Soundness and Completeness for Sentence Logic Derivations

13

13–1. SOUNDNESS FOR DERIVATIONS: INFORMAL INTRODUCTION

Let's review what soundness comes to. Suppose I hand you a correct derivation. You want to be assured that the corresponding argument is valid. In other words, you want to be sure that an interpretation which makes all the premises true also makes the final conclusion true. Soundness guarantees that this will always be so. With symbols, what we want to prove is

> T5 (Soundness for sentence logic derivations): For any set of sentences, **Z**, and any sentence, **X**, if **Z**$\vdash$**X**, then **Z**$\models$**X**.

with '$\vdash$' meaning derivability in the system of sentence logic derivations.

The recipe is simple, and you have already mastered the ingredients: We take the fact that the rules for derivations are truth preserving. That is, if a rule is applied to a sentence or sentences (input sentences) which are true in **I**, then the sentence or sentences which the rule licenses you to draw (output sentences) are likewise true in **I**. We can get soundness for derivations by applying mathematical induction to this truth preserving character of the rules.

Consider an arbitrary derivation and any interpretation, **I**, which makes all of the derivation's premises true. We get the derivation's first conclusion by applying a truth preserving rule to premises true in **I**. So this first conclusion will be true in **I**. Now we have all the premises and the first conclusion true in **I**. Next we apply a truth preserving rule to sentences taken from the premises and/or this first conclusion, all true in **I**. So the second conclusion will also be true in **I**. This continues, showing each conclusion down the derivation to be true in **I**, including the last.

Mathematical induction makes this pattern of argument precise, telling us that if all the initial premises are true in **I** (as we assume because we are interested only in such **I**), then all the conclusions of the derivation will likewise be true in **I**.

This sketch correctly gives you the idea of the soundness proof, but it does not yet deal with the complication arising from rules which appeal to subderivations. Let's call a rule the inputs to which are all sentences a *Sentence Rule* and a rule the inputs to which include a subderivation a *Subderivation Rule*. My foregoing sketch would be almost all we need to say if all rules were sentence rules. However, we still need to consider how subderivation rules figure in the argument.

What does it mean to say that the subderivation rule, ⊃I, is truth preserving? Suppose we are working in the outermost derivation, and have, as part of this derivation, a subderivation which starts with assumption **X** and concludes with **Y**. To say that ⊃I is truth preserving is to say that if all the premises of the outer derivation are true in **I**, then **X⊃Y** is also true in **I**. Let's show that ⊃I is truth preserving in this sense.

We have two cases to consider. First, suppose that **X** is false in **I**. Then **X⊃Y** is true in **I** simply because the antecedent of **X⊃Y** is false in **I**. Second, suppose that **X** is true in **I**. But now we can argue as we did generally for outer derivations. We have an interpretation **I** in which **X** is true. All prior conclusions of the outer derivation have already been shown to be true in **I**, so that any sentence reiterated into the subderivation will also be true in **I**. So by repeatedly applying the truth preserving character of the rules, we see that **Y**, the final conclusion of the subderivation, must be true in **I** also. Altogether, we have shown that, in this case, **Y** as well as **X** are true in **I**. But then **X⊃Y** is true in **I**, which is what we want to show.

This is roughly the way things go, but I hope you haven't bought this little argument without some suspicion. It appeals to the truth preserving character of the rules as applied in the subderivation. But these rules include ⊃I, the truth preserving character of which we were in the middle of proving! So isn't the argument circular?

The problem is that the subderivation might have a sub-subderivation to which ⊃I will be applied within the subderivation. We can't run this argument for the subderivation until we have run it for the sub-subderivation. This suggests how we might deal with our problem. We hope we can descend to the deepest level of subderivation, run the argument without appealing to ⊃I, and then work our way back out.

Things are sufficiently entangled to make it hard to see for sure if this strategy is going to work. Here is where mathematical induction becomes indispensable. In chapter 11 all my applications of induction were trivial. You may have been wondering why we bother to raise induction to the status of a **principle** and make such a fuss about it. You will see in the next section that, applied with a little ingenuity, induction will work to straighten out this otherwise very obscure part of the soundness argument.

EXERCISES

13–1. Using my discussion of the ⊃I rule as a model, explain what is meant by the rule ~I being truth preserving and argue informally that ~I is truth preserving in the sense you explain.

13–2. Explain why, in proving soundness, we only have to deal with the primitive rules. That is, show that if we have demonstrated that all derivations which use only primitive rules are sound, then any derivation which uses any derived rules will be sound also.

13–2. SOUNDNESS FOR DERIVATIONS: FORMAL DETAILS

The straightforward but messy procedure in our present case is to do a double induction. One defines the complexity of a derivation as the number of levels of subderivations which occur. The inductive property is that all derivations of complexity n are sound. One then assumes the inductive hypothesis, that all derivations with complexity less than n are sound, and proves that all derivations of complexity n are sound. In this last step one does another induction on the number of lines of the derivation. This carries out the informal thinking developed in the last section. It works, but it's a mess. A different approach takes a little work to set up but then proceeds very easily. Moreover, this second approach is particularly easy to extend to predicate logic.

This approach turns on a somewhat different way of characterizing the truth preserving character of the rules, which I call *Rule Soundness*, and which I asked you to explore in exercises 10–4, 10–5, and 10–6. One might argue about the extent to which this characterization corresponds intuitively to the idea of the rules being truth preserving. I will discuss this a little, but ultimately it doesn't matter. It is easy to show that the rules are truth preserving in the sense in question. And using the truth preserving character thus expressed, proof of soundness is almost trivial.

Here is the relevant sense of rule soundness, illustrated for the case of &I. Suppose we are working within a derivation with premises **Z**. Suppose we have already derived **X** and **Y**. Then we have $Z \vdash X$ and $Z \vdash Y$. &I then licenses us to conclude **X&Y**. In other words, we can state the &I rule by saying

&I Rule: If $Z \vdash X$ and $Z \vdash Y$, then $Z \vdash X\&Y$.

There is a fine point here, about whether this really expresses the &I rule. The worry is that '$Z \vdash X$' means **there exists** a derivation from **Z** to **X**, and '$Z \vdash Y$' means that **there exists** a derivation from **Z** to **Y**. But the two derivations may well not be the same, and they could both differ extensively from some of the derivations in virtue of which '$Z \vdash X\&Y$' is true.

For sentence rules, this worry can be resolved. But it's really not important because, as with rule soundness, this way of stating the rules will provide us with all we need for the soundness proof. We proceed by introducing the sense in which the &I rule is sound. We do this by taking the statement of the rule and substituting '$\vDash$' for '$\vdash$':

L7 (Soundness of &I): If $Z \vDash X$ and $Z \vDash Y$, then $Z \vDash X\&Y$.

Why should we call this soundness of the &I rule? First, it has the same form as the rule &I. It is the semantic statement which exactly parallels the syntactic statement of the &I rule. And it tells us that if we start with any interpretation **I** which makes the premises **Z** true, and if we get as far as showing that **X** and **Y** are also true in **I**, then the conjunction **X&Y** is likewise true in **I**.

In particular, you can show that L7 directly implies that &I is truth preserving in the original sense by looking at the special case in which $Z = \{X,Y\}$. $\{X,Y\} \vDash X$ and $\{X,Y\} \vDash Y$ are trivially true. So L7 says that $\{X,Y\} \vDash X\&Y$, which just says that any interpretation which makes **X** true and also makes **Y** true makes the conjunction **X&Y** true.

We treat the other sentence rules in exactly the same way. This gives

L8 (Soundness of &E: If $Z \vDash X\&Y$, then $Z \vDash X$; and if $Z \vDash X\&Y$, then $Z \vDash Y$.

L9 (Soundness of vI): If $Z \vDash X$, then $Z \vDash X \vee Y$; and if $Z \vDash Y$, then $Z \vDash X \vee Y$.

L10 (Soundness of vE): If $Z \vDash X \vee Y$ and $Z \vDash \sim X$, then $Z \vDash Y$; and if $Z \vDash X \vee Y$ and $Z \vDash -Y$, then $Z \vDash X$.

L11 (Soundness of ~E): If $Z \vDash \sim\sim X$, then $Z \vDash X$.

L12 (Soundness of ⊃E: If $Z \vDash X \supset Y$ and $Z \vDash X$, then $Z \vDash Y$.

L13 (Soundness of ≡I): If $Z \vDash X \supset Y$ and $Z \vDash Y \supset X$, then $Z \vDash X \equiv Y$.

L14 (Soundness of ≡E): If $Z \vDash X \equiv Y$, then $Z \vDash X \supset Y$; and if $Z \vDash X \equiv Y$, then $Z \vDash Y \supset X$.

EXERCISES

13–3. Prove lemmas L7 to L14. Note that in proving these you do not need to deal with $\vdash$ at all. For example, to prove L7, you need to show, using the antecedent, that $Z \vDash X\&Y$. So you assume you are given an **I** for which all sentences in **Z** are true. You then use the antecedent of L7 to show that, for this **I**, **X&Y** is also true.

13–4. In this problem you will prove that for sentence rules, such as the rules described in L7 to L14, what I have called rule soundness and the statement that a rule is truth preserving really do come to the same thing. You do this by giving a general expression to the correspondence between a syntactic and a semantic statement of a rule:

Suppose that **X, Y,** and **W** have forms such that

(i) $(\forall I)\{[Mod(I,X)\ \&\ Mod(I,Y)] \supset Mod(I,W)\}$.

That is, for all **I**, if **I** makes **X** true and makes **Y** true, then **I** makes **W** true. Of course, this won't be the case for just any **X**, **Y**, and **W**. But in special cases, **X**, **Y**, and **W** have special forms which make (i) true. For example, this is so if $X = U$, $Y = U \supset V$, and $W = V$. In such cases, thinking of **X** and **Y** as input sentences of a rule and **W** as the output sentence, (i) just says that the rule that allows you to derive **W** from **X** and **Y** is truth preserving in our original sense.

Now consider

(ii) If $Z \vDash X$ and $Z \vDash Y$, then $Z \vDash W$.

This is what I have been calling soundness of the rule stated by saying that if $Z \vdash X$ and $Z \vdash Y$, then $Z \vdash W$. (ii) gives turnstyle expression to the statement that the rule which licenses concluding **W** from **X** and **Y** is truth preserving.

Here is your task. Show that, for all **X**, **Y**, and **W**, (i) holds iff and (ii) holds. This shows that for sentence rules (rules which have only sentences as inputs) the two ways of saying that a rule is truth preserving are equivalent. Although for generality, I have expressed (i) and (ii) with two input sentences, your proof will work for rules with one input sentence. You can show this trivially by letting $Y = A \vee \sim A$ for rules with one input sentence.

I have not yet discussed the two subderivation rules, ⊃I and ~I. Soundness of these rules comes to

L15 (Soundness of ⊃I): If $Z\cup\{X\}\models Y$, then $Z\models X\supset Y$.

L16 (Soundness of ~I): If $Z\cup\{X\}\models Y$ and $Z\cup\{X\}\models\sim Y$, then $Z\models\sim X$.

In the case of ⊃I and ~I there is a more substantial question of whether, and in what sense, L15 and L16 also express the intuitive idea that these rules are truth preserving. The problem is that the turnstyle notion makes no direct connection with the idea of subderivations. Thus, if the syntactic counterpart of L15 is assumed (if $Z\cup\{X\}\vdash Y$, then $Z\vdash X\supset Y$), it is not clear whether, or in what sense, one can take this to be a statement of the ⊃I rule. (The converse is clear, as you will show in exercise 13–6.) However, this issue need not sidetrack us, since L15 and L16 will apply directly in the inductive proof, however one resolves this issue.

EXERCISES

13–5. Prove L15 and L16.

13–6. Prove that if the system of derivations includes the rule ⊃I, then if $Z\cup\{X\}\vdash Y$, then $Z\vdash X\supset Y$. Also prove that if the system of derivations includes the rule ~I, then if both $Z\cup\{X\}\vdash Y$ and $Z\cup\{X\}\vdash\sim Y$, then $Z\vdash\sim X$.

We are now ready to prove T5, soundness for derivations. Here is an outline of the proof: We will start with an arbitrary derivation and look at an arbitrary line, n. We will suppose that any interpretation which makes governing premises and assumptions true makes all prior lines true. Rule soundness will then apply to show that the sentence on line n must be true too. Strong induction will finally tell us that all lines are true when their governing premises and assumptions are true. The special case of the derivation's last line will constitute the conclusion we need for soundness.

To help make this sketch precise, we will use the following notation:

X_n is the sentence on line n of a derivation. Z_n is the set of premises and assumptions which govern line n of a derivation.

Now for the details. Suppose that for some Z and X, $Z\vdash X$. We must show that $Z\models X$. The assumption $Z\vdash X$ means that there is some derivation with premises a subset of Z, final conclusion X, and some final line number which we will call n*. The initial premises are the sentences, Z_{n^*}, governing the last line, n*; and the final conclusion, X, is the sentence on the last line, which we are calling X_{n^*}. We will show that $Z_{n^*}\models X_{n^*}$. This will establish $Z\models X$ because $X_{n^*} = X$ and Z_{n^*} is a subset of Z. (Remember exercise 10–10.)

We will establish $Z_{n^*}\models X_{n^*}$ by showing that $Z_n\models X_n$ for all n, $1 \le n \le n^*$. And in turn we will establish this by applying strong induction. We will use the

Inductive property: $Z_i\models X_i$.

and the

Inductive hypothesis: $Z_i\models X_i$ holds for all $i < n$.

So let's consider an arbitrary line, n, and assume the inductive hypothesis. What we have to do is to consider each of the ways in which line n might be justified and, applying the inductive hypothesis, show that the inductive property holds for line n.

First, X_n might be a premise or an assumption. Notice, by the way, that this covers the special case of the first line (n = 1), since the first line of a derivation is either a premise or, in the case of a derivation with no premises, the assumption of a subderivation. But if X_n is a premise or assumption, X_n is a member of Z_n. Therefore, $Z_n\models X_n$.

Next we consider all the sentence rules. I'll do one for you and let you do the rest. Suppose that X_n arises by application of &I to two previous lines, X_i and X_j, so that $X_n = X_i\&X_j$. By the inductive hypothesis

$Z_i\models X_i$ and $Z_j\models X_j$ (Inductive hypothesis)

Since we are dealing with a sentence rule, X_i, X_j, and X_n all occur in the same derivation. Consequently, $Z_i = Z_j = Z_n$. So

$Z_n\models X_i$ and $Z_n\models X_j$.

This is just the antecedent of lemma 7, which thus applies to the last line to give $Z_n\models X_n$.

EXERCISE

13–7. Apply lemmas L8 to L14 to carry out the inductive step for the remaining sentence rules. Your arguments will follow exactly the same pattern just illustrated for &I.

Turning to the other rules, suppose that X_n arises by reiteration from line i. That is just to say that $X_n = X_i$. We have as inductive hypothesis that $Z_i\models X_i$. If lines i and n are in the same derivation, $Z_n = Z_i$, so that $Z_n\models X_n$, as we require. If we have reiterated X_i into a subderivation, Z_n

differs from Z_i by adding the assumption of the subderivation (or the assumptions of several subderivations if we have reiterated several levels down). That is, Z_i is a subset of Z_n. But as you have shown in exercise 10–10, if $Z_i \vDash X_n$ and Z_i is a subset of Z_n, then $Z_n \vDash X_n$.

Now suppose that X_n arises by $\supset$I. Then on previous lines there is a subderivation, beginning with assumption X_i and concluding with X_j, so that $X_n = X_i \supset X_j$. By inductive hypothesis,

$Z_j \vDash X_j$ (Inductive hypothesis for line j)

The trick here is to notice that the subderivation has one more assumption than Z_n. Though not perfectly general, the following diagram will give you the idea:

Set of Premises and Assumptions

Z	Z_n
.	
.	
.	
$\quad X_i$	$Z_i = Z_n \cup \{X_i\}$
$\quad$.	
$\quad$.	
$\quad$.	
$\quad X_j$	$Z_j = Z_n \cup \{X_i\}$
$X_n (= X_i \supset X_j)$	Z_n

When we start the subderivation with the assumption of X_i, we **add** the assumption X_i to Z_n to get $Z_i = Z_n \cup \{X_i\}$ as the total set of premises and assumptions on line i. When we get to line n and discharge the assumption of X_i, moving back out to the outer derivation, we revert to Z_n as the set of governing premises and assumptions.

Since $Z_j = Z_n \cup \{X_i\}$, we can rewrite what the inductive hypothesis tells us about line j as

$Z_n \cup \{X_i\} \vDash X_j$.

But this is just the antecedent of lemma L15! Thus lemma L15 immediately applies to give $Z_n \vDash X_i \supset X_j$, or $Z_n \vDash X_n$, since $X_n = X_i \supset X_j$.

EXERCISE

13–8. Carry out the inductive step for the case in which X_n arises by application of ~I. Your argument will appeal to lemma L16 and proceed analogously to the case for $\supset$I.

We have covered all the ways in which X_n can arise on a derivation. Strong inducton tells us that $Z_n \vDash X_n$ for all n, including n*, the last line of the derivation. Since Z_{n^*} is a subset of Z and $X_{n^*} = X$, this establishes $Z \vDash X$, as was to be shown.

13–3. COMPLETENESS FOR DERIVATIONS: INFORMAL INTRODUCTION

We still need to prove

T6 (Completeness for sentence logic derivations): For any finite set of sentences, **Z**, and any sentence, **X**, if $Z \vDash X$, then $Z \vdash X$.

where '$\vdash$' is understood to mean $\vdash_d$, derivability in our natural deduction system. The proof in this section assumes that **Z** is finite. Chapter 14 will generalize to the case of infinite **Z**.

The proof of completeness for derivations is really an adaptation of the completeness proof for trees. If you have studied the tree completeness proof, you will find this and the next section relatively easy. The connection between trees and derivations on this matter is no accident. Historically, the tree method was invented in the course of developing the sort of completeness proof that I will present to you here.

Begin by reading section 12–1, if you have not already done so, since we will need lemma L1 and the notation from that section. Also, do exercises 12–1 and 12–2. (If you have not studied trees, you will need to refresh your memory on the idea of a counterexample; see section 4–1, volume I.) For quick reference, I restate L1:

L1: $Z \vDash X$ iff $Z \cup \{\sim X\}$ is inconsistent.

The basis of our proof will be to replace completeness with another connection between semantic and syntactic notions. Let us say that

D19: **Z** is *Syntactically Inconsistent* iff $Z \vdash A \& \sim A$.

Semantic inconsistency is just what I have been calling 'inconsistency', defined in chapter 10, D7, as $(\forall I)\sim$**Mod(I,Z)**. L1 says that an argument is valid iff the premises together with the negation of the conclusion form a semantically inconsistent set. Analogously

L17: $Z \cup \{\sim X\} \vdash A \& \sim A$ iff $Z \vdash X$.

says that ~**X** together with the sentences in **Z** form a syntactically inconsistent set iff there is a proof using sentences in **Z** as premises to the conclusion **X**. Together, L1 and L17 show that T6 is equivalent to

> T7: For any finite set of sentences, **Z**, if **Z** is semantically inconsistent, then **Z** is syntactically inconsistent; that is, if (∀I)~**Mod**(**I**,**Z**), then **Z**⊢A&~A.

EXERCISES

13–9. Prove L17.

13–10. Using L1 and L17, prove that T6 is equivalent to T7.

We have boiled our problem down to proving T7. We do this by developing a specialized, mechanical kind of derivation called a *Semantic Tableau Derivation.* Such a derivation provides a systematic way of deriving a contradiction if the original premises form an inconsistent set.

If you haven't done trees, it is going to take you a little time and patience to see how this method works. On a first reading you may find the next few paragraphs very hard to understand. Read them through even if you feel quite lost. The trick is to study the two examples. If you go back and forth several times between the examples and the text you will find that the ideas will gradually come into focus. The next section will add further details and precision.

A semantic tableau derivation is a correct derivation, formed with a special recipe for applying derivation rules. Such a derivation is broken into segments, each called a *Semantic Tableau,* marked off with double horizontal lines. We will say that one tableau *Generates* the next tableau. Generating and generated tableaux bear a special relation. If all of a generated tableau's sentences are true, then all the sentences of previous generating tableaux are true also. In writing a derivation, each tableau we produce has shorter sentences than the earlier tableaux. Thus, as the derivation develops, it provides us with a sequence of tableaux, each a list of sentences such that the sentences in the later tableaux are shorter. The longer sentences in the earlier tableaux are guaranteed to be true if all of the shorter sentences in the later tableaux are true.

A tableau derivation works to show that if a set, **Z**, of sentences is semantically inconsistent, then it is syntactically inconsistent. Such derivations accomplish this aim by starting with the sentences in **Z** as its premises. The derivation is then guaranteed to have 'A&~A' as its final conclusion if **Z** is (semantically) inconsistent.

To see in outline how we get this guarantee, suppose that **Z** is an arbitrary finite set of sentences, which may or may not be inconsistent. (From now on, by 'consistent' and 'inconsistent' I will always mean **semantic** consistency and inconsistency, unless I specifically say 'syntactic consistency' or 'syntactic inconsistency'.) A tableau derivation, starting from **Z** as premises, will continue until it terminates in one of two ways. In the first way, some final tableau will have on it only atomic and/or negated atomic sentences, none of which is the negation of any other. You will see that such a list of sentences will describe an interpretation which will make true all the sentences in that and all previous tableaux. This will include the original premises, **Z**, showing this set of sentences to be consistent. Furthermore, we will prove that if the initial sentences form a consistent set, the procedure **must** end in this manner.

Consequently, if the original set of sentence forms an **inconsistent** set, the tableau procedure cannot end in the first way. It then ends in the second way. In this alternative, all subderivations end with a contradiction, 'A&~A'. As you will see, argument by cases will then apply repeatedly to make 'A&~A' the final conclusion of the outermost derivation.

Altogether we will have shown that if **Z** is (semantically) inconsistent, then **Z**⊢A&~A, that is, **Z** is syntactically inconsistent.

To see how all this works you need to study the next two examples. First, here is a tableau derivation which ends in the first way (in writing lines 3 and 4, I have omitted a step, '~B&~C', which gives 3 and 4 by &E):

```
 1 | ~(BvC)          P
 2 |  BvD            P
   |==========1
 3 | ~B              1, DM, &E
 4 | ~C              1, DM, &E
 5 |  BvD            2, R
   |==========2
 6 |   |  B          A
   |   |----
 7 |   | ~B          3, R
 8 |   | ~C          4, R
 9 |   | A&~A        6, 7, CD
   |       x
   |   ==========3
10 |   | D           A
   |   |----
11 |   | ~B          3, R
12 |   | ~C          4, R
   |
   |   ==========4
```

You can see that this is a correct derivation in all but two respects: I have abbreviated by omitting the step '~B&~C', which comes from 1 by DM and gives 3 and 4 by &E; and I have not discharged the assumptions of the subderivations to draw a final conclusion in the outer derivation.

Each tableau is numbered at the end of the double lines that mark its end. A tableau may generate one new tableau *(Sequential Generation):* In this example tableau 1 generated tableau 2 by applying the rules DM, &E, and R. Or a tableau may generate two new tableaus *(Branching Generation):* In the example tableau 2 generated tableaux 3 and 4 by starting two new subderivations, each using for its assumption one of the disjuncts, 'B' and 'D' of 'BvD' on line 5, and each reiterating the rest of tableau 2.

Tableau 3 ends in a contradiction. It can't describe an interpretation. We mark it with an '×' and say that it is *Closed.* Tableau 4, however is *Open.* It does not contain any sentence and the negation of the same sentence; and all its sentences are *Minimal,* that is, either atomic or negated atomic sentences. Tableau 4 describes an interpretation by assigning f to all sentence letters which appear negated on the tableau and t to all the unnegated sentence letters. In other words, the interpretation is the truth value assignment which makes true all the sentences on this terminal tableau.

Note how the interpretation described by tableau 4 makes true all the sentences on its generator, tableau 2. The truth of '~B' and '~C' carries upward simply because they are reiterated, and the truth of 'D' guarantees the truth of 'BvD' by being a disjunct of the disjunction. You should check for yourself that the truth of the sentences in tableau 2 guarantees the truth of the sentences in tableau 1.

Examine this example of a tableau derivation which ends in the second way:

```
 1 | ~(BvC)        P
 2 |  BvC          P
   |=========1
 3 | ~B            1, DM, &E
 4 | ~C            1, DM, &E
 5 |  BvC          2, R
   |=========2
 6 |    |  B       A
   |    |----
 7 |    | ~B       3, R
 8 |    | ~C       4, R
 9 |    | A&~A     6, 7, CD
   |       ×
   |    =========3
10 |    |  C       A
   |    |----
11 |    | ~B       3, R
12 |    | ~C       4, R
13 |    | A&~A     10, 12, CD
   |       ×
   |    =========4
14 | A&~A          5, 6-9, 10-13, AC
```

In this example, all terminal tableaux (3 and 4) close, that is, they have both a sentence and the negation of the same sentence, to which we apply

the rule CD. We can then apply AC to get the final desired conclusion, 'A&~A'.

Again, here is the key point: I am going to fill in the details of the method to guarantee that a consistent initial set of sentences will produce a derivation like the first example and that an inconsistent set will give a result like the second example. More specifically, we will be able to prove that if there is an open terminal tableau, like tableau 4 in the first example, then that tableau describes an interpretation which makes true all its sentences and all the sentences on all prior tableaux. Thus, if there is an open terminal tableau, there is an interpretation which constitutes a model of all the initial sentences, showing them to form a consistent set. Conversely, if the original set is inconsistent, all terminal tableaux must close. We will than always be able to apply argument by cases, as in the second example, to yield 'A&~A' as a final conclusion. But the last two sentences just state T7, which is what we want to prove.

To help you get the pattern of the argument, here is a grand summary which shows how all our lemmas and theorems connect with each other. We want to show T6, that if $\mathbf{Z}\vDash\mathbf{X}$, then $\mathbf{Z}\vdash\mathbf{X}$. We will assume $\mathbf{Z}\vDash\mathbf{X}$, and to take advantage of lemmas L1 and L17, we then consider a semantic tableau derivation with the sentences in $\mathbf{Z}\cup\{\sim\mathbf{X}\}$ as the initial tableau. Then we argue

(1) $\mathbf{Z}\vDash\mathbf{X}$. (Assumption)
(2) If $\mathbf{Z}\vDash\mathbf{X}$, then $\mathbf{Z}\cup\{\sim\mathbf{X}\}$ is inconsistent. (By L1)
(3) If some terminal tableau is open, then $\mathbf{Z}\cup\{\sim\mathbf{X}\}$ is consistent. (By L18, to be proved in the next section)
(4) If $\mathbf{Z}\cup\{\sim\mathbf{X}\}$ is inconsistent, then all terminal tableaux close. (Contrapositive of (3))
(5) If all terminal tableaux close, then $\mathbf{Z}\cup\{\sim\mathbf{X}\}\vdash A\&\sim A$. (L20, to be proved in the next section)
(6) If $\mathbf{Z}\cup\{\sim\mathbf{X}\}\vdash A\&\sim A$, then $\mathbf{Z}\vdash\mathbf{X}$. (By L17)

Now all we have to do is to discharge the assumption, (1), applying it to (2), (4), (5), and (6), giving

T6: If $\mathbf{Z}\vDash\mathbf{X}$, then $\mathbf{Z}\vdash\mathbf{X}$.

In the next section we carry out this strategy more compactly by proving T7 (corresponding to (4) and (5) above), which you have already proved to be equivalent to T6.

13–4. COMPLETENESS FOR DERIVATIONS: FORMAL DETAILS

To keep attention focused on the main ideas, I'm going to restrict consideration to sentences in which '~' and 'v' are the only connectives used.

Once you understand this special case, extension to the other connectives will be very easy. As I mentioned, I will also carry out the proof only under the restriction that the initial set of sentences, **Z**, is finite. Chapter 14 will generalize the result to infinite sets, **Z**.

To help fix ideas, I'll start with a slightly more extended example. Skip over it now and refer back to it as an illustration as you read the details.

```
 1 | ~Pv(DvM)                 P
 2 | ~[(~PvD)v(~PvM)]         P
   |=================1
 3 | ~(~PvD)                  2, ~v
 4 | ~(~PvM)                  2, ~v
 5 | ~Pv(DvM)                 1, R
   |=================2
 6 | ~~P                      3, ~v
 7 | ~D                       3, ~v
 8 | ~~P                      4, ~v
 9 | ~M                       4, ~v
10 | ~Pv(DvM)                 5, R
   |=================3
11 |   | ~P                   A
   |   |--------
12 |   | ~~P                  6, R
13 |   | ~D                   7, R
14 |   | ~M                   9, R
15 |   | A&~A                 11, 12, CD
   |      ×
   |   ==============4
16 |   | DvM                  A
   |   |--------
17 |   | ~~P                  6, R
18 |   | ~D                   7, R
19 |   | ~M                   9, R
   |   ==============5
20 |   |   | D                  A
   |   |   |------
21 |   |   | ~~P                17, R
22 |   |   | ~D                 18, R
23 |   |   | ~M                 19, R
24 |   |   | A&~A               20, 22, CD
   |   |      ×
   |   |   ==========6
25 |   |   | M                  A
   |   |   |------
26 |   |   | ~~P                17, R
27 |   |   | ~D                 18, R
28 |   |   | ~M                 19, R
29 |   |   | A&~A               25, 28, CD
   |   |      ×
   |   |   ==========7
30 |   | A&~A                   16, 20-24, 25-29, AC
31 | A&~A                       10, 11-15, 16-30, AC
```


The method of semantic tableau derivations constitutes a way of testing a finite initial set of sentences for consistency. Here are the rules for generating such a derivation:

> R1 *Initial Tableau:* The method begins by listing the sentences in the set to be tested as the premises of the derivation. This initial list constitutes the initial tableau.

Lines 1 and 2 in the example are an initial tableau.

Each further tableau (the *Generated Tableau*) is generated from some prior tableau (the *Generating Tableau*) by one of two methods:

> R2 *Sequential generation*
>
> a) Each line of the generated tableau is a new line of the same derivation as the generating tableau.
> b) If a sentence of the form ~~**X** occurs on the generating tableau, enter **X** on the generated tableau.
> c) If a sentence of the form ~(**X**v**Y**) occurs on the generating tableau, enter ~**X** and ~**Y** as separate lines on the generated tableau.
> d) Reiterate all remaining sentences of the generating tableau as new lines of the generated tableau.

Tableaux 2 and 3 in the example illustrate sequentially generated tableaux. c) is illustrated in the example by lines 3, 4, 6, 7, 8, and 9. d) is illustrated by lines 5 and 10. Note that the rule I apply for c), which I have called '~v', is a new derived rule, constituted by simply applying DM followed by &E.

> R3 *Branching generation:*
>
> a) If a sentence of the form **X**v**Y** occurs on the generating tableau, start two new subderivations, one with assumption **X** and the other with assumption **Y**.
> b) Reiterate **all** the remaining sentences of the generating tableau on each of the subderivations.
> c) Each of the (initial parts of) the subderivations started by steps a) and b) constitutes a generated tableau.

Branching generation is illustrated in the example by tableaux 4, 5, 6, 7.

Tableaux 4, 6, and 7 illustrate what happens when both a sentence and the negation of a sentence appear on a tableau. No interpretation will make all the sentences on such a tableau true. So such a tableau will never provide an interpretation which will prove the original sentences consistent. We record this fact by extending the tableau by applying CD to derive 'A&~A'. We say that such a tableau is *Closed* and mark it with an '×'.

We have applied CD to draw the explicit contradiction, 'A&~A', on closed tableaux because this contradiction will be helpful in deriving

'A&~A' in the outermost derivation. We will see that, if the original set of sentences is inconsistent, then all chains of tableaux will terminate with a closed tableau. Argument by cases will then allow us to export 'A&~A' from subderivations to outer derivations, step by step, until we finally get 'A&~A' as the final conclusion of the outermost derivation.

We make these ideas more precise with two further instructions:

> R4: If both a sentence and the negation of the same sentence appear on a tableau, apply CD to derive 'A&~A' as the last line of the tableau, and mark the end of the tableau with an '×' to indicate that it is *Closed.* Do not generate any new tableaux from a closed tableau.
>
> R5: If 'A&~A' appears on two subderivations, both generated by the same disjunction in the outer derivation, apply AC to write 'A&~A' as the final conclusion on the outer derivation.

Look again at tableaux 4, 6, and 7, as illustrations of R4. Lines 30 and 31 illustrate R5.

We now need to prove that semantic tableau derivations do what they are supposed to do. Here is the intuitive idea. We start with a set of sentences. The tableau procedure constitutes a way of determining whether or not this set is consistent. This works by systematically looking for all possible ways of making the original sentences true. If the systematic search turns up a way of making all the original sentences true (a model), then we know that the original set is consistent. Indeed, we will prove that if the original set is consistent, the procedure will turn up such an interpretation. Thus we know that if the procedure **fails** to turn up such an interpretation, the original set **must** be inconsistent. This is signaled by all chains of tableaux terminating with a closed tableau.

The procedure accomplishes these aims by resolving the original sentences into simpler and simpler sentences which enable us to see what must be true for the original set to be true. Each new tableau consists of a set of sentences, at least some of which are shorter than previous sentences. If all of the generated tableau's sentences are true, then all of the sentences on the generating tableau will be true. For a sequentially generated tableau, the new sentences give us what has to be true for the sentences on the generating tableau to be true. When we have branching generation, each of the two new tableaux gives one of the only two possible ways of making all sentences of the generating tableau true. In this way the procedure systematically investigates all ways in which one might try to make the original sentences true. Attempts that don't work end in closed tableaux.

We need to work these ideas out in more detail. We will say that

> A tableau is a *Terminal Tableau* if it has not generated any other tableau, and no rule for tableau generation applies to it.

It can happen that no rule applies to a tableau for one of two reasons:

The tableau can be closed. Or it might be open but have only minimal sentences (atomic or negated atomic sentences). We will discuss these two cases separately.

First we will prove

> L18: An open terminal tableau describes an interpretation in which all sentences of the initial tableau are true.

An open terminal tableau has only minimal sentences, none of which is the negation of any other. The interpretation such a tableau specifies is the one which makes all its sentences true, that is, the assignment of t to all the tableau's unnegated atomic sentences and f to the atomic sentences which appear negated on the tableau. Let's call such an interpretation a *Terminal Interpretation,* for short.

Our strategy will be to do an induction. Suppose we are given an open terminal tableau, and so the terminal interpretation, **I**, which it specifies. The fact that all the sentences of the terminal tableau are true in **I** provides our basis step. For the inductive step you will show that instructions for constructing a tableau derivation guarantee that if all the sentences of a generated tableau are true in an interpretation, then all the sentences of the generating tableau are true in the same interpretation. Thus all the sentences of the tableau which generated the terminal tableau will be true in **I**. In turn, that tableau's generator will have all its sentences true in **I**. And so on up. In short, induction shows that all the *Ancestors* of the open terminal tableau are true.

To fill in the details of this sketch, you will first prove the inductive step:

> L19: If tableau $\mathbf{T}_2$ is generated from tableau $\mathbf{T}_1$ and all sentences of $\mathbf{T}_2$ are true in interpretation **I**, then all the sentences of $\mathbf{T}_1$ are also true in **I**.

EXERCISE

13–11. Prove L19.

Since the proof of L18 will be inductive, we need to specify more clearly the sequence of cases on which to do the induction:

> A terminal tableau's generator will be called the tableau's first *Ancestor.* In general, the i + 1st ancestor of a terminal tableau is the generator of the ith ancestor.

We will do the induction starting from a 0th case, namely, the terminal tableau. The ith case will be the terminal tableau's ith ancestor.

We are now ready to prove L18. Suppose we are given a semantic tableau derivation, with an open terminal tableau. This tableau specifies an interpretation, **I**, in which all the terminal tableau's sentences are true. The inductive property is: The nth ancestor of the terminal tableau has all its sentences true in **I**. The terminal tableau provides the basis case. By L19, if the nth ancestor of the terminal tableau has all its sentences true in **I**, then so does the n + 1st ancestor. Then, by induction, all the terminal tableau's ancestors have all their sentences true in **I**, which includes the derivation's initial tableau, as required to prove L18.

I have now said all I need about tableau derivations which terminate with one or more open tableaux. What happens if all the terminal tableaux are closed? In a word, rule R5 applies repeatedly until, finally, 'A&~A' appears as the final conclusion of the outermost derivation:

> L20: If in a semantic tableau derivation all the terminal tableaux are closed, then 'A&~A' appears as the derivation's final conclusion.

We will prove this with another induction.

We need a sequence of cases on which to do the induction. The natural choice is the level or depth of subderivations, as measured by the number of nested scope lines. But we want to start with the deepest level of subderivation and work our way back out. So we need to reverse the ordering: The first level of subderivations will be the deepest, the second will be the subderivations one level less deep, and so on. More exactly defined

> Given a tableau derivation, let k be the largest number of nested scope lines on the derivation (including the outermost scope line). The *Inverted Level* of each subderivation is k less the number of scope lines to the left of the subderivation.

(I will henceforth omit the word 'inverted' in 'inverted level'.)

The key to the proof will be the inductive step:

> L21: Let D be a semantic tableau derivation in which all terminal tableaus are closed. Then, if all of D's subderivations of level n have 'A&~A' as their final conclusion, so do all the subderivations of level n + 1.

(I construe 'subderivation' broadly to include the outermost derivation, a sort of null case of a subderivation.)

EXERCISE

13–12. Prove L21.

We are now ready to prove L20. Let D be a semantic tableau derivation in which all terminal tableaux are closed. Our inductive property will be: All the subderivations of level n have 'A&~A' as their final conclusion. At level 1 all subderivations have no sub-subderivations. So all of the subderivations must end in terminal tableaux. By assumption, all of these are closed. So the inductive property holds for level 1. L21 gives the inductive step. By induction, the derivations at all levels conclude with 'A&~A', which includes the outermost derivation.

We are at long last ready to prove T7. Suppose that **Z**, a finite set of sentences, is inconsistent. (Note that, if inconsistent, **Z** must have at least one sentence.) Make the sentences of this set the first tableau of a semantic tableau derivation. Suppose that the derivation has an open terminal tableau. Then, by L18, there is an interpretation which makes true all the sentences in **Z**. But this is impossible since **Z** is supposed to be inconsistent. Therefore all terminal tableaux are closed. Then L20 tells us that the derivation terminates with 'A&~A', so that **Z**⊢A&~A, as was to be shown.

We have one more detail to complete. My proof of T7 is subject to the restriction that 'v' and '~' are the only connectives which appear in any of the sentences. We easily eliminate this restriction by exchanging sentences with other connectives for logical equivalents which use 'v' and '~' instead. At each stage we deal only with the main connective or, for negated sentences, with the negation sign and the main connective of the negated sentence. We rewrite rule R2 for sequential generation to read:

R2 *Sequential generation:*

a) Each line of the generated tableau is a new line of the same derivation as the generating tableau.
b) If a sentence of the form ~~**X** occurs on the generating tableau, enter **X** on the generated tableau.
c) If a sentence of the form ~(**X**v**Y**) occurs on the generating tableau, enter both ~**X** and ~**Y** as separate lines on the generated tableau.
d) If a sentence of the form **X**&**Y** occurs on the generating tableau, enter both **X** and **Y** as separate lines on the generated tableau.
e) If a sentence of the form **X**⊃**Y** occurs on the generating tableau, enter ~**X**v**Y** on the generated tableau.
f) If a sentence of the form **X**≡**Y** occurs on the generating tableau, enter (**X**&**Y**)v(~**X**&~**Y**) on the generated tableau.
g) If a sentence of the form ~(**X**&**Y**) occurs on the generating tableau, enter ~**X**v~**Y** on the generated tableau.
h) If a sentence of the form ~(**X**⊃**Y**) occurs on the generating tableau, enter both **X** and ~**Y** as separate lines on the generated tableau.
i) If a sentence of the form ~(**X**≡**Y**) occurs on the generating tableau, enter (**X**&~**Y**)v(~**X**&**Y**) on the generated tableau.
j) Reiterate all remaining sentences of the generating tableau as new lines of the generated tableau.

We could provide a more complicated version of R2 which would produce more efficient tableau derivations, but it's not worth the effort since true efficiency is only obtained with the truth tree method. In the next exercises you will show that the proof for the special case, using only the connectives 'v' and '~', extends to the general case covered by our reformulated R2.

EXERCISES

Generalizing the proof of T7 only requires checking three points.

13–13. I argued that a tableau derivation always comes to an end because each new tableau shortens at least one sentence of the previous tableau. This argument no longer works, at least not as just stated. Show that tableau derivations, with sentences using any sentence logic connectives and the new rule R2, always come to an end.

13–14. Check that when all terminal tableaux close, a tableau derivation created using the new rule R2 is a correct derivation. You will have to prove two new derived rules, one for biconditionals and one for negated biconditionals.

13–15. Reprove lemma L19 for our fully general tableau derivations.

13–16. Explain why the proof of completeness in this section shows that the primitive sentence logic derivation rules of chapter 5 (volume I) are complete for sentence logic.

CHAPTER CONCEPTS

As a check on your mastery of this material, review the following ideas to make sure you understand them clearly:

a) Rule Soundness
b) Sentence Rule
c) Subderivation Rule
d) Semantic and Syntactic Inconsistency
e) Semantic Tableau Derivation (or Tableau Derivation)
f) Tableau
g) Initial Tableau
h) Generating Tableau
i) Generated Tableau
j) Sequential Generation

k) Branching Generation
l) Derived Rule ~v
m) Closed Tableau
n) Minimal Sentence
o) Terminal Tableau
p) Terminal Interpretation
q) Ancestors of a Tableau