3.3. Negations as premises

3.3.0. Overview

A second group of rules for negation reverses the roles of an affirmative sentence and its negation.

Glen Helman 18 Sep 2004

3.3.1. Indirect proof

The last section pursued consequences of the law for negation as a conclusion. The rules of this section will implement the other basic law for negation, the law for it as a premise:

$$\Gamma, \neg \phi \Rightarrow \bot \text{ if and only if } \Gamma \Rightarrow \phi$$

This says that a negation is $\neg \phi$ inconsistent with a set Γ if and only if the sentence ϕ is entailed by that set.

There are two lessons we can learn from this law. First, the *only-if*-statement tells us that negative conclusions are not the only ones that can be established by way of *reductio* arguments. That is, an entailment $\Gamma \Rightarrow \phi$ can be supported by the *reductio* $\Gamma, \neg \phi \Rightarrow \bot$. The *if*-statement tells us in part that such an approach is safe, that the *reductio* is valid whenever the argument we wish to support by it is valid. But *if*-statement tells us more. Notice that ϕ is just the sort of resource that would enable us to complete a *reductio* that has $\neg \phi$ as a premise. The *if*-claim above tells us that, if a *reductio* with $\neg \phi$ as a premise can be completed at all, we would be able to validly conclude ϕ as a lemma—and that concluding it would not depend on using $\neg \phi$ itself as a premise. This further lesson will provide the basis for exploiting negative resources, but its full application depends on the broader use of *reductio* arguments supported by the other two lessons, and that is what we will consider first.

Here is an example of this broader use. If we take *No one is home* to be the negation $\neg$ *someone is home*, the law for negation as a premise says we can rest the validity of the left-hand argument below on the validity of the right-hand argument.

If no one was out, the car was in the driveway	*If no one was out, the car was in the driveway*
The car wasn't in the driveway	*The car wasn't in the driveway*
	No one was out
———	———
Someone was out	$\bot$

The right-hand argument depends in part on the logical properties of *if*; but, as far as negation is concerned, it depends on only the fact that a sentence and its negation are mutually exclusive.

The same basic fact gives us $\neg \neg \phi, \neg \phi \Rightarrow \bot$. If we apply the law for negation as a premise to this, we get the principle $\neg \neg \phi \Rightarrow \phi$. Moreover, the latter principle can be combined with the law for negation as a conclusion to establish the law for negation as a premise. So the further logical properties of negation that are captured by the law for negation as a premise can be summarized in the principle that a double negation entails the corresponding positive claim.

This principle is one that was rejected by Brouwer in his intuitionistic mathematics. And one of his chief reasons for rejecting it was that it would allow us to draw a conclusion of the form *Something has the property P* when the corresponding claim *Nothing has the property P* was inconsistent with our premises—that is, just the sort of thing done in the example above. His concern with this is that it would enable us to conclude *Something has the property P* in cases where we were unable, even in principle, to provide an actual example of a thing with that property *P*. Brouwer did not object to such an argument in ordinary reasoning about the physical world (like the example above); but he held that, in reasoning concerning infinite mathematical structures, we were not reasoning about an independently existing realm of objects but instead about procedures for constructing abstract objects and that we had no business claiming the existence of such objects without having procedures enabling us to construct them. Brouwer's concerns may not lead you to question the law for negation as a premise; but they highlight the indirectness of basing a positive conclusion on the fact that its denial is inconsistent with our premises. This aspect of these arguments is reflected in a common term for them, ***indirect proofs***.

Although we will employ indirect proofs, we will need them for only a limited range of conclusions. We have other ways of planning for a goal that is a conjunction or a negation. We can simply close a gap whose goal is ⊤. And we will not adopt any rule to plan for the goal ⊥ of a *reductio argument*. At the moment, that leaves only unanalyzed components; and, until the last chapter, those are the only goals for which we will use indirect proofs. We have often closed gaps whose goals are atomic so, even for them, we know that indirect proof is not always necessary. However, it will serve us as a last resort.

In chapter 6, we will begin to analyze sentences into components that are not sentences, and we will still use indirect proof for goals that are analyzed in that way. In anticipation of this, we will use the term ***atomic*** for the kind of goals to which we will apply indirect proof; and we will refer to other sentences as ***non-atomic***. Until chapter 6, any sentence we analyze will be compounds formed by applying a connective to one or more sentences, so, for the time being, the atomic sentences will be the unanalyzed sentences. ⊤ and ⊥ count as non-atomic since identifying them as logical constants counts as an analysis of their logical form. As a result, for the time being, the atomic sentences will be simple letters, and all other sentences will be non-atomic.

The rule implementing indirect proofs in derivations will be called ***Indirect Proof*** (IP). It takes the following form:

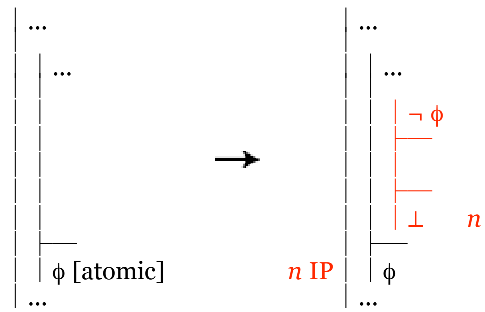


Fig. 3.3.1-1. Developing a derivation by planning for an atomic sentence at stage *n*.

Here is an example, which is related to the argument at the beginning of 3.2.2.

	¬ ((A ∧ B) ∧ ¬ C)	(4)
	A	(2)
	B	(2)
	¬ C	(3)
2 Adj	A ∧ B	X,(3)
3 Adj	(A ∧ B) ∧ ¬ C	X,(4)
	•	
4 Nc	⊥	1
1 IP	C	

This example adds to the premise *Ann and Bill were not both home without the car being in the driveway* further premises telling us that each of Ann and Bill was home, and we conclude that the car was in the driveway. Although the initial premises and conclusion differ from those of the argument in 3.2.2, the *reductio* argument that is set up at stage 1 here has the same resources as the *reductio* set up at stage 3 in the derivation for the argument of 3.2.2 that was given at the end of 3.2.3.

The rule IP is not direct (in the sense used in 2.3.3) because it introduces a sentence more complex than the goal it plans for. It is, however, progressive. We will treat both atomic sentences and their negations as equally basic when they are resources: neither sort of resource will be exploited. And, as was noted above, we will treat ⊥ as the basic form of goal, the only one without a corresponding planning rule. Thus IP leaves us with a goal that requires no planning and introduces no resources that need to be exploited further. This is, of course, not to say that applying IP will eliminate the need for further exploitations; indeed, since negated compounds will be exploited only in *reductio* arguments, we will often be in a position to exploit such resources only after we have used IP. The rule we will use to do that is the one we will consider next.

3.3.2. Using lemmas to complete *reductio*s

Now that we have IP, we are in a position to provide a proof for any argument whose validity depends only on the properties of ⊤, ⊥, conjunction, and negation. To do so we would often need to use LFR—or, in simpler cases, Adj—to make any use of negative resources. This poses no problems when we construct derivations for valid arguments, but LFR does not itself exploit resources so negated compounds would remain as active resources until a gap is closed. In order to count an open gap as having reached a dead end, we would need some description of the conditions under which LFR had been used often enough. Such a description could certainly be given; and, in the last two chapters, we will need to take an analogous approach in the case of one of the rules for quantifiers. But it is possible to keep track of the use of negative resources by way of a genuine exploitation rule.

The basis for such an approach was the third lesson drawn from the law of negation as a premise: if a *reductio* that has ¬ ϕ as a premise is valid, then the lemma ϕ that we need in order to use ¬ ϕ to complete the *reductio* is a valid conclusion from the premises other than ¬ ϕ. That means not only that is it safe to introduce ϕ as a lemma but also that ¬ ϕ can be dropped from the active resources of the gap in which we establish the lemma. Of course, ¬ ϕ is needed along with the lemma to reach the goal ⊥, but we need not introduce a second gap to reach this goal (as would be done with LFR), for such a gap would close immediately by Nc. And, indeed, the law for negation as a premise tells us not only that we can reach the needed lemma (provided that the *reductio* is valid to begin with) but also that this is all that we need to do, for it says that the *reductio* argument Γ, ¬ ϕ / ⊥ is valid only if the argument Γ / ϕ is, too—but also that Γ, ¬ ϕ / ⊥ is valid if Γ / ϕ is.

We will call the rule that implements these ideas ***Completing a Reductio*** (CR).

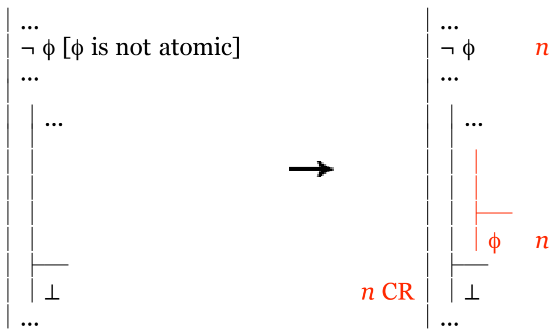
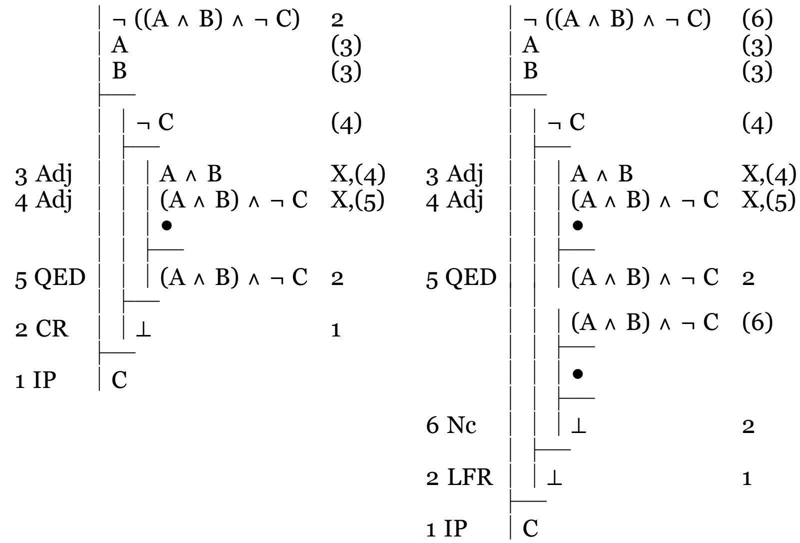


Fig. 3.3.2-1. Developing a derivation by exploiting a negated compound at stage *n*.

The motivation for this rule lies in its use with the negations of non-atomic sentences; and, in fact, we must limit is use to such sentences. It is sound and safe in the case of atomic sentence, but it would not be progressive in that case (given the way we are measuring distance from a dead end) because it would replace a resource that we never exploit by a goal that we could go on to plan for by IP. The rules IP and CR carry us in different directions between gaps whose proximate arguments have the forms Γ, ¬ ϕ / ⊥ and Γ / ϕ, so, if there is any overlap in the sentences ϕ to which they apply, a derivation could move back and forth between the two forever. We block such circles by limiting IP to cases where ϕ is atomic and CR to cases where ϕ is non-atomic.

The following derivations show, on the left, the use of CR in a derivation for the argument from 3.2.2 that was used as an illustration in the last subsection and, on the right, an analogous use of LFR:



The most obvious difference between the two is the extra argument in the second in which the lemma is (A ∧ B) ∧ ¬ C is explicitly used. But the more important difference is that while the first premise is exploited at stage 2 in the left-hand derivation, it remains unexploited in the second. It is true that the absence of the second gap introduced with LFR makes the first derivation shorter, but these derivations are not designed to be the most efficient way of reaching a conclusion and the left-hand derivation is longer than the one at the end of 3.3.1. This extra length is the result of introducing (A ∧ B) ∧ ¬ C as an explicit goal. That provides us with a stage at which we can mark its negation as exploited, but it also makes explicit the resource aimed at by the two uses of adjunction.

3.3.3. More examples

Here is an English argument whose derivation exhibits all of the rules for negation:

Ann's proposal wasn't unfunded without Bill's and Carol's each being funded
Bill's proposal was not funded
―――――――――――――――――――――――――――――――
Ann's proposal was funded

And here is the derivation:

```
         | ¬ (¬ A ∧ ¬ (B ∧ C))     2
         | ¬ B                     (7)
         |----
         | | ¬ A                   (4)
         | |----
         | | | | •
         | | | |----
4 QED    | | | | ¬ A               3
         | | | | | B ∧ C           6
         | | | | |----
6 Ext    | | | | | B
6 Ext    | | | | | C
         | | | | | •
         | | | | |----
7 Nc     | | | | | ⊥             5
         | | | |----
5 RAA    | | | | ¬ (B ∧ C)        3
         | | |----
3 Cnj    | | | ¬ A ∧ ¬ (B ∧ C)   2
         | |----
2 CR     | | ⊥                     1
         |----
1 IP     | A
```

One alternative approach would be to introduce ¬ (B ∧ C) as a lemma at the second stage using LFR.

In the absence of the rules of this section, the exercise **2d** of 3.2.x required use of LFR. Here are two derivations for it that differ in the choice of the premise to be exploited by CR.

```
         | ¬ (A ∧ B)               3
         | ¬ (C ∧ ¬ B)             (8)
         |----
         | | A ∧ C                 2
         | |----
2 Ext    | | A                     (5)
2 Ext    | | C                     (7)
         | | | | •
         | | | |----
5 QED    | | | | A                 4
         | | | | | ¬ B           (7)
         | | | | |----
7 Adj    | | | | | C ∧ ¬ B       X,(8)
         | | | | | •
         | | | | |----
8 Nc     | | | | | ⊥             6
         | | | |----
6 IP     | | | | B                 4
         | | |----
4 Cnj    | | | A ∧ B               3
         | |----
3 CR     | | ⊥                     1
         |----
1 RAA    | ¬ (A ∧ C)
```

```
         | ¬ (A ∧ B)               (8)
         | ¬ (C ∧ ¬ B)             3
         |----
         | | A ∧ C                 2
         | |----
2 Ext    | | A                     (7)
2 Ext    | | C                     (5)
         | | | | •
         | | | |----
5 QED    | | | | C                 4
         | | | | | B             (7)
         | | | | |----
7 Adj    | | | | | A ∧ B         X,(8)
         | | | | | •
         | | | | |----
8 Nc     | | | | | ⊥             6
         | | | |----
6 RAA    | | | | ¬ B               4
         | | |----
4 Cnj    | | | C ∧ ¬ B             3
         | |----
3 CR     | | ⊥                     1
         |----
1 RAA    | ¬ (A ∧ C)
```

These derivations have the same number of stages as the answer in 3.2.xa for **2d**, but their scope lines are nested one deeper. Each of the arguments completing the gaps set up by LFR in the earlier derivation appears in one of these; but we arrive at these arguments in a different way.

It is possible to dispense with Adj, too, and exploit both premises by CR. This leads to a derivation with two more stages and scope lines that are nested more deeply. What we get in return for that increased complexity is direction in how to complete the derivation. In effect, all the thinking required to identify appropriate lemmas is done on paper. The next subsection shows stage by stage how this derivation proceeds.

3.3.4. Approaching derivations

The general rule for starting and continuing derivations is to do anything the rules permit you to do. That is, any rule that can be applied to a goal or active resource of any gap is a legitimate way of proceeding. Some choices may lead to longer derivations than others; but the safety of the rules insures that you can never go off in the wrong direction, and their progressiveness insures that you will always move some distance toward the end. The differences in length that result from different choices of rules are likely to be greatest in valid arguments. When arguments are not valid, you may well have to apply all possible rules. The differences between derivations will then result only from the order in which the rules are applied (though such differences can make for significant differences in length).

The basic rules we have accumulated are shown in the following tables. The one on the left shows the exploitation rules for resources and the planning rules for goals. The simplest way of approaching derivations is to apply these rules as often as possible using the rules from the right-hand table to close gaps whenvever possible.

Rules for developing gaps

	for resources	*for goals*
conjunction $\phi \wedge \psi$	Ext	Cnj
negation $\neg \phi$	CR (if ϕ is not atomic and the goal is $\bot$)	RAA
atomic sentence		IP

Rules for closing gaps

when to close	*rule*
the goal is also a resource	QED
sentences ϕ and $\neg\phi$ are resources & the goal is $\bot$	Nc
$\top$ is the goal	ENV
$\bot$ is a resource	EFQ

The rules LFR and Adj are not shown. They can be used to simplify derivations in some cases but they are never needed; and, when a gap will not close, they may simply delay the inevitable dead end. For this reason, the rules in the tables are labeled ***basic rules*** and are counted as part of the ***basic system of derivations***.

As an example of the use of the basic system, let us look at the promised further derivation for the argument of 3.2.x **2d**. The possible ways of proceeding at each stage are described in the commentary at the left.

Stage 1. We have two premises and a goal and we look to any of them for our starting point. But our premises are negations and can be exploited only in a *reductio* argument—that is, only when the goal is $\bot$. So we must begin by planning for the goal, and RAA is the rule for doing that.

| $\neg (A \wedge B)$
| $\neg (C \wedge \neg B)$
|———
|
|———
| $\neg (A \wedge C)$

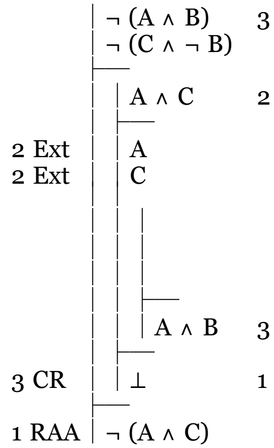
Stage 2. The goal is now $\bot$ and there is no rule to plan for such a goal; but we have three resources, and we are now in a position to exploit any one of them. The rule Ext for exploiting conjunctions is easy, and it sometimes leads to a shorter derivation to do that as soon as possible, so that is what we will do. But there would be nothing wrong with exploiting either of the premises with CR; we will eventually need to do that in any case.

| $\neg (A \wedge B)$
| $\neg (C \wedge \neg B)$
|———
| | $A \wedge C$
| |———
| |
| |———
| | $\bot$ 1
|———
1 RAA | $\neg (A \wedge C)$

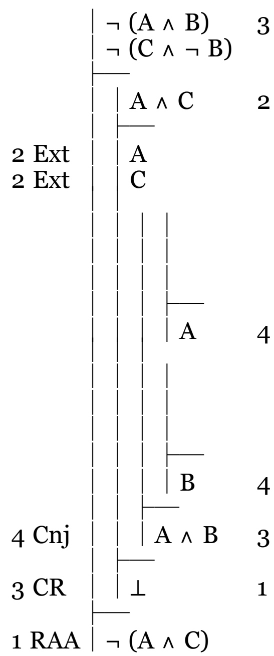
Stage 3. We now have four active resources, the two premises and the two sentences we extracted at stage 2, and our goal is still $\bot$. The two added resources are atomic sentences and can never be exploited, so we must now exploit one of the premises by CR. Either one will do, but we will choose the first.

| $\neg (A \wedge B)$
| $\neg (C \wedge \neg B)$
|———
| | $A \wedge C$ 2
| |———
2 Ext | | A
2 Ext | | C
| |
| |———
| | $\bot$ 1
|———
1 RAA | $\neg (A \wedge C)$

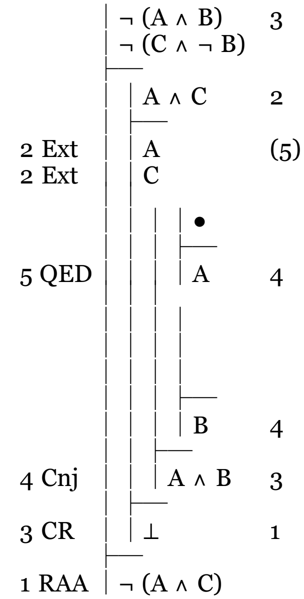
Stage 4. Our goal is now the conjunction A ∧ B so we could plan to get that by Cnj. And that's all we can do because we cannot exploit the second premise until our goal is again ⊥.



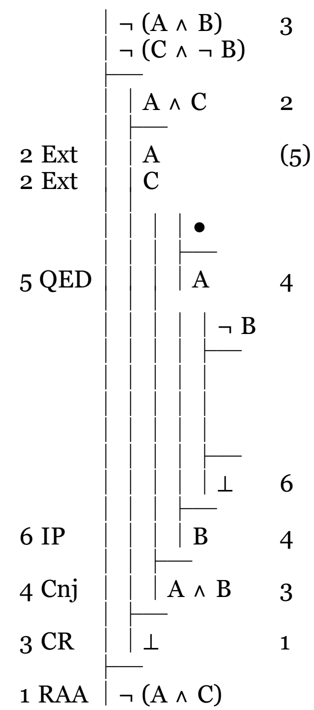
Stage 5. We now have two open gaps, and we could go on to work on either of them. The goal of the first is also one of its resources, so we can close it immediately by QED and that's what we will do. But it would be fine to leave it open while we developed the second gap. It would even be possible to develop the first gap by planning for its goal with IP. While, of course, that would make for a longer derivation, we would eventually run out of things to do and would be forced to notice that the gap could be closed. (It would close on different grounds but, because the rules are safe and sufficient, there would be some reason for closing it.)



Stage 6. Since the first gap is now closed, we can only work on the second. And, since the goal of this gap is not ⊥, we cannot exploit the second premise. Moreover, our other two resources are atomic sentences. So we must plan for the atomic goal, and the rule for doing that is IP.



Stage 7. Our goal is now ⊥ again, so we are forced to turn to our resources for guidance. We have added one, ¬ B; but it is the negation of an atomic sentence so, like A and C, it will never be exploited. But, since we are again working on a *reductio* argument, we can now exploit the second premise by CR.



Stage 8. Our active resources are all atomic sentences or negated atomic sentences so they can never be exploited, but our goal is a conjunction so we can plan to derive it by Cnj.

```
        │ ¬ (A ∧ B)                 3
        │ ¬ (C ∧ ¬ B)               7
        ├───
        │ │ A ∧ C                   2
        │ ├───
2 Ext   │ │ A                       (5)
2 Ext   │ │ C
        │ │ │ │ •
        │ │ │ ├───
5 QED   │ │ │ │ A                   4
        │ │ │ │ │ ¬ B
        │ │ │ │ ├───
        │ │ │ │ │ │
        │ │ │ │ │ ├───
        │ │ │ │ │ │ C ∧ ¬ B       7
        │ │ │ │ ├───
7 CR    │ │ │ │ │ ⊥                 6
        │ │ │ ├───
6 IP    │ │ │ │ B                   4
        │ │ ├───
4 Cnj   │ │ │ A ∧ B                 3
        │ ├───
3 CR    │ │ ⊥                       1
        ├───
1 RAA   │ ¬ (A ∧ C)
```

Stages 9-10. Each of the two open gaps we have now can be closed by QED, and we will go on to do that at the next two stages. Each gap also has a goal that we might plan for; and, as noted earlier, there would be nothing wrong with doing that. And doing it in this case would not lead to a much longer derivation since, once we planned for the goals of these gaps, there would be nothing more we could do with either one except close it.

```
        │ ¬ (A ∧ B)                   3
        │ ¬ (C ∧ ¬ B)                 7
        ├───
        │ │ A ∧ C                     2
        │ ├───
2 Ext   │ │ A                         (5)
2 Ext   │ │ C
        │ │ │ │ •
        │ │ │ ├───
5 QED   │ │ │ │ A                     4
        │ │ │ │ │ ¬ B
        │ │ │ │ ├───
        │ │ │ │ │ │ │
        │ │ │ │ │ │ ├───
        │ │ │ │ │ │ │ C               8
        │ │ │ │ │ │ │
        │ │ │ │ │ │ ├───
        │ │ │ │ │ │ │ ¬ B             8
        │ │ │ │ │ ├───
8 Cnj   │ │ │ │ │ │ C ∧ ¬ B           7
        │ │ │ │ ├───
7 CR    │ │ │ │ │ ⊥                   6
        │ │ │ ├───
6 IP    │ │ │ │ B                     4
        │ │ ├───
4 Cnj   │ │ │ A ∧ B                   3
        │ ├───
3 CR    │ │ ⊥                         1
        ├───
1 RAA   │ ¬ (A ∧ C)
```

The complete derivation is shown below. You can replay its development by moving the cursor across the series of numbers above it.

```
0  1  2  3  4  5  6  7  8  9  10

        | ¬ (A ∧ B)                 3
        | ¬ (C ∧ ¬ B)               7
        |----
        | | A ∧ C                   2
        | |----
2 Ext   | | A                       (5)
2 Ext   | | C                       (9)
        | | | | •
        | | | |----
5 QED   | | | | A                   4
        | | | | | ¬ B               (10)
        | | | | |----
        | | | | | | | •
        | | | | | | |----
9 QED   | | | | | | | C             8
        | | | | | | | •
        | | | | | | |----
10 QED  | | | | | | | ¬ B           8
        | | | | | |----
8 Cnj   | | | | | | C ∧ ¬ B         7
        | | | | |----
7 CR    | | | | | ⊥                 6
        | | | |----
6 IP    | | | | B                   4
        | | |----
4 Cnj   | | | A ∧ B                 3
        | |----
3 CR    | | ⊥                       1
        |----
1 RAA   | ¬ (A ∧ C)
```


3.3.s. Summary

The law for negation as a premise tells us two things about entailment. The first is that any conclusion is valid if and only if the denial of that conclusion can be reduced to absurdity given the premises. This is the principle of indirect proof; it is closely tied to the entailment $\neg \neg \phi \Rightarrow \phi$ (and is subject to the same concerns as is that entailment). We have no need for this principle except in the case of unanalyzed components, which we will begin to call atomic sentences. And, for reasons noted later, we need to limit the use of the rule Indirect Proof (IP) to them.

Another lesson we can draw from the law for negation as a premise is that a *reductio* argument with a negative premise $\neg \phi$ is valid if and only if the sentence ϕ is entailed by the other premises. This tells us that ϕ can be safely introduced as a lemma even if we drop $\neg \phi$ from our active resources. The rule implementing this idea, Completing a *Reductio* (CR) serves as our rule for exploiting negative resources. It applies only to *reductio* arguments but the availability of IP insures that any gap will eventually turn into a gap in a *reductio* argument (unless it closes before that point). Since CR, by dropping a resource $\neg \phi$ and adding a goal ϕ has an effect opposite to that of IP, we must apply them to different sentences ϕ to avoid going in circles. So, just as IP is limited to atomic sentences, CR is limited to negations of non-atomic sentences.

The rule CR can lead us to set as goals any lemmas we need to use negations in completing *reductio* arguments. It therefore eliminates any need for LFR.

The rule Adj is also no longer needed since the rules CR and Cnj will lead us to identify and prove any lemma that Adj would introduce. Indeed, derivations for arguments involving conjunction can now be constructed by letting the rules guide us completely. Any step that is allowed by the basic rules (that is, for now, all rules except LFR and Adj) is safe and will take the derivation some way towards completion. We call the system of derivations limited to those rules the basic system. There will often be different orders in which the basic rules can be applied, and such differences may lead to longer or shorter derivations. The use of non-basic rules can sometimes shorten derivations still further, but they may not bring a derivation any closer to is final state.

The following table collects all rules we have now seen (and, as with the table of 2.4.s, the rule labels are links to the original statements of the rules):

Rules for developing gaps		
	for resources	*for goals*
atomic sentence		IP
negation $\neg \phi$	CR (if ϕ is not atomic and the goal is $\bot$)	RAA
conjunction $\phi \wedge \psi$	Ext	Cnj

Rules for closing gaps		
when to close	*rule*	
the goal is also a resource	QED	
sentences ϕ and $\neg \phi$ are resources & the goal is $\bot$	Nc	
$\top$ is the goal	ENV	
$\bot$ is a resource	EFQ	*Basic system*

Attachment rule		
added resource	*rule*	
$\phi \wedge \psi$	Adj	*Added rules (optional)*
Rule for lemmas		
prerequisite	*rule*	
the goal is $\bot$	LFR	

3.3.x. Exercise questions

Use derivations to establish each of the claims of entailment shown below. You can maximize your practice in the use of CR by avoiding LFR and using Adj only when the goal is a conjunction.

1. $\neg (A \wedge \neg B), A \Rightarrow B$
2. $J \wedge \neg (J \wedge \neg C) \Rightarrow J \wedge C$ (see exercise **1j** of 3.1.x)
3. $\neg (\neg (A \wedge B) \wedge C), \neg A \Rightarrow \neg C$
4. $\neg (A \wedge \neg (B \wedge C)) \Rightarrow \neg (A \wedge \neg B)$
5. $\neg (A \wedge \neg B), \neg (B \wedge \neg C) \Rightarrow \neg (A \wedge \neg C)$
6. $\neg (A \wedge \neg B), \neg (A \wedge \neg C) \Rightarrow \neg (A \wedge \neg (B \wedge C))$

3.3.xa. Exercise answers

1.

```
            │ ¬ (A ∧ ¬ B)        2
            │ A                  (3)
            ├──
            │ │ ¬ B              (3)
            │ ├──
3 Adj       │ │ │ A ∧ ¬ B        X,(4)
            │ │ │ •
            │ │ ├──
4 QED       │ │ │ A ∧ ¬ B        2
            │ ├──
2 CR        │ │ ⊥                1
            ├──
1 IP        │ B
```

2.

```
            │ J ∧ ¬ (J ∧ ¬ C)    1
            ├──
1 Ext       │ J                  (3),(6)
1 Ext       │ ¬ (J ∧ ¬ C)        5
            │ │ •
            │ ├──
3 QED       │ │ J                2
            │ │ │ ¬ C            (6)
            │ │ ├──
6 Adj       │ │ │ │ J ∧ ¬ C      X,(7)
            │ │ │ │ •
            │ │ │ ├──
7 QED       │ │ │ │ J ∧ ¬ C      5
            │ │ ├──
5 CR        │ │ │ ⊥            4
            │ ├──
4 IP        │ │ C                2
            ├──
2 Cnj       │ J ∧ C
```

3.

```
            │ ¬ (¬ (A ∧ B) ∧ C)          2
            │ ¬ A                        (7)
            ├──
            │ │ C                        (4)
            │ ├──
            │ │ │ │ │ A ∧ B              6
            │ │ │ │ ├──
6 Ext       │ │ │ │ │ A                  (7)
6 Ext       │ │ │ │ │ B
            │ │ │ │ │ •
            │ │ │ │ ├──
7 Nc        │ │ │ │ │ ⊥                  5
            │ │ │ ├──
5 RAA       │ │ │ │ ¬ (A ∧ B)            3
            │ │ │ │ •
            │ │ │ ├──
4 QED       │ │ │ │ C                    3
            │ │ ├──
3 Cnj       │ │ │ ¬ (A ∧ B) ∧ C          2
            │ ├──
2 CR        │ │ ⊥                        1
            ├──
1 RAA       │ ¬ C
```

4.

```
            │ ¬ (A ∧ ¬ (B ∧ C))          3
            ├──
            │ │ A ∧ ¬ B                  2
            │ ├──
2 Ext       │ │ A                        (5)
2 Ext       │ │ ¬ B                      (8)
            │ │ │ │ •
            │ │ │ ├──
5 QED       │ │ │ │ A                    4
            │ │ │ │ │ B ∧ C              7
            │ │ │ │ ├──
7 Ext       │ │ │ │ │ B                  (8)
7 Ext       │ │ │ │ │ C
            │ │ │ │ │ •
            │ │ │ │ ├──
8 Nc        │ │ │ │ │ ⊥                  6
            │ │ │ ├──
6 RAA       │ │ │ │ ¬ (B ∧ C)            4
            │ │ ├──
4 Cnj       │ │ │ A ∧ ¬ (B ∧ C)          3
            │ ├──
3 CR        │ │ ⊥                        1
            ├──
1 RAA       │ ¬ (A ∧ ¬ B)
```

5.

```
         │ ¬ (A ∧ ¬ B)                3
         │ ¬ (B ∧ ¬ C)                7
         ├──
         │ │ A ∧ ¬ C                  2
         │ ├──
2 Ext    │ │ A                        (5)
2 Ext    │ │ ¬ C                      (8)
         │ │ │ │ •
         │ │ │ ├──
5 QED    │ │ │ │ A                    4
         │ │ │ │ │ B                  (8)
         │ │ │ │ ├──
8 Adj    │ │ │ │ │ │ B ∧ ¬ C        X,(9)
         │ │ │ │ │ │ •
         │ │ │ │ │ ├──
9 QED    │ │ │ │ │ │ B ∧ ¬ C        7
         │ │ │ │ │ ├──
7 CR     │ │ │ │ │ ⊥                  6
         │ │ │ │ ├──
6 RAA    │ │ │ │ ¬ B                  4
         │ │ │ ├──
4 Cnj    │ │ │ A ∧ ¬ B                3
         │ │ ├──
3 CR     │ │ ⊥                        1
         ├──
1 RAA    │ ¬ (A ∧ ¬ C)
```

6.

```
         │ ¬ (A ∧ ¬ B)                      3
         │ ¬ (A ∧ ¬ C)                      7
         ├──
         │ │ A ∧ ¬ (B ∧ C)                  2
         │ ├──
2 Ext    │ │ A                              (5),(9)
2 Ext    │ │ ¬ (B ∧ C)                      10
         │ │ │ │ •
         │ │ │ ├──
5 QED    │ │ │ │ A                          4
         │ │ │ │ │ B                        (11)
         │ │ │ │ ├──
         │ │ │ │ │ │ │ •
         │ │ │ │ │ │ ├──
9 QED    │ │ │ │ │ │ │ A                    8
         │ │ │ │ │ │ │ │ C                  (11)
         │ │ │ │ │ │ │ ├──
11 Adj   │ │ │ │ │ │ │ │ │ B ∧ C            X,(12)
         │ │ │ │ │ │ │ │ │ •
         │ │ │ │ │ │ │ │ ├──
12 QED   │ │ │ │ │ │ │ │ │ B ∧ C            10
         │ │ │ │ │ │ │ ├──
10 CR    │ │ │ │ │ │ │ │ ⊥                  9
         │ │ │ │ │ │ ├──
9 RAA    │ │ │ │ │ │ │ ¬ C                  8
         │ │ │ │ │ ├──
8 Cnj    │ │ │ │ │ │ A ∧ ¬ C                7
         │ │ │ │ ├──
7 CR     │ │ │ │ │ ⊥                        6
         │ │ │ ├──
6 RAA    │ │ │ │ ¬ B                        4
         │ │ ├──
4 Cnj    │ │ │ A ∧ ¬ B                      3
         │ ├──
3 CR     │ │ ⊥                              1
         ├──
1 RAA    │ ¬ (A ∧ ¬ (B ∧ C))
```

Choosing ¬ (B ∧ C) as the resource to exploit by CR at stage 3 would lead to a somewhat shorter and simpler derivation.

Glen Helman 18 Sep 2004