if we have a Cut-free derivation of a sequent \( X \vdash Y \), then the circuit constructed by this derivation is normal. The new nodes at each stage of construction always have their dots facing outwards, so a dot is never added to an already existing wire. So, Cut-free derivations construct normal circuits.

The process of normalising a circuit is simplicity itself: Pairs of introduction and elimination nodes can be swapped out by node-free wires in any circuit in which they occur. The square indicates the “active” port of the node, and if we have a circuit in which two active ports are joined, they can “react” to simplify the circuit. The rules of reaction are presented in Figure 5.7.

The process of normalisation is completely local. We replace a region of the circuit by another region with the same periphery. At no stage do any global transformations have to take place in a circuit, and so, normalisation can occur in parallel. It is clearly terminating, as we delete nodes and do not add them. Furthermore, the process of normalisation is confluent. No matter what order we decide to process the nodes, we will always end with the same normal circuit in the end.

Here is an example non-normal circuit:
We can normalise it by first snipping out the $\oplus I/\oplus E$ pair in the centre. The result is this circuit.

This circuit is not yet normal. We now have a $\otimes I/\otimes E$ pair. This can be eliminated in another step:

and the result is a normal circuit, with fewer nodes than the original.

**Theorem 5.13 [Normalisation for Linear Circuits]** Any non-normal circuit can be converted into a normal one by a series of normalisation steps. These steps may be processed in any order. The result is a normal
circuit, with fewer nodes than the original circuit, found in finitely many steps.

5.7 | CLASSICAL CIRCUITS

To design circuits for classical logic, you must incorporate the effect of the structural rules in some way. The most straightforward way to do this is to introduce (switched) contraction nodes and (unswitched) weakening nodes. In this way the parallel with the sequent system is completely explicit.

The first account of multiple-conclusion proofs is Kneale’s “Tables of Development” [43]. Shoesmith and Smiley’s Multiple Conclusion Logic [78] is an extensive treatment of the topic. The authors explain why Kneale’s formulation is not satisfactory due to problems of substitution of one proof into another — the admissibility of cut. Shoesmith and Smiley introduce a notation similar to the node and wire diagrams used here. The problem of substitution is further discussed in Ungar’s Normalization, Cut-elimination, and the theory of proofs [84], which proposes a general account of what it is to substitute one proof into another. One account of classical logic that is close to the account given here is Edmund Robinson’s, in “Proof Nets for Classical Logic” [74].

DEFINITION 5.14 [CLASSICAL CIRCUITS] These are the inductively generated classical circuits:

- An identity wire: $\lambda A \cdot A$ for any formula $A$ is an inductively generated circuit. The sole input type for this circuit is $A$ and its output type is also (the very same instance) $A$. As there is only one wire in this circuit, it is near to itself.

- Each boolean connective node presented below is an inductively generated circuit. The negation nodes:

  $\neg E$ $\neg I$

The conjunction nodes:

$\land I_1$ $\land E_1$ $\land E_2$

And disjunction nodes:

$\lor I_1$ $\lor I_2$ $\land I$
The inputs of a node are those wires pointing into the node, and the outputs of a node are those wires pointing out.

- Given an inductively generated circuit $\pi$ with an output wire labelled $A$, and an inductively generated circuit $\pi'$ with an input wire labelled $A$, we obtain a new inductively generated circuit in which the output wire of $\pi$ is plugged in to the input wire of $\pi'$. The output wires of the new circuit are the output wires of $\pi$ (except for the indicated $A$ wire) and the output wires of $\pi'$, and the input wires of the new circuit are the input wires of $\pi$ together with the input wires of $\pi'$ (except for the indicated $A$ wire).

- Given an inductively generated circuit $\pi$ with two input wires $A$, a new inductively generated circuit is formed by plugging both of those input wires into the input contraction node $\downarrow$. Similarly, two output wires with the same label may be extended with a contraction node $\uparrow$.

- Given an inductively generated circuit $\pi$, we may form a new circuit with the addition of a new output, or output wire (with an arbitrary label) using a weakening node $\text{KI}$ or $\text{KE}$.  

Here is an example circuit:

---

Using an unlinked weakening node like this makes some circuits disconnected. It also forces a great number of different sequent derivations to be represented by the same circuit. Any derivation of a sequent of the form $X \vdash Y, B$ in which $B$ is weakened in at the last step will construct the same circuit as a derivation in which $B$ is weakened in at an earlier step. If this identification is not desired, then a more complicated presentation of weakening, using the ‘supporting wire’ of Blute, Cockett, Seely and Trimble [10] is possible. Here, I opt for a simple presentation of circuits rather than a comprehensive account of “proof identity.”
This circuit essentially uses two contraction nodes in order to provide two proofs of \(\neg(p \land \neg p)\) to provide as inputs for the final \(\land I\) node. Here is another example circuit, in which a proof from \(p \lor p\) to \(p\) (the left half of the circuit) is composed with a proof from \(p\) to \(p \land p\). Notice that the corresponding derivation uses a \textit{Cut} step, despite the fact that the circuit does not ever feature a formula which is introduced and then eliminated. Regardless, this proof is not \textit{normal} because the two contraction nodes are cut against one another.

This circuit can normalise in two ways, corresponding to whether or not the cut step (From \(p \lor p\) to \(p\) and \(p \land p\)) It has two derivations which composed. Either we take the proof from \(p \lor p\) to \(p\) and push it to the right, past the duplicating \(\land I\) node, thereby duplicating that part of the proof:

and we take the proof from \(p\) to \(p \land p\) and push it leftwards, past the \(\lor E\) node \(\lor\), thereby duplicating that proof:

In either case, the circuit that results is normal, but both circuits are quite different. They are non-equivalent proofs. This is an example of how normalisation for classical circuits is a difficult matter. The traditional account of normalisation is non-confluent. The original proof can normalise in two different ways.

5.8 | HISTORY AND OTHER MATTERS

We can rely on the duality of \(\otimes\) and \(\oplus\) to do away with half of our rules, if we are prepared to do a little bit of work. Translate the sequent \(X \vdash Y\) into \(\vdash \neg X, Y\vdash\), and then trade in \(\neg(A \otimes B)\) for \(\neg A \oplus \neg B; \neg(A \oplus B)\) for
\( \neg A \otimes \neg B, \) and \( \neg \neg A \) for \( A \). The result will be a sequent where the only negations are on atoms. Then we can have rules of the following form:

\[
\begin{align*}
\vdash p, \neg p & \quad \text{[Id]} \\
\vdash X, A, B & \quad \text{[\text{\&}R]} \\
\vdash X, A \oplus B & \quad \text{[\text{\&}R]} \\
\vdash X, A & \vdash X', B & \vdash X, X' & \vdash A \otimes B
\end{align*}
\]

The circuits that result on this picture are also much simpler. They only have outputs and no inputs. These are Girard’s proofnets [32]. They are very useful when it comes to the analysis of the structure of derivations. They are less successful as a model of the surface structure of a process of reasoning from premises to a conclusion. For that, it seems that the directed structure of circuits, with inputs and outputs, serves better.

### 5.9 | Exercises

#### Basic exercises

Q1 Which of the following sequents can be proved in the sequent calculus for intuitionistic logic? For those that can, find a derivation. For those that cannot, find a derivation in classical sequent calculus:

1. \( p \rightarrow (q \rightarrow p \land q) \)
2. \( \neg (p \land \neg p) \)
3. \( p \lor \neg p \)
4. \( (p \rightarrow q) \rightarrow ((p \land r) \rightarrow (q \land r)) \)
5. \( \neg \neg \neg p \rightarrow \neg p \)
6. \( \neg (p \lor q) \rightarrow (\neg p \land \neg q) \)
7. \( (p \land (q \rightarrow r)) \rightarrow (q \rightarrow (p \land r)) \)
8. \( p \lor (p \rightarrow q) \)
9. \( (\neg p \lor q) \rightarrow (p \rightarrow q) \)
10. \( ((p \land q) \rightarrow r) \rightarrow ((p \rightarrow r) \lor (q \rightarrow r)) \)

Q2 Consider all of the formulas unprovable in Q1 on page 46. Find derivations for these formulas in the classical sequent System 1.

Q3 Fill in the reasoning showing that the rules for System 2 (given in Figure 5.2) are no stronger and no weaker than those in System 1. Look at a few example connective rules, and do with them what we do in Lemma 5.5. Then take a derivation of some sequent in System 1, and convert it into a derivation in System 2. Do the same for the derivation of some different sequent in System 2: convert it into a derivation in System 1.

Q4 Define the dual of a classical sequent in a way generalising the result of Exercise 15 on page 69, and show that the dual of a derivation of a sequent is a derivation of the dual of a sequent. What is the dual of a formula involving implication?

Q5 Define \( A \rightarrow^* B \) as \( \neg (A \land \neg B) \). Show that any classical derivation of \( X \vdash Y \) may be transformed into a classical derivation of \( X^* \vdash Y^* \), where
X* and Y* are the multisets X and Y respectively, with all instances of the connective $\rightarrow$ replaced by $\rightarrow^\ast$. Take care to explain what the transformation does with the rules for implication. Does this work for intuitionistic derivations?

Q6 Consider the rules for classical propositional logic in Figure 5.1. Delete the rules for negation. What is the resulting logic like? How does it differ from intuitionistic logic, if at all?

Q7 Define the Double Negation Translation $d(A)$ of formula A as follows:

\[
\begin{align*}
d(p) &= \neg \neg p \\
d(\neg A) &= \neg d(A) \\
d(A \land B) &= d(A) \land d(B) \\
d(A \lor B) &= \neg(\neg d(A) \land \neg d(B)) \\
d(A \rightarrow B) &= d(A) \rightarrow d(B)
\end{align*}
\]

What formulas are $d((p \rightarrow q) \lor (q \rightarrow p))$ and $d(\neg\neg p \rightarrow p)$? Show that these formulas have intuitionistic proofs by giving a sequent derivation for each.

Q8 Construct circuits for the following sequents:

1. $\vdash p \oplus \neg p$
2. $p \otimes \neg p \vdash$
3. $\neg\neg p \vdash p$
4. $p \vdash \neg\neg p$
5. $\neg(p \otimes q) \vdash \neg p \oplus \neg q$
6. $\neg p \oplus \neg q \vdash \neg p \otimes q$
7. $\neg(p \otimes q) \vdash \neg p \otimes \neg q$
8. $p \otimes \neg q \vdash (p \otimes q)$
9. $p \otimes q \vdash q \otimes p$
10. $p \oplus (q \otimes r) \vdash p \oplus (q \otimes q)$
11. $p \oplus (q \otimes r) \vdash q \otimes (r \otimes p)$
12. $\neg p \oplus q \vdash \neg(p \otimes \neg q)$
13. $\neg p \oplus q \vdash (\neg r \otimes q) \oplus (r \otimes \neg p)$.

Q9 Show that every formula $A$ in the language $\oplus, \otimes, \neg$ is equivalent to a formula $n(A)$ in which the only negations are on atomic formulas.

Q10 For every formula $A$, construct a circuit $\text{encode}_A$ from A to $n(A)$, and $\text{decode}_A$ from $n(A)$ to A. Show that $\text{encode}_A$ composed with $\text{decode}_A$ normalises to the identity arrow $\xymatrix{A \ar[r] & A}$, and that $\text{decode}_A$ composed with $\text{encode}_A$ normalises to $\xymatrix{n(A) \ar[r] & n(A)}$. (If this doesn’t work for the encode and decode circuits you chose, then try again.)

Q11 Given a circuit $\pi_1$ for $A_1 \vdash B_1$ and a circuit $\pi_2$ for $A_2 \vdash B_2$, show how to construct a circuit for $A_1 \otimes A_2 \vdash B_1 \otimes B_2$ by adding two more nodes. Call this new circuit $\pi_1 \otimes \pi_2$. Now, suppose that $\tau_1$ is a proof from $B_1$ to $C_1$, and $\tau_2$ is a proof from $B_2$ to $C_2$. What is the relationship between the proof $(\pi_1 \otimes \pi_2) \cdot (\tau_1 \otimes \tau_2)$ (composing the two proofs $\pi_1 \otimes \pi_2$ and $\tau_1 \otimes \tau_2$ with a Cut on $B_1 \otimes B_2$) from $A_1 \otimes A_2$ to $C_1 \otimes C_2$ and the proof $(\pi_1 \otimes \tau_1) \otimes (\pi_2 \otimes \tau_2)$, also from $A_1 \otimes A_2$ to $C_1 \otimes C_2$?
Prove the same result for $\oplus$ in place of $\otimes$. Is there a corresponding fact for negation?

Q12 Re-prove the results of all of the previous questions, replacing $\otimes$ by $\land$ and $\oplus$ by $\lor$, using the rules for classical circuits. What difference does this make?

Q13 Construct classical circuits for the following sequents

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$q \vdash p \lor \neg p$</td>
</tr>
<tr>
<td>2</td>
<td>$p \land \neg p \vdash q$</td>
</tr>
<tr>
<td>3</td>
<td>$p \vdash (p \land q) \lor (p \land \neg q)$</td>
</tr>
<tr>
<td>4</td>
<td>$(p \land q) \lor (p \land \neg q) \vdash p$</td>
</tr>
<tr>
<td>5</td>
<td>$(p \land q) \lor r \vdash p \land (q \lor r)$</td>
</tr>
<tr>
<td>6</td>
<td>$p \land (q \lor r) \vdash (p \land q) \lor r$</td>
</tr>
</tbody>
</table>

Q14 Look at this:

There is no linear sequent derivation of $(p \otimes q) \oplus r \vdash p \otimes (q \oplus r)$ (it is not truth-table valid, for a start). Why is this circuit not correct? Use both the switching criterion and the retracting criterion to explain why this is not a correct circuit.

**INTERMEDIATE EXERCISES**

Q15 Using the double negation translation $d$ of the previous question, show how a classical derivation of $X \vdash Y$ may be transformed (with a number of intermediate steps) into an intuitionistic derivation of $X^d \vdash Y^d$, where $X^d$ and $Y^d$ are the multisets of the $d$-translations of each element of $X$, and of $Y$ respectively.

Q16 Construct a system of rules for intuitionistic logic with as similar as you can to the classical system in Figure 5.2. Is it quite as nice? Why, or why not?

Q17 Relate Cut-free sequent derivations of $X \vdash Y$ with tableaux refutations of $X, \neg Y$ [42, 71, 79]. Show how to transform any Cut-free sequent derivation of $X \vdash Y$ into a corresponding closed tableaux, and vice-versa. What are the differences and similarities between tableaux and derivations?

Q18 Relate natural deduction proofs for intuitionistic logic in Gentzen–Prawitz style with natural deduction proofs in other systems (such as Lemmon [46], or Fitch [26]). Show how to transform proofs in one system into proofs in the other. How do the systems differ, and how are they similar?
q19 The following statement is a tautology:

\[\neg((p_{1,1} \lor p_{1,2}) \land (p_{2,1} \lor p_{2,2}) \land (p_{3,1} \lor p_{3,2}) \land \\
\neg(p_{1,1} \land p_{2,1}) \land \neg(p_{1,1} \land p_{3,1}) \land \neg(p_{2,1} \land p_{3,1}) \land \neg(p_{1,2} \land p_{3,2}) \land \neg(p_{2,2} \land p_{3,2}))\]

It is the pigeonhole principle for \( n = 2 \). The general pigeonhole principle is the formula \( P_n \).

\[P_n: \neg\left(\bigwedge_{i=1}^{n+1} p_{i,j} \land \bigwedge_{i=1}^{npg+1} p_{i',j'} \land \bigwedge_{i'=i+1}^{n+1} \neg(p_{i,j} \land p_{i',j'})\right)\]

\( P_n \) says that you cannot fit \( n + 1 \) pigeons in \( n \) pigeonholes, no two pigeons in the one hole. Find a proof of the pigeonhole principle for \( n = 2 \). How large is your proof? Describe a proof of \( P_n \) for each value of \( n \). How does the proof increase in size as \( n \) gets larger? Are there non-normal proofs of \( P_n \) that are significantly smaller than any non-normal proofs of \( P_n \)?

ADVANCED EXERCISES

q20 Consider what sort of rules make sense in a sequent system with sequents of the form \( A \vdash X \), where \( A \) is a formula and \( X \) a multiset. What connectives make sense? (One way to think of this is to define the dual of an intuitionistic sequent, in the sense of Exercise 4 in this section and Exercise 15 on page 69.)

This defines what Igor Urbas has called 'Dual-intuitionistic Logic' [85].
REFERENCES


[18] VINCENT DANOS AND LAURENT REGNIER. “The Structure of Multipli-


[27] MICHAEL J. GABBY AND MURDOCH J. GABBY. “Some Formal Considera-


