

2.4. Using lemmas

2.4.0. Overview

Although our system of derivations as it stands is both sound and complete, we will add rules that reflect the use of lemmas, both because of the importance of lemmas in ordinary explicit deductive reasoning and because the sorts of organization and simplification they provide in that context are of value here, too.

2.4.1. The dangers of lemmas

Although the use of lemmas is valuable in general, not all individual lemmas are valuable: the uncontrolled use of lemmas can lead us into blind alleys or delay the progress of a derivation.

2.4.2. Lemmas for *reductio* arguments

A lemma that is entailed by our goal is safe (though not necessarily progressive); this means that any lemma is safe when the goal is \perp .

2.4.3. Attachment rules

Lemmas are, of course, safe when we know we can prove them. We will use such lemmas to add to the available resources. The sentences added will generally be more complex than those already present, so this use of lemmas can interfere with decisiveness.

Glen Helman 25 Aug 2005

2.4.1. The dangers of lemmas

A fully general rule for introducing lemmas was cited in 2.3.2 as an example of an unsafe rule because argument from our resources to the lemma might fail even though the proximate argument of the gap was valid. Such would also prevent a system from being decisive because it would always be possible to develop a gap further by introducing a lemma. However, as was noted earlier, more limited rules for introducing lemmas can be safe, and we will see that they can also be progressive. In this section we will look at the problems posed by lemmas more closely before consider a couple of special cases where they do not arise.

The **law for lemmas** of 1.4.2 can be stated as follows:

$\Gamma \Rightarrow \phi$ if both $\Gamma \Rightarrow \psi$ and $\Gamma, \psi \Rightarrow \phi$

Any possible world that is a counterexample to the first entailment will be a counterexample to one of the two on the right—the first of them if it makes ψ false and the second if it makes it true. So if both entailments on the right hold (that is, neither has a counterexample), then the one on the left will hold, too. But this principle holds only as an *if* claim; the corresponding *only if* does not hold in all cases. When $\Gamma \Rightarrow \phi$, we know that $\Gamma, \psi \Rightarrow \phi$ by the **monotonicity of \Rightarrow** ; but, since ϕ and ψ need have no connection with one another, knowing that $\Gamma \Rightarrow \phi$ would by itself give us no reason to suppose that $\Gamma \Rightarrow \psi$. Of course, in a case where we know that that $\phi \Rightarrow \psi$, we would know $\Gamma \Rightarrow \psi$ because of the **chain law** and there are other cases where would know $\Gamma \Rightarrow \psi$ because of special connections between Γ and ψ . These are the two sorts of cases in which we will use lemmas but, before turning to them, let's look at what a fully general rule for lemmas would be like.

If used in tree-form proofs this rule would take the following form:

$$\text{Lem} \frac{\psi \quad \boxed{\begin{array}{c} \psi \\ \vdots \\ \phi \end{array}}}{\phi}$$

Here the proof of ϕ divides into two branches. The first is a proof of the lemma ψ and the second is a proof ϕ using ψ in addition to

the already available assumptions and the exploitation chains that grow from them. The box around the right-hand branch is intended to indicate that the use of ψ is limited to that part of the proof and ψ 's presence at the upper left is intended to indicate that it is available for this branch as a further assumption from which we may begin exploitation chains.

In the notation of derivations, we use scope lines to mark the scope of added assumptions, which are marked off from other resources along a scope line by the sort of horizontal line we use to indicate the premises of the ultimate argument.

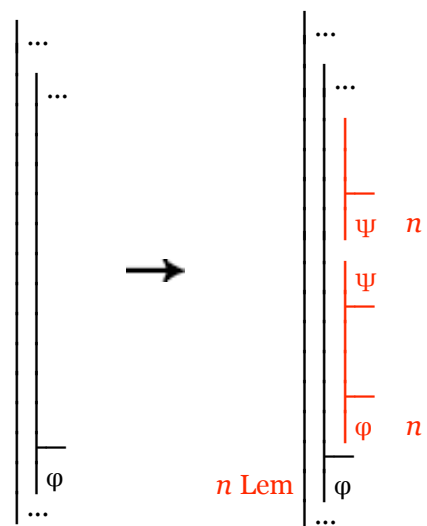


Fig. 2.4.1-1. Developing a derivation by introducing a lemma at stage n (a rule that will be part of our systems of derivations only in more restricted forms).

The assumption ψ is available only as long as the second of the two short scope lines continues, so it is effectively boxed off in derivations just as it is in the rule above for tree-form proofs.

The second of the two new gaps should be, if anything, easier to close than the original gap because it has a further resource. This increased ease is the point of introducing a lemma. The price we pay for this is the need to close the first new gap also. If the lemma is properly chosen, that may also be easier than closing the original gap but, because this rule is unsafe, we cannot be sure in general that the first new gap can be closed at all even if the original one

can be in other ways. Because of this, when lemmas are introduced in ordinary deductive reasoning we must be prepared to backtrack, to abandon the attempt to work by way of the lemma and look for another approach to the proof. The notation of derivations is not designed to incorporate backtracking, so we will use lemmas only in cases where we can be sure there will be no need to do that.

Even in cases where we can sure backtracking is not necessary the introduction of lemmas can interfere with decisiveness because there may be enough safe lemmas to keep introducing them forever. So our restrictions of the rule Lem will be more severe than would be required merely to insure that the lemma it introduces is safe.

Glen Helman 25 Aug 2005

2.4.2. Lemmas for *reductio* arguments

We have seen that a lemma is bound to be safe if it is entailed by the goal we seek. That is, we can state following principle:

if $\varphi \Rightarrow \psi$, then an interpretation divides Γ from φ if and only if either if it divides Γ from ψ or it divides Γ together with ψ from φ

which tells us that when $\varphi \Rightarrow \psi$, it is both sound and safe to introduce a lemma ψ in a derivation whose goal is φ .

In order to apply this idea, we can look for appropriate choices of φ and ψ in valid single-premised arguments. The obvious arguments among those we have identified so far are EFQ and the two forms of Ext. Although EFQ will prove to be the more important, Ext is a better source of examples at the moment and we will consider it first. Here is a derivation which uses the rule Lem to introduce a lemma that is the result of applying left Ext to the final goal.

	$A \wedge B$	1
1 Ext	A	(5),(9)
1 Ext	B	(4)
	•	
4 QED	B	3
	•	
5 QED	A	3
3 Cnj	B \wedge A	2
	B \wedge A	(7),(10)
	•	
7 QED	B \wedge A	6
	•	
9 QED	A	8
	•	
10 QED	B \wedge A	8
8 Cnj	A \wedge (B \wedge A)	6
6 Cnj	(B \wedge A) \wedge (A \wedge (B \wedge A))	2
2 Lem	(B \wedge A) \wedge (A \wedge (B \wedge A))	

Here the rule Lem is applied at stage 2 with the left component of the goal as the lemma. This yields a slight shortening of the derivation since we are able to use the lemma to conclude $B \wedge A$ by QED at stages 7 and 9 rather than repeating the proof used at stages 3-5 twice.

The simplification here is slight and it occurs at all only because of a repetition in the goal that we would not expect to encounter often. While we would have more opportunities to use this sort of lemma in later chapters, there would not be enough to lead us to introduce a special rule and this will serve us only as an initial example. It is worth remembering, however, that it is legitimate pattern of deductive reasoning to conclude one of the two components of a conjunction and then use that component to conclude the other (as we here have used the lemma $B \wedge A$ in concluding $A \wedge (B \wedge A)$).

The pattern *Ex Falso Quodlibet* provides the basis for a much more important use of lemmas. An argument whose conclusion is \perp is often called a **reductio argument**; *reductio* here is short for the Latin phrase *reductio ad absurdum* ('reduction to absurdity'). We will often need to use a lemma to complete such an argument and, since EFQ tells us that \perp entails any sentence, we know that any lemma we choose is safe. We will call the rule implementing this idea **Lemma for Reductio** or LFR:

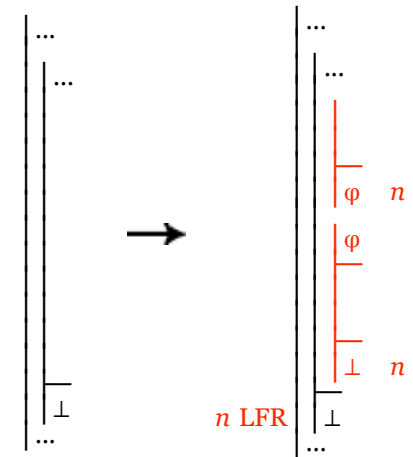


Fig. 2.4.2-1. Developing a derivation by introducing a lemma for a *reductio* at stage n .

We know this is safe from earlier arguments, but it is also easy to see that directly. Any interpretation that divided either of the new gaps would certainly have to make all active resources of the original gap true; but an interpretation that did that would divide the original gap since its goal \perp is bound to be false. So neither of the two new gaps divided unless their parent was.

While this rule is certainly important, we are not yet in a position to illustrate it because, as yet, we have no non-trivial examples of formally valid *reductio* arguments. A *reductio* is formally valid only if its premises constitute a formally inconsistent set (that is, one whose members cannot be all true on any extensional interpretation) and the only formally inconsistent sets available with our current analyses of sentences contain \perp either as a premise or as a component of one. And such set a can be shown to entail \perp with use of nothing but Ext and QED. This situation will change in the next chapter but, even there, our chief use of lemmas will be in a special modified version of this rule that is designed to actually exploit resources.

That rule will be direct but rules that introduce lemmas usually will not be and, in order to be sure a system employing them was decisive we would need to show that they could be considered progressive (on the right measure of distance from the end). The use of Lem to introduce a component of the goal can be regarded as progressive provided we require that the lemma is not already an active resource. But the free use of LFR would undermine decisiveness even if we forbid such repetition since the form of the rule places no constraints on the number of different lemmas that might be introduced. Something like a limitation to components of active resources and goals would be sufficient but more minimal restrictions would also work. In general, we will not attempt to formulate the sort of restriction that would enable us to prove decisiveness for a system with LFR. The value of the rule is a practical one and in practice the constraint of good sense in its use is restrictive enough.

Glen Helman 25 Aug 2005

2.4.3. Attachment rules

When discussing the minimal soundness of QED in 2.3.2 we saw that it would be legitimate for a rule to close a gap when its goal is not among its active resources—or even among the active resources of its ancestor gaps—provided it is entailed by available resources. We will not employ such a sweeping rule but we will extend the use of QED (and later rules which use inactive resources) by rules which add to the available resources of a gap without changing either its active resources or its goal. An example is the following way of developing a gap, which we will call **Adjunction**:

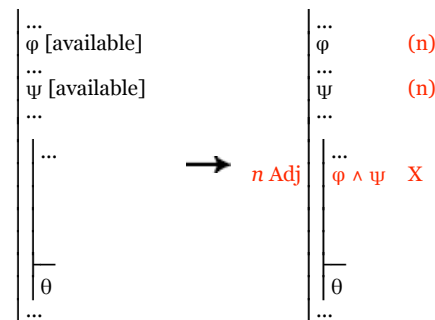


Fig. 2.4.3-1. Developing a derivation by applying Adj at stage n .

The added conjunction functions as a lemma so this rule represents a way of using lemmas but it has a number of special features both by comparison with a rule like LFR and by comparison with other rules we have seen. The lemma $\varphi \wedge \psi$ does not lie to the right of a new scope line, as it does in the second gap introduced by LFR, for two reasons. First, we have not branched the gap so the added resource is available throughout the gap. And, second, we do not need to mark this new resource off as an added assumption because it is entailed by those already present. Notice also that we treat this rule not as a way to plan for our goal but simply as a way to add resources. However, it does not exploit resources in order to add others and the X to the right of $\varphi \wedge \psi$ is intended to indicate that this resource need not be exploited further. One way to think about this is to suppose that $\varphi \wedge \psi$ has been introduced as something already exploited. That is, although it need not have been a once active but exploited resource (and there would be no point in adding it if it was) it has a status similar to such resources.

Adjunction is one example of a group of rules we will refer to as **attachment rules**. Any such rule R will exhibit the following general pattern.

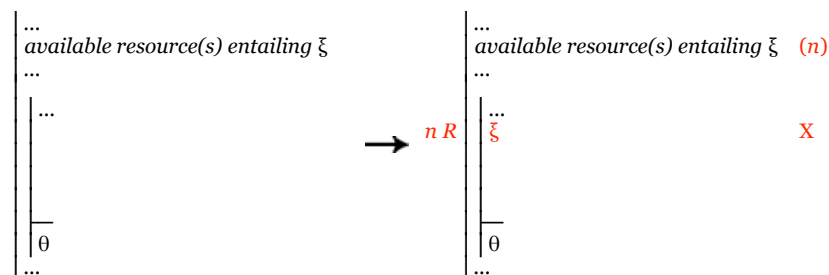


Fig. 2.4.3-2. Developing a derivation by applying an attachment rule R at stage n .

Since the lemma is not an active resource, the proximate argument of child gap is the same as the parent's proximate argument so safety and (utter) soundness hold as they

would for a gap that is completely unchanged. The only question concerns the impact of such a rule on the (minimal) soundness of rules like QED that use merely available resources. And the fact that the added resource is entailed by available resources means that if we extend the available resources beyond once active ones only by using attachment rules, all resources available in a gap will be entailed by the active resources of the gap and together with the active resources of its ancestors. Any interpretation that divides a gap and all its ancestors will then make all available resources true and this will be enough for us to establish the soundness of rules using available resources. (To rehearse the argument for QED again: if the goal of a gap is among its available resources no interpretation can divide it and all its ancestors because, to do this, an interpretation would need to make its goal false while making not only its active resources but all its available resources true and this is impossible when the goal is among the available resources.)

Although, as in the case of LFR, the most important constraint on the use of attachment rules will be good sense, a rule like Adj clearly raises questions about decisiveness since the lemma it introduces is more complex than the premises it is based on. This increased complexity will be typical of attachment rules and is the reason for their name. Apart from good sense, the requirement that the lemma be a component of a goal or active resource is a natural one since such cases will represent the most valuable instances of attachment rules. Here, though, we need to remember that a sentence is a component of itself and one common use of these rules will be to introduce the goal itself as an available resource in order to apply QED to close the gap. The following derivation is a simple example of this in the case of Adj.

	A ∧ B	1
	C	(4)
1 Ext	A	(3)
1 Ext	B	(4)
	•	
	A	2
3 QED	B ∧ C	X,(5)
	•	
	B ∧ C	2
4 Adj	B ∧ C	2
	A ∧ (B ∧ C)	
5 QED	A ∧ (B ∧ C)	
2 Cnj	A ∧ (B ∧ C)	

With two uses of Cnj, we would not have needed Adj and, with two uses of Adj, we would not have needed Cnj; but it is this sort of mixed use of the two that brings us closest to typical patterns of explicit deductive argument.

Glen Helman 25 Aug 2005

2.4.s. Summary

The introduction of a lemma is one way of dividing up the work of a proof. We can implement this idea in derivations by dividing a gap into two, one with the lemma as a goal and the other with it as a further assumption to use in reaching the goal of the parent gap. The rule [Lemma \(Lem\)](#) that does this is not safe in general nor is it always progressive, and we will use only special instances of it.

A lemma is always safe when it is entailed by the goal it is designed to help us reach. The principal use of this idea will come in arguments whose goal is \perp —that is, in [reductio arguments](#). Since \perp entails any sentence a rule [Lemma for Reductio \(LFR\)](#) which allows free use of lemmas in *reductio* arguments will be safe (though some restriction on its use is needed to insure it is progressive).

A lemma is also safe if we know we can reach it. Rules applying this idea will be designed for particular sorts of entailment and, since the lemma is known to follow from our resources, there is no need to divide the gap or even introduce a new scope line. Indeed, we will use this sort of lemma only in [attachment rules](#) that add the lemma as an available but inactive resource. The first example of this sort of rule is [Adjunction \(Adj\)](#) which adds a conjunction when both conjuncts are already available. Although attachment rules can help us to close gaps sooner, the rules themselves are not direct so some care is needed in their use if they are to be progressive.

The derivation rules we have so far are summarized in the table below. The names of the rules are links to the point in the text where they were initially described; look there to see the actual form taken by the rule.

Rules for developing gaps

for resources for goals

conjunction
 $\varphi \wedge \psi$

Ext

Cnj

Rules for closing gaps

when to close rule

the goal is also
a resource QED

\top is the goal ENV

\perp is a resource EFQ

Basic system

Attachment rule Added rules

added resource rule (optional)

$\varphi \wedge \psi$ Adj

Glen Helman 25 Aug 2005

2.4.x. Exercise questions

Use the basic system of derivations along with the attachment rule Adj to establish the following. These repeat entailments from earlier exercises and examples (specifically, **b** and **d** of exercise 2.2.x.2, exercises **2** and **4** of 2.3.x, and the example of 2.4.2). They will work best as exercises in the use of Adj if you avoid using Cnj.

1. $A \Rightarrow A \wedge A$
2. $A \wedge B, B \wedge C, C \wedge D \Rightarrow A \wedge D$
3. $A \wedge B \Rightarrow A \wedge (B \wedge A)$
4. $A, B \wedge C, D \Rightarrow (C \wedge (B \wedge A)) \wedge B$
5. $A \wedge B \Rightarrow (B \wedge A) \wedge (A \wedge (B \wedge A))$

Glen Helman 25 Aug 2005

2.4.xa. Exercise answers

The answers below avoid the use of Cnj in order to maximize the use of the rule Adj. In some cases, a mixed use of the two would have produced a more natural argument.

1.

	A	(1)
1 Adj	A \wedge A	X,(2)
	•	
2 QED	A \wedge A	

2.

	A \wedge B	1
	B \wedge C	2
	B \wedge D	3
	—	
1 Ext	A	(4)
1 Ext	B	
2 Ext	B	
2 Ext	C	
3 Ext	B	
3 Ext	D	(4)
4 Adj	A \wedge D	X,(5)
	•	
5 QED	A \wedge D	

3.

	A \wedge B	1
	—	
1 Ext	A	(2),(3)
1 Ext	B	(2)
2 Adj	B \wedge A	X,(3)
3 Adj	A \wedge (B \wedge A)	X,(4)
	•	
4 QED	A \wedge (B \wedge A)	

4.

	A	(2)
	B \wedge C	1
	D	
	—	
1 Ext	B	(2),(4)
1 Ext	C	(3)
2 Adj	B \wedge A	X,(3)
3 Adj	C \wedge (B \wedge A)	X,(4)
4 Adj	(C \wedge (B \wedge A)) \wedge B	X,(5)
	•	
5 QED	(C \wedge (B \wedge A)) \wedge B	

5.

	A \wedge B	1
	—	
1 Ext	A	(2),(3)
1 Ext	B	(2)
2 Adj	B \wedge A	X,(3),(4)
3 Adj	A \wedge (B \wedge A)	X,(4)
4 Adj	(B \wedge A) \wedge (A \wedge (B \wedge A))	X,(5)
	•	
5 QED	(B \wedge A) \wedge (A \wedge (B \wedge A))	