Tableaux

Formal Methods in Philosophy, 2017-03-27

Contents

1 Derivations 2

2 Tableaux 3

3 Tableaux in practice 6

4 Tableaux and Arguments 9

5 Order in Tableaux 10

6 Proofs about Derivations 11

7 Why Bother? 14

8 Soundness and Completeness 15

9 Soundness of the Tableaux Derivation System 15

10 Completeness for Tableaux 17

11 Exercises 19

A Tableaux Formally Defined 20

B Tableaux Completeness for Arguments with Infinitely Many Premises 21

C Compactness Revisited via Tableaux 23

1. Derivations

The key notion in logic is, as we've said before, *consequence*: the notion of some sentences following from another. So far, we've understood consequence in terms of *entailment*: ϕ follows from some sentences Γ just in case every structure in which all the sentences in Γ are true (in which Γ is satisfied), is also a structure in which ϕ is true.

There is, however, something a bit strange about understanding consequence in this way. To tell whether some sentences have a certain consequence ϕ, we talk about a class of structures in which those sentences are true, then whether that class is a subset of some further class which corresponds to the class in which ϕ is true. Rather than reasoning about Γ and ϕ, we reason about the structures in which Γ and ϕ are true. And this seems to be a detour. Compare these two English arguments:

(1)	If it's raining, I won't ride my bike.	(1′)	'If it's raining, I won't ride my bike' is true.
(2)	It's raining.	(2′)	'It's raining' is true.
(3)	Therefore, I won't ride my bike.	(3′)	Therefore, 'I won't ride my bike' is true.

The second argument does the same job as the first, when supplemented with this principle: For any sentence S,

(T) $\ulcorner S \urcorner$ is true iff S.

But it is needlessly complex, detouring through the metalanguage when the first object-language only argument does the same job. English is its own metalanguage, so the detour isn't so noticeable. But in $\mathscr{L}_1$, where the metalanguage is English, and therefore quite distinct from $\mathscr{L}_1$ itself, the results can be striking. The standard technique for evaluating an argument in $\mathscr{L}_1$ – truth tables – produces results that require a great deal more to understand them than is required to understand $\mathscr{L}_1$. $\mathscr{L}_1$ is a very simple language, which cannot express much. To understand a truth table, however, one has to understand the notion of truth; of a structure; of a sentence having a truth value in a structure; and all the set theoretic and mathematical background to these notions.

At present, we have no choice, since we have no tools for reasoning directly between $\mathscr{L}_1$ sentences rather than detouring via the semantics. So let us remedy that.

A *derivation* will be a certain structured collection of $\mathscr{L}_1$ sentences, and a *derivation system* will be a collection of rules that dictate how to create a derivation. We will use derivations to implement inference and argument in $\mathscr{L}_1$ directly, without needing to appeal to semantic notions like truth and satisfaction.

There are many derivation systems, giving rise to many different kinds of derivation. In this book, we look at two: *truth trees* or *(semantic) tableaux*; and *natural deduction*. In the present chapter we look at tableaux.

2. Tableaux

The idea of a tableau derivation is simple. A tableau is a *tree*, in a somewhat rarefied sense. (To begin with, it's upside down.) A tree, generally, is a collection of *nodes* that begin from a single *root* and are arranged into *branches*, descending downward from the root. Each node can either continue the branch it is currently on, or split that branch into two new branches. In a tableau, each node is occupied by a sentence of $\mathscr{L}_1$, and the decision whether to split a branch or continue it is determined by the tableau rules. The rules tell us, for each sentence-type, how to continue the tree. Beall and van Fraassen (2003: ch. 4) uses tableaux; other texts which use tableaux are Jeffrey (2006), Smith (2012) and Bostock (1997: ch. 4).

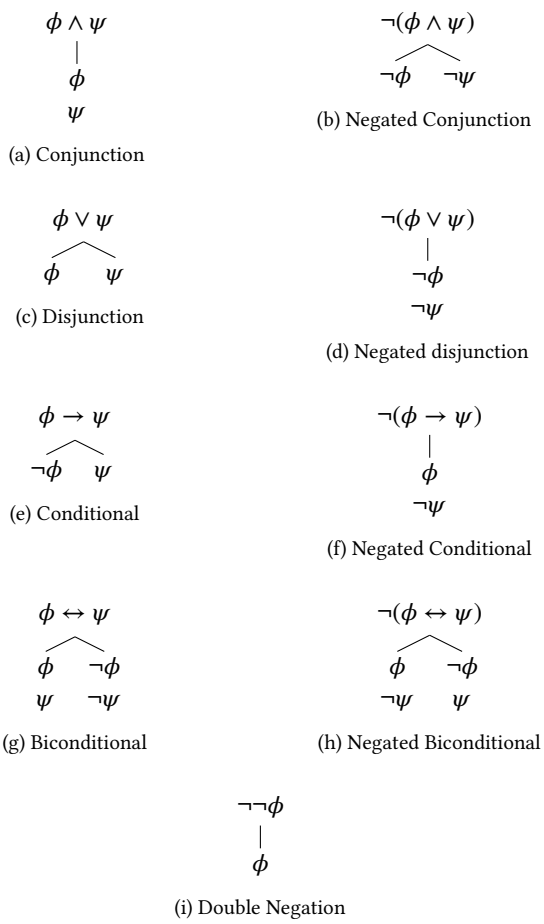
Let us be a little more formal. (For the even more formal definition, see the Appendix A.)

Definition 1 (Tableaux). A *tableau* is a tree, in the mathematical sense, with the following properties: it has a root node, with one or more branches descending from that root, each branch of which begins at the root and traces a sequences of nodes and terminates in a leaf node. (We say that a branch *passes through* the nodes that comprise it, and those nodes *lie on* the branch.) Because we permit branching, typically many branches will pass through a given single node and will thus *overlap*; and every branch passes through the root. (Thus our trees are not particularly arboreal.)

Each node of the tree contains one sentence of $\mathscr{L}_1$. A finite set of $\mathscr{L}_1$ sentences $\Sigma = \{\sigma_1, \ldots, \sigma_n, \ldots\}$ *generates* a tableau if the i-th node of each branch of the tableau is the sentence $\sigma_i \in \Sigma$, and each further node of the tree (if there are any) is the result of applying one of the patterns of sentences in Fig. 1 to a sentence that lies earlier on the same branch (i.e., closer to the root).

Definition 2 (Finished). A branch is called *finished* if, for every sentence ϕ at a node through which that branch passes, at least one of the resulting sentences appears on every branch that passes through the node at which ϕ appears (if the rule is not a branching one, this entails that the single resultant sentence ψ from applying the appropriate non-branch rule to ϕ is also already on that branch). A tableau in which every branch is finished is itself finished.

Theorem 1 (Tableaux are Finite). *Every finished tableau generated by a finite set contains only finitely many nodes.*



(a) Conjunction

(b) Negated Conjunction

(c) Disjunction

(d) Negated disjunction

(e) Conditional

(f) Negated Conditional

(g) Biconditional

(h) Negated Biconditional

(i) Double Negation

Figure 1: Propositional Tableau Rules

Proof. By König's Lemma (proved below as Lemma 7 in Appendix A), every infinite tableau has an infinite branch.

Every tableau generated by a finite set Σ has only finite branches. For each branch contains at most the elements of Σ, and any less complex constituents of Σ, since the tableau rules permit only less complex sentences to be added to a branch than sentences already on it. Since even the most complex member of Σ is still of finite complexity, even adding every less complex subsentence of any member of Σ to a branch will still involve adding only finitely many sentences. So even the longest possible branch added in accordance with the tableau rules is finite; so no tableau generated by a finite set has an infinite branch, and hence (contraposing König's Lemma), no tableau generated by a finite set is infinite. □

Definition 3 (Closed). A branch in a tableau is *closed* if there exist a sentence ϕ such that both ϕ and $\ulcorner\neg\phi\urcorner$ occur in nodes through which that branch passes; otherwise, it

is open. A tableau is closed iff every branch in it is closed.[1]

We are now in a position to describe our derivations.

Definition 4 (Tableau Derivation). A successful tableau derivation of ϕ from Σ is a finished closed tableau generated by $\Sigma \cup \{\neg\phi\}$.

Definition 5 (Derivability). We say that ϕ is *derivable* from Γ, written $\Gamma \vdash \phi$ iff there exists a finite set $\Sigma \subseteq \Gamma$ such that there is a successful tableau derivation of ϕ from Σ. If there is no successful tableau derivation of ϕ from any subset of Γ, we write $\Gamma \nvdash \phi$. (The weird subset clause is to permit derivability from infinite sets even though every finished $\mathscr{L}_1$ tableaux is finite.)

Definition 6 (Consistency). We say that a set of $\mathscr{L}_1$ sentences Γ is *inconsistent* iff there is a finished closed tableau generated by some finite set $\Sigma \subseteq \Gamma$, which we write $\Gamma \vdash$. A set of sentences is *consistent* iff it is not inconsistent, i.e., if every finished tableau generated by any finite subset of Γ contains at least one open branch.

In a sense, these definitions don't need any further explanation. This tells you the rules for manipulating a set of $\mathscr{L}_1$ sentences in such a way as to construct a successful derivation. What *justifies* our adopting these rules? Nothing, as yet: we can treat them as *purely formal*, like the rules of chess. The rules of chess tell you what things are permissible at each stage of the game. So too, the tableau rules tell you what extensions of the existing tree are permissible, given what is currently on the tree. When the tableau is finished, like when a game of chess is finished, one can look back and consider the aesthetics of the resulting tree or game. But there isn't much more that can be said by way of justification for why one constructed it in that way, other than that in chess one aims to finish in a winning position, and in the 'game' of tableaux, one aims to produce a finished tableau.[2]

In another sense, of course, we want our derivations to have certain properties, and we've used some suggestive notation. We *want* every inconsistent set to be unsatisfiable; and we want every successful derivation to correspond to an entailment. So we hope that $\Gamma \vdash \phi$ iff $\Gamma \vDash \phi$. We don't yet know if our hopes are to be granted; perhaps we made a mistake in formulating our derivation system. Later in this chapter, when we prove *soundness* and *completeness*, we'll see that our notation wasn't

[1] Some people prefer to define a closed branch more strictly as one on which a *sentence letter* and its negation both appear; it is a fairly immediate though tedious consequence of the tableau rules that a finished tableau will be closed in the stricter sense if it is closed in our sense. (The converse is trivial since sentence letters are sentences.)

[2] In chess, unlike tableaux, playing one permissible move often precludes playing other permissible moves, and the game finishes before every permissible move has been played. In tableaux, there is only one permissible move at each node, and a finished tableau is such that every node which could have a rule apply to it, has had that rule applied to it. So the aesthetic interest in tableaux is restricted to the order one considers the nodes, but that isn't a huge amount of artistic freedom.

presumptuous: it is the case that whenever Γ entails ϕ, ϕ is also derivable from Γ, and *vice versa*.

The key feature of tableaux, that we will prove below, is this: *if an unfinished tableau has a satisfiable branch, then any tableau that results from applying one of the tableau rules also has a satisfiable branch.* This is a claim about tableaux that needs to be proved, but informally it is the motivation for the rules we've chosen, and why tableaux basically amount to a mechanical test of satisfiability and hence validity.

3. Tableaux in practice

Suppose we want to construct a tableau. We begin with a finite set of sentences Σ, that will generate our tableau. Inscribe σ_1 as the root, and then each σ_i successively below it, all on the trunk with no branching.

If all the members of Σ were literals – either sentence letters or negated sentence letters – we're done: no tableau rules apply, and this is already a finished and boring tableau. But this is a rare case.

So we'll take our unfinished tableau, and bring it closer to being finished by applying a tableaux rule to some σ_i on the trunk. This involves adding to the bottom of *each* unfinished branch, some sentence that is less complex than σ_i. (Since we're trying to finish the tableau, we add sentences to the bottom of *each* branch on which the sentence we're applying the rule to occurs.)

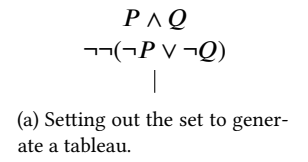
Once we've applied a rule to a sentence, and added the relevant sentences, we're done with that sentence. We can informally indicate that we're done with a sentence by drawing a box around it, as Beall and van Fraassen (2003) do. But these boxes – as well as the signs to indicate a closed branch – are useful scaffolding for the construction of a tableau by a person, not part of the official definition.

Eventually all the tableau rules that apply to sentences in the trunk will have been applied; we then apply the rules to sentences on particular branches. Since

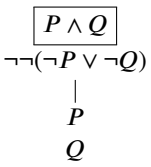
1. there were only finitely many sentences to start with, and
2. all of these sentences were of finite complexity, and
3. each tableau rule 'eliminates' a complex sentence in favour of either 1 or 2 sentences of less complexity,

it is easy to see that the process of constructing a tableau will finish after finitely many steps, after which one will have run out of sentences that the rules apply to. The tableau is now finished.

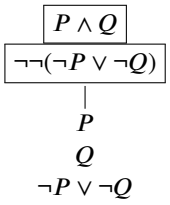
Then we check each branch for whether it is closed or open; there are only finitely many such branches (we've applied branching rules that produce finitely many branches only finitely many times, so we must have ended up with at most finitely



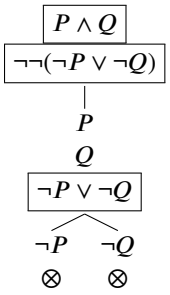
(a) Setting out the set to generate a tableau.



(b) First steps in the construction of a tableau: conjunction.



(c) Dealing with double negation.



(d) Dealing with disjunction using a branching tableau rule, and closing the result.

Figure 2: Steps in the construction of a tableau.

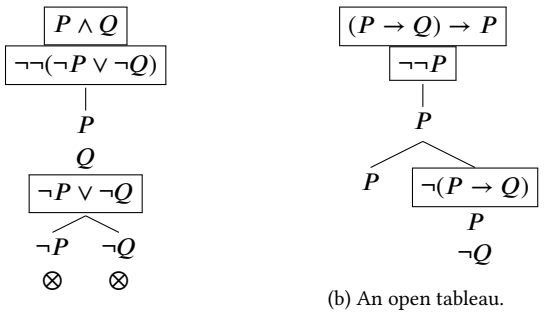
many branches), so we can check every branch and thus determine whether the whole tableau is open or closed.

This procedure is illustrated, for a schematic generating set, in Figure 2.

TABLEAU RULES: CONJUNCTION So, for instance, take the first sentence of the tableau above, $\phi \wedge \psi$. It is a conjunction; by the rules for conjunctions, we know that if this sentence occurs on the tableau, both conjuncts can be inscribed, extending the branch, as in Fig. 2(b).

TABLEAU RULES: DOUBLE NEGATION Next we turn to $\neg\neg(\neg\phi \vee \neg\psi)$; we see by the rules for negation, a double negation permits the inscription of the enclosed sentence without the two negation signs. So we inscribe $\neg\phi \vee \neg\psi$, and box the original sentence, as in Fig. 2(c).

TABLEAU RULES: DISJUNCTION Now we come to something of a problem. The rules for disjunction are *branching rules*: they don't tell us to inscribe new sentences on the same branch, but to inscribe one sentence on one branch, and another sentence on another: one for each disjunct, as in Fig. 2(d).



(a) A closed tableau.

(b) An open tableau.

Figure 3: Open and closed tableaux.

LEAVES, ROOTS AND BRANCHES Now we can break the sentences in the tableau down no further: we have the smallest parts of the sentences in question. We see, now, if the tableau is closed.

The topmost sentence is the *root* of a tableau. The sentences inscribed at the bottom of a tableau are the *leaves*. A *branch* is a sequence of sentences that begins with the root, and terminates with exactly one leaf, and includes every sentence above and below every sentence within the branch. So, for instance, $\langle \phi \wedge \psi, \neg\neg(\neg\phi \vee \neg\psi), \phi, \psi, \neg\phi \vee \neg\psi, \neg\phi \rangle$ is a branch. A branch is called *closed* if there are two sentences on the branch of the form ϕ and $\neg\phi$. So this example branch is closed; indeed, our whole tableau has only closed branches, and we call the whole tableau closed in this case. We mark a branch closed by putting a big bold $\otimes$ at its tip, as in Fig. 3(a). (Again, this is decoration; really, a branch is closed iff a sentence and its negation both occur, whether we write the $\otimes$ or not.)

Another example: Consider the schematic tableau (schematic, because it involves variables over sentences – ϕ, ψ, etc., rather than particular sentences P, Q, etc.) pictured in Figure 4. This demonstrates that $\phi \to \psi, \psi \to \chi \vdash \phi \to \chi$, for any ϕ, ψ, χ. (Note that I don't bother to add new sentences below the occurrence of χ on the rightmost branch even though strictly speaking our mechanical tableau construction procedure requires it: that branch is closed, so every extension of it would remain closed, as adding new sentences to an inconsistent set cannot render it consistent.)

Two more complicated examples:

1. Consider the argument $P \to (R \wedge Q_1); (Q_1 \vee P_1) \to \neg Q$; therefore $\neg(P \wedge Q)$. The tableau is shown in Fig. 5(a). This tableau is closed; hence the original argument was valid.

2. Consider the argument $(P \wedge \neg Q) \to P_1; \neg Q \vee \neg R$; therefore $\neg(P \wedge Q_1)$. The tableau is pictured in Fig. 5(b). As is readily seen, this tableau is open; there is no successful derivation corresponding to this argument.

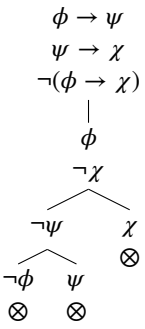
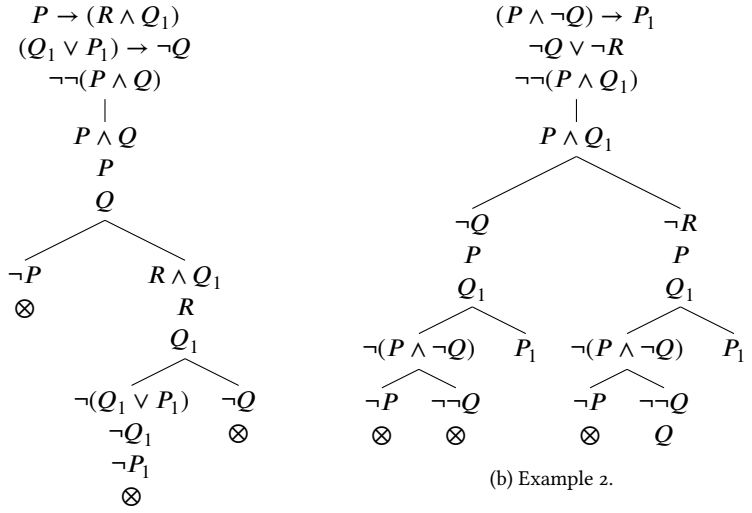


Figure 4: Tableaux for $\phi \to \psi, \psi \to \chi \vdash \phi \to \chi$.



(a) Example 1.

(b) Example 2.

Figure 5: Tableaux for examples 1 and 2.

4. Tableaux and Arguments

A tableau tests a set for inconsistency. But it doesn't actually care which is the conclusion and which are the premises. So one and the same tableau derivation can correspond to many arguments. Consider the tableau in Figure 6. Applying the definition of derivability, Definition 5, we see that this tableau corresponds to these arguments:

1. $\neg\neg P, \neg\neg(P \to Q) \vdash Q$;
2. $\neg\neg P, \neg Q \vdash \neg(P \to Q)$;
3. $\neg\neg(P \to Q), \neg Q \vdash \neg P$.

In fact, if we had been a little more liberal with our definition of negation, so that ϕ is the negation of $\neg\phi$, then there is a single tableau that corresponds to these three

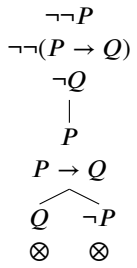


Figure 6: A tableau corresponding to many arguments

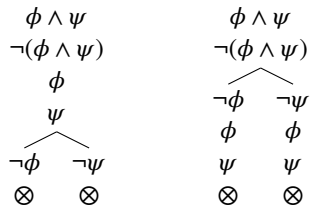
arguments: $P, P \to Q \vdash Q$; $P, \neg Q \vdash \neg(P \to Q)$; $P \to Q, \neg Q \vdash \neg P$. Truly it may be said that one persons *modus ponens* is another persons *modus tollens*: they correspond to exactly the same derivation!

5. Order in Tableaux

Does it matter which order you apply the tableau rules? No. Abstract a little bit from our pictorial representation of tableaux as trees. The fundamental constituents of a tableau are *branches*: sequences of $\mathscr{L}_1$ sentences. When we represent them pictorially, we draw branches that overlap in their initial segment (as all do, since the trunk of any tableau must contain at least one sentence at the root), as *coinciding*. But we could equally have decided to draw the branches as running parallel when they have the same members, and diverging only when they have different members.

Now compare the two tableau in Figure 7. They differ in which order the list and branch rules are applied. So the branches differ in the order of what is on them. But the set of sentences on each branch is the same, no matter in what order the rules are applied: in both tableau, there is a branch with the members $\{\phi \wedge \psi, \neg(\phi \wedge \psi), \phi, \psi, \neg\phi\}$ and a branch with the members $\{\phi \wedge \psi, \neg(\phi \wedge \psi), \phi, \psi, \neg\psi\}$. Since a sentence gets to be on a branch either by being on the trunk of the tableau, or by resulting from an application of a rule to an earlier sentence on the branch, it is evident that the order of application of the rules does not change the membership of a finished branch. (Suppose it did: then a sentence would be on one branch but not on the other, even though the trunk is the same, and both branches are finished. From where did that sentence come? Not from the trunk; and not from anything that results from an application of the rules to the trunk, since the branches are finished. But there is no other way for a sentence to get on a branch; so there is no such sentence.)

Since the property of being open (or closed) depends only on the members of the branch, and not on their order, rearranging the order of the branches preserves openness (and closedness). So if one tableau varies from another just in the order to which the rules have been applied to sentences on branches, then the first is closed iff the second is closed. So really, order doesn't matter: mechanically applying the



(a) List then branch.

(b) Branch then list.

Figure 7: Tableaux showing order irrelevant.

rules to Σ in any order you wish, to produce a finished tableau, will generate a closed tableau if *any* tableau generated by Σ closes.

6. Proofs about Derivations

Just as we proved the deduction theorem (??) for the semantic turnstile, we can prove a syntactic deduction theorem for the syntactic turnstile.

Theorem 2 (Syntactic Deduction). *$\Gamma, \phi \vdash \psi$ iff $\Gamma \vdash \phi \to \psi$.*

Proof. If: Assume $\Gamma \vdash \phi \to \psi$. Then every finished tableau T which is generated by $\Gamma \cup \{\neg(\phi \to \psi)\}$ is closed. Take one such tableau, where the first tableau rule applied was the negated conditional rule, Fig. 1(f), so that each branch of the tableau begins $\langle \gamma_1, \ldots, \gamma_n, \neg(\phi \to \psi), \phi, \neg\psi \rangle$. We have already dealt with $\neg(\phi \to \psi)$, so the sentences which close every branch on this tableau derive from Γ or from ϕ or $\neg\psi$. Hence, we can construct a closed tableau where every branch begins $\langle \gamma_1, \ldots, \gamma_n, \phi, \neg\psi \rangle$; hence the set $\Gamma \cup \{\phi, \neg\psi\}$ generates a closed tableau, hence $\Gamma, \phi \vdash \psi$. The only interesting case is if the original tableau had a branch which closed because of the presence of $(\phi \to \psi)$ on it. But then an application of the negated conditional rule to that node gives one branch with $\neg\phi$ occurring on it, which closes since ϕ occurs in the trunk; and another branch with ψ occurring on it, which closes since $\neg\psi$ occurs in the trunk.

Only if: We assume that $\Gamma, \phi \vdash \psi$. Then there is a closed finished tableaux that is generated by $\Gamma \cup \{\phi, \neg\psi\}$. If we modify this tableau by inserting $\neg(\phi \to \psi)$ above ϕ on every branch, the modified tableau is closed also. Since ϕ and $\neg\psi$ can come from $\neg(\phi \to \psi)$ (Fig. 1(f)), this modified tableau is a finished tableau generated by $\{\Gamma, \neg(\phi \to \psi)\}$, which means that $\Gamma \vdash \phi \to \psi$. □

Now I'll prove some lemmas, useful for proving a historically important theorem, Cut (Theorem 3).

Lemma 1. *$\Gamma \vdash \phi$ iff $\Gamma, \neg\phi \vdash$.*

Proof. $\Gamma \vdash \phi$ iff there is a finished closed tableau generated by $\Gamma \cup \{\neg\phi\}$ iff $\Gamma \cup \{\neg\phi\}$ is inconsistent (by definitions). □

When $\Gamma \cup \{\phi\} \vdash \psi$, we also write it – by analogy with the way we wrote entailments – $\Gamma, \phi \vdash \psi$.

Lemma 2. *If* $\Gamma, \neg\neg\phi \vdash$, *then* $\Gamma, \phi \vdash$.

Proof. If $\Gamma, \neg\neg\phi \vdash$, there is a finished closed tableau generated by $\Gamma \cup \{\neg\neg\phi\}$. Take such a tableau in which the first rule applied to the generating set was the $\neg\neg$ rule (Fig. 1(i)). Remove the node containing $\neg\neg\phi$: the tableau will still be closed. Why? Either Γ itself was already closed; or the tableau closed because of some application of a tableau rule to ϕ (the result of applying the double negation rule to $\neg\neg\phi$); or the tableau closed because $\neg\neg\phi$ and its negation occurred on the tableau. In the first two cases, it is obvious that removing $\neg\neg\phi$ made no difference to whether the tableau is closed. So consider the third case. There are two subcases: either $\neg\neg\neg\phi$ occurs on the tableau, or $\neg\phi$ does (both are negations of $\neg\neg\phi$). If the latter, the tableau is closed because both ϕ and $\neg\phi$ occur. If the former, since the tableau is finished, some application of the double negation rule was made to $\neg\neg\neg\phi$ at some stage, so that $\neg\phi$ occurs as well as ϕ, again closing the relevant branch. □

Lemma 3. *If* $\Gamma \vdash \neg(\phi \wedge \psi)$ *then* $\Gamma, \phi \vdash \neg\psi$.

Proof. If $\Gamma \vdash \neg(\phi \wedge \psi)$, then every finished tableau generated by $\Gamma \cup \{\phi \wedge \psi\}$ is closed. Consider such a tableau in which the first rule applied was the conjunction rule (Fig. 1(a)); this is also a finished closed tableau generated by $\Gamma \cup \{\phi \wedge \psi, \phi, \psi\}$. Remove the node containing $\phi \wedge \psi$, and the tableau remains closed. Either the tableau is closed because of the results of applying the tableau rules to ϕ and ψ; or some branch of the tableau is closed because it contains $\neg(\phi \wedge \psi)$. But, because the tableau is finished, that branch also contains either $\neg\phi$ or $\neg\psi$; either way, it will then close because of the presence of ϕ and ψ earlier. □

Lemma 4. *If* $\Gamma \vdash (\phi \wedge \psi)$, *then* $\Gamma \vdash \phi$ *and* $\Gamma \vdash \psi$.

Proof. If $\Gamma \vdash (\phi \wedge \psi)$, there is a finished closed tableau generated by $\Gamma \cup \{\neg(\phi \wedge \psi)\}$. Suppose the first rule applied in the negated conjunction rule (Fig. 1(b)), giving two branches, both beginning with the members of Γ and $\neg(\phi \wedge \psi)$, but one having $\neg\phi$ as a leaf, and the other with $\neg\psi$ as a leaf. Because the tableau is closed, every branch beginning like those two branches is closed. Suppose we delete the node containing $\neg(\phi \wedge \psi)$, but keep $\neg\phi$. We will also therefore prune off the branch with $\neg\psi$ occurring on it. But the pruned tableau remains closed, either because the tableau branches close due to the application of some tableau rule to $\neg\phi$, or because $\phi \wedge \psi$ occurs somewhere on the tableau, but since it is finished, so too then does ϕ, which makes it closed. The same goes, *mutatis mutandis*, for the other pruning. □

Let us restrict our attention to the sublanguage of $\mathscr{L}_1$ where the only connectives are $\neg$ and $\wedge$. We know that these connectives form a functionally complete set, and that this sublanguage is thus expressively adequate, so this is not a significant restriction – but it does make the proof of the following theorem considerably easier.

Theorem 3 (Cut). *If $\Gamma \vdash \phi$ and $\Gamma \vdash \neg\phi$ then $\Gamma \vdash$.*

Proof. We prove by induction on the complexity of ϕ. The base case: suppose ϕ is a sentence letter. Then there is a finished closed tableau generated by $\Gamma \cup \{\neg\phi\}$ and a finished closed tableau generated by $\Gamma \cup \{\phi\}$. Consider the second tableau, with the node containing ϕ removed – so it is a tableau generated by Γ alone. If that modified tableau remains closed, it must be because Γ itself already generated a closed tableau, since no tableau rules apply to a sentence letter ϕ. If the modified tableau could be rendered open by the removal of ϕ, then any branch that is newly open must contain $\neg\phi$. But since every finished tableau generated by $\Gamma \cup \{\neg\phi\}$ is closed, we can simply append a closed tableau generated by $\Gamma \cup \{\neg\phi\}$ to those open branches, thus showing that the modified tableau is closed, and hence that a tableau generated just by Γ is closed.

The induction step: ϕ is complex, and the induction hypothesis is that for all constituents ψ of ϕ, and any set Δ, if $\Delta \vdash \psi$ and $\Delta \vdash \neg\psi$, then $\Delta \vdash$. There are two cases in this sublanguage:

1. $\phi = \neg\psi$. We assume $\Gamma \vdash \phi$ and $\Gamma \vdash \neg\phi$, i.e., by Lemma 1, $\Gamma, \phi \vdash$ and $\Gamma, \neg\phi \vdash$. Because $\phi = \neg\psi$, we also have $\Gamma, \neg\psi \vdash$ and $\Gamma, \neg\neg\psi \vdash$. By Lemma2, $\Gamma, \psi \vdash$. By the induction hypothesis applied to the simpler constituent ψ, $\Gamma \vdash$.

2. $\phi = (\psi \wedge \chi)$. Again, we assume (a) $\Gamma \vdash (\psi \wedge \chi)$ and (b) $\Gamma \vdash \neg(\psi \wedge \chi)$. By Lemma 4 applied to (a), we have $\Gamma \vdash \chi$. But if a tableau generated by $\Gamma \cup \{\neg\chi\}$ closes, so does one generated by $\Gamma \cup \{\psi, \neg\chi\}$, so $\Gamma, \psi \vdash \chi$. By Lemma 3 applied to (b), we have $\Gamma, \psi \vdash \neg\chi$. By the induction hypothesis $\Gamma, \psi \vdash$. But from Lemma 4 applied to (a), we also have $\Gamma \vdash \psi$; by Lemma 1, $\Gamma, \neg\psi \vdash$; and by the induction hypothesis, $\Gamma \vdash$. □

What does the Cut Theorem show? Basically this: if you can get a tableau generated by Γ and $\neg\phi$ to close, and you can get a tableau generated by Γ and ϕ to close, then you could already have got a tableau generated by Γ to close: whatever appeal you made respectively to ϕ and $\neg\phi$ were inessential. The main interest of Cut, though, is in proving this theorem:

Theorem 4 (Transitivity of Derivability). *If $\Gamma \vdash \phi$ and $\phi \vdash \psi$, then $\Gamma \vdash \psi$.*

Proof. Assume that (a) $\Gamma \vdash \phi$, and (b) $\phi \vdash \psi$.

By (a), since a finished tableau generated by $\Gamma \cup \{\neg\phi\}$ is closed, so is one generated by $\Gamma \cup \{\neg\psi\} \cup \{\neg\phi\}$, so $\Gamma \cup \{\neg\psi\} \vdash \phi$.

By (b), since a finished tableau generated by $\{\phi\} \cup \{\neg\psi\}$ is closed, so is one generated by $\{\phi\} \cup \Gamma \cup \{\neg\psi\}$, so $\Gamma \cup \{\neg\psi\} \vdash \neg\phi$, by rearrangement and Lemma 1.

Here is an instance of the Cut theorem: if $\Gamma \cup \{\neg\psi\} \vdash \phi$ and $\Gamma \cup \{\neg\psi\} \vdash \neg\phi$, then $\Gamma \cup \{\neg\psi\} \vdash$. But we've established both conjuncts of the antecedent; so we may conclude, $\Gamma \cup \{\neg\psi\} \vdash$. By Lemma 1 again, $\Gamma \vdash \psi$, as required. □

What this shows is that any two-step derivation has a short-cut: if we can derive ψ from something we can derive from Γ, well, we could have derived ψ directly.

7. Why Bother?

A natural question at this point is why do we bother to define formally syntactic tableaux without reference to truth or structures? The first motive is to continue the process of abstraction from meaning that led us to formal logic in the first place: the continued presence of truth as a basic part of our judgements of validity is potentially confusing. What we want is a completely meaning-independent route to judge the validity of an argument, and truth and meaning are too closely connected. Secondly, some logicians have regarded the very notion of truth as suspect, and regard the activity of *derivation* as the key to mathematical and other reasoning. They, of course, desire a system that formalises what it means to give a derivation and makes no reference to the suspect notions, and even if we do not share their suspicions it seems well to separately define notions, if they are truly distinct. Derivability of a theorem and the truth of a theorem are quite different activities, that may not necessarily go together, as some past mathematicians with poor methods of proof have shown. (This harks back to our discussion of intuitionism/constructivism in **??**.)

There are some more interesting philosophical reasons as well. Many philosophers have been puzzled about the concepts of meaning and representation, particularly with the deep question of what special features does an object like a sentence possess in virtue of which a sentence can mean something—unlike, say, a rock, which is also an object but doesn't appear to have the capacity to mean anything. A derivation that doesn't require understanding, and hence doesn't seem to require the meaningfulness of the syntactic items upon which a derivation operates, might seem to hold out the potential of dissolving these philosophical perplexities over meaning and representation. It might also seem to hold out the promise of explaining why objects, like machines, that can perform syntactic manipulations, can seem to exhibit intelligent behaviour; because, as it turns out, those syntactic manipulations that constitute a derivation can be independently specified and yet (by the theorems to be proved in subsequent weeks) end up corresponding to the semantically significant notions of entailment and validity. So we might have the possibility of artificial intelligence. Indeed, we might be able to explain how it is that a 'meat computer', like the human

brain, can perform the intelligent action of which it is so clearly capable, without having to appeal to the essentially mysterious notions of consciousness or a 'soul'.[3]

8. Soundness and Completeness

A derivation system P in a given language is *sound* with respect to a semantic interpretation of the language just in case whenever there is a derivation which establishes $\Gamma \vdash_P \phi$, it is the case that $\Gamma \vDash \phi$.

A derivation system P in a given language is *complete* with respect to a semantic interpretation of the language just in case whenever it is the case that $\Gamma \vDash \phi$, there is a derivation which establishes $\Gamma \vdash_P \phi$.

Obviously we've already talked informally about the correspondence between $\vDash$ and $\vdash$ we are about to establish, and we designed our derivation systems with one eye on capturing all and only correct entailments in derivations. But those intentions in the design of these derivation systems could have gone awry, so it is important to check they have not. Today, we prove soundness for both tableaux and natural deduction derivation systems.

9. Soundness of the Tableaux Derivation System

We begin by proving soundness of the tableaux system: that is, every syntactic theorem is a semantic theorem. We start with an intermediate lemma.[4]

Lemma 5 (Tableau preserve satisfiability). *If a tableau is generated by a satisfiable negated sentence, there is at least one branch on that tableau such that every sentence on that branch is simultaneously satisfiable. That is, if there is an $\mathscr{L}_1$ structure such that $|\neg\phi|_{\mathscr{A}} = 1$, there is some branch B on a tableau generated by $\{\neg\phi\}$ such that for all sentences $\beta \in B$, $|\beta|_{\mathscr{A}} = 1$.*

Proof. We prove the lemma by *induction on the length of tableau branch B*. In effect, we consider a sequence of tableaux, such that each member extends the preceding member by application of one of the tableau rules.

BASE consider the smallest tableaux T generated by $\{\neg\phi\}$: the single node $\langle\neg\phi\rangle$. Since this is the only member of the only branch on T, and by hypothesis it is assigned 1 by $\mathscr{A}$, the lemma holds in this case.

[3] Of course this is not an argument that conscious awareness or the soul do not exist; rather it rests on the fact that, even if they did, they could hardly be expected to explain how the brain can exhibit intelligent behaviour.

[4] You might wish to contrast the proof of the same theorem (25.10) in Hodges (2001: 118–9), or that given in Bostock (1997: §4.5).

INDUCTION Assume that the lemma holds of a branch B on tableau T_n. Then we show the lemma holds of a branch B^+ on a tableau T_{n+1} obtained from T_n by one additional application of a rule in Fig. 1 to a sentence in B on T_n. There are three cases.

1. We apply the Double Negation Rule (Fig. 1(i)). Then some sentence on B is of the form $\neg\neg\chi$, and we add χ to the bottom of the branch to get B^+. Since the valuation function induced by the $\mathscr{L}_1$ structure $\mathscr{A}$ is classical valuation, $|\chi|_{\mathscr{A}} = |\neg\neg\chi|_{\mathscr{A}} = 1$, as required by the lemma.

2. We apply a non-branching rule to some sentence on B. Then we add two sentences to the bottom of B. If we applied, for example, the Negated Conditional rule (Fig. 1(f)) to $\neg(\psi \to \chi)$, we added ψ and $\neg\chi$ to the bottom of B to get B^+. By the rules on classical valuations, $|\neg(\psi \to \chi)|_{\mathscr{A}} = 1$ iff $|\psi \to \chi|_{\mathscr{A}} = 0$ iff $|\psi|_{\mathscr{A}} = 1$ and $|\chi|_{\mathscr{A}} = 0$ iff $|\psi|_{\mathscr{A}} = 1$ and $|\neg\chi|_{\mathscr{A}} = 1$, as required. Similarly for the Conjunction rule: $|\psi \wedge \chi|_{\mathscr{A}} = 1$ iff $|\psi|_{\mathscr{A}} = 1$ and $|\chi|_{\mathscr{A}} = 1$. So the lemma holds of B^+. (The Negated Disjunction rule is left as an exercise.)

3. We apply a branching rule to some sentence on B. We then get two new branches, B_1^+ and B_2^+, either of which can satisfy the lemma (but we only need one of them to satisfy it, since the lemma only says that there is at least one branch on a tableau where all the sentences on it are assigned 1 by $\mathscr{A}$). For instance, if we apply Disjunction (Fig. 1(c)) to $\psi \vee \chi$, $B_1^+ = B \cup \{\psi\}$ and $B_2^+ = B \cup \{\chi\}$. Since by the rules for classical valuations, $|\psi \vee \chi|_{\mathscr{A}} = 1$ iff either $|\psi|_{\mathscr{A}} = 1$ or $|\chi|_{\mathscr{A}} = 1$; that is, the lemma holds of at least one of B_1^+ or B_2^+. Similarly for the Negated Conjunction rule: If $\neg(\phi \wedge \psi) \in B$, then $\neg\phi \in B_1^+$ and $\neg\psi \in B_2^+$. $|\neg(\phi \wedge \psi)|_{\mathscr{A}} = 1$ iff $|\phi \wedge \psi|_{\mathscr{A}} = 0$ iff $|\phi|_{\mathscr{A}} = 0$ or $|\psi|_{\mathscr{A}} = 0$ iff $|\neg\phi|_{\mathscr{A}} = 1$ or $|\neg\psi|_{\mathscr{A}} = 1$. (The Conditional rule is left as an exercise.)

That completes the induction. □

Lemma 5 shows that the tableau rules preserve satisfiability: any tableau, finished or otherwise, generated by a satisfiable sentence has at least one simultaneously satisfiable branch. Now we are in a position to prove soundness.

Theorem 5 (Soundness). *If* $\vdash \phi$ *then* $\vDash \phi$.

Proof. Assume that $\vdash \phi$. Then every finished tableau generated by $\{\neg\phi\}$ is closed.

Assume for *reductio* that $\nvDash \phi$. Then there is an $\mathscr{L}_1$structure $\mathscr{A}$ that makes $\neg\phi$ true: $|\neg\phi|_{\mathscr{A}} = 1$. By Lemma 5, there is a finished tableau T generated by $\neg\phi$, with a branch B such that for every $\beta \in B$, $|\beta|_{\mathscr{A}} = 1$. Since the valuation function induced by $\mathscr{A}$ is classical, for any $\mathscr{L}_1$ sentence ψ if $\psi \in B$, then $\neg\psi \notin B$ – since for any structure A, $|\psi|_{\mathscr{A}} \neq |\neg\psi|_{\mathscr{A}}$, and every sentence on B is satisfied in some structure and hence all have the same truth value in that structure. Hence B cannot be closed; hence T is not

closed. But T is generated by $\{\neg\phi\}$, and our initial assumption was that any finished tableau generated by $\{\neg\phi\}$ is closed. So our *reductio* hypothesis must be wrong; that is, it must instead be true that $\vDash \phi$. But now we've shown $\vDash \phi$ on the assumption that $\vdash \phi$. □

Having proved the theorem, we can easily extend it to the general case of an arbitrary argument.

Theorem 6 (General Soundness). *If $\Gamma \vdash \phi$ then $\Gamma \vDash \phi$.*

Proof. Recall that $\Gamma \vdash \phi$ iff there is a finished closed tableau generated by $\Sigma \cup \{\neg\phi\}$, where Σ is a finite subset of Γ. So $\Gamma \vdash \phi$ iff there exists a finite $\Sigma \subseteq \Gamma$ such that $\Sigma \vdash \phi$.

Assume that $\Gamma \vdash \phi$. Then there is a finite set $\Sigma \cup \{\neg\phi\} = \{\sigma_1, \ldots, \sigma_n, \neg\phi\}$ which generates a finished closed tableau. By repeated applications of the Deduction theorem for tableaux (Theorem 2), $\vdash (\sigma_1 \to (\sigma_2 \to (\ldots(\sigma_n \to \phi))))$. By Soundness (Theorem 5), $\vDash (\sigma_1 \to (\sigma_2 \to (\ldots(\sigma_n \to \phi))))$. By repeated applications of the Deduction theorem for entailment (??), $\Sigma \vDash \phi$. Since $\Sigma \subseteq \Gamma$, $\Gamma \vDash \phi$, by Weakening (??).[5] □

10. Completeness for Tableaux

The proof of completeness for the tableaux derivation system is relatively straightforward. The idea is simple: we show that, if set of sentences Γ is consistent, then Γ is satisfiable. Recall that (Definition 6) a set of sentences Γ is consistent if any finished tableau generated by Γ has an open branch. We will use this fact to show that there is a structure in which all the members of Γ are true. To do this, we have to show an intermediate lemma, to the effect that an open branch can be used to determine a structure which makes each sentence on the branch true.

Lemma 6 (Hintikka's Lemma). *If B is an open branch on a finished open tableau generated by a set of $\mathscr{L}_1$ sentences Γ, then there is an $\mathscr{L}_1$ structure $\mathscr{B}$ such that for every sentence β on B, $|\beta|_{\mathscr{B}} = 1$.*

Proof. Suppose there is an open branch on a finished tableau generated by Γ, B. Define an $\mathscr{L}_1$ structure $\mathscr{B}$ as follows: for every sentence letter s,

$$\mathscr{B}(s) = \begin{cases} 1 & \text{if } s \text{ occurs in some node on } B; \\ 0 & \text{if } \neg s \text{ occurs in some node on } B; \\ 1 & \text{otherwise.} \end{cases}$$

[5] Actually (because Weakening as stated only applies to a single sentence) what we need is a stronger claim, that if $\Sigma \vDash \phi$, then for any set Δ, $\Sigma, \Delta \vDash \phi$. And we then let $\Delta = \Gamma \setminus \Sigma$, so that $\Sigma \cup \Delta = \Gamma$, and the result follows.

Because $\boldsymbol{B}$ is an open branch, this successfully defines an $\mathscr{L}_1$ structure: no sentence letter and its negation both occur on $\boldsymbol{B}$, so this definition assigns one and only one value to each $\mathscr{L}_1$ sentence letter. (It is thus a classical structure, and induces a classical Boolean valuation function.)

We show by induction on complexity on sentences a the branch that for every sentence β on $\boldsymbol{B}$, $|\beta|_{\mathscr{B}} = 1$. Because this is a tableau construction, the nicest way to think about the induction on complexity is that the base case comprises those sentences to which no tableau rule applies, and the induction step involves consideration of sentences to which tableau rules apply. (Our base 'case' thus encompasses literals, rather than just sentence letters.)

BASE CASE: Suppose β occurs on $\boldsymbol{B}$, and is a literal: either is a sentence letter or a negated sentence letter. If the former, by construction, $\mathscr{B}(\beta) = 1$, so $|\beta|_{\mathscr{B}} = 1$, by the definition of an $\mathscr{L}_1$ structure. If the latter, i.e., $\beta = \neg s$ for some s, $|\beta|_{\mathscr{B}} = 1 - |s|_{\mathscr{B}} = 1 - \mathscr{B}(s) = 1 - 0 = 1$.

INDUCTION STEP: Suppose β is a complex sentence, and the lemma holds of its less complex constituents. There are three cases of interest, corresponding to different tableau rules: β is a double negation; a branch rule can be applied to β; or a list rule can be applied to β.

1. β is a double negation, $\neg\neg\phi$. Then, because this is a finished branch, ϕ appears on β; because the lemma holds of less complex constituents, $|\phi|_{\mathscr{B}} = 1$. So $|\neg\phi|_{\mathscr{B}} = 0$, and $|\neg\neg\phi|_{\mathscr{B}} = 1$, i.e., $|\beta|_{\mathscr{B}} = 1$.

2. β can have a list rule applied to it. Let us choose Negated Conditional, so $\beta = \neg(\phi \to \psi)$. Then ϕ and $\neg\psi$ appear on $\boldsymbol{B}$, because the tableau is finished, and by the induction hypothesis $|\phi|_{\mathscr{B}} = |\neg\psi|_{\mathscr{B}} = 1$. Therefore by the rules on the valuation function, $|\phi \to \psi|_{\mathscr{B}} = 0$, and $|\neg(\phi \to \psi)|_{\mathscr{B}} = 1$, i.e., $|\beta|_{\mathscr{B}} = 1$, as required. Analogous reasoning applies to the other list rules.

3. β can have a branch rule applied to it. Let $\beta = \phi \vee \psi$. Then either ϕ or ψ appears on $\boldsymbol{B}$; hence either $val\phi\boldsymbol{B} = 1$ or $|\psi|_{\mathscr{B}} = 1$, which by the rules on the valuation function, means that $|\phi \vee \psi|_{\mathscr{B}} = 1$, i.e., $|\beta|_{\mathscr{B}} = 1$, as required. Analogous reasoning applies to the other branch rules.

That suffices to show the lemma. □

Theorem 7 (Completeness for tautologies). *If* $\vDash \phi$ *then* $\vdash \phi$.

Proof. We prove the equivalent contrapositive, namely, that if it is *not* the case that ϕ is a syntactic theorem (which we write '$\nvdash \phi$') then it is not the case that ϕ is a semantic theorem ('$\nvDash \phi$').

Assume that $\nvdash \phi$. Then there is a finished open tableau generated by $\{\neg\phi\}$ which has an open branch, $\boldsymbol{B}$. Let $\mathscr{B}$ be the structure induced by $\boldsymbol{B}$, in accordance with Hintikka's Lemma (Lemma 6). By that lemma, every sentence occurring on $\boldsymbol{B}$ is true in $\mathscr{B}$. Since $\neg\phi$ appears on $\boldsymbol{B}$, as the root, $|\neg\phi|_{\mathscr{B}} = 1$. Hence there is a structure which makes ϕ false, so $\nvDash \phi$. □

Theorem 7 states that every tautology is derivable. This can easily be extended to arguments with finitely many premises:

Theorem 8 (Completeness for finite arguments). *When Γ is finite, if $\Gamma \vDash \phi$ then $\Gamma \vdash \phi$.*

Proof. If $\Gamma \vDash \phi$, and Γ is a finite set $\{\gamma_1, \ldots, \gamma_n\}$, then (by repeated applications of the Deduction theorem for entailment (**??**), $(\gamma_1 \to (\gamma_2 \to (\ldots(\gamma_n \to \phi)\ldots)))$ is a tautology. By Theorem 7, $\vdash (\gamma_1 \to (\gamma_2 \to (\ldots(\gamma_n \to \phi)\ldots)))$. By repeated applications of the deduction theorem for tableaux (Theorem 2), $\Gamma \vdash \phi$. □

11. Exercises

1. Prove that if $\vdash \phi$, then for any ψ, $\vdash \psi \to \phi$.
2. Provide tableaux derivations demonstrating the following:
   - **(a)** $\vdash ((P \leftrightarrow (P \vee P)) \wedge (P \leftrightarrow (P \wedge P)))$;
   - **(b)** $\vdash ((P \leftrightarrow Q) \leftrightarrow ((P \to Q) \wedge (\neg Q \vee P)))$;
   - **(c)** $\vdash ((P \to Q) \leftrightarrow ((P \wedge \neg Q) \to (P \wedge \neg P)))$;
   - **(d)** $\vdash (((P \to Q) \wedge \neg Q) \to \neg P)$;
   - **(e)** $\vdash ((P \to Q) \to ((P \vee R) \to (Q \vee R)))$.
3. Show, without making use of the soundness and completeness theorems, that the following features hold of the tableaux derivation system:
   - **(a)** If $\Gamma, \phi, \phi \vdash \psi$, then $\Gamma, \phi \vdash \psi$.
   - **(b)** If $\Gamma, \phi, \psi \vdash \chi$, then $\Gamma, \psi, \phi \vdash \chi$.
   - **(c)** If $\Gamma \vdash \phi$, then $\Gamma, \psi \vdash \phi$.
4. Assuming the soundness theorem, prove that if $\Gamma \vDash \phi$ is an *incorrect* sequent then $\Gamma \vdash \phi$ is also incorrect.
5. Consider an operator $\odot$ defined by the following tableau rules:

(a) Which truth-function does $\odot$ express? (I.e., which truth function is characterised by these tableau rules?)

(b) If these rules were the only rules for a system of tableau, under what conditions should a branch be counted as closed?

(c) Are these tableau rules sound?

A. Tableaux Formally Defined

Definition 7 (Tree). A tree is a triple $\langle N, R, \rho \rangle$, where N is a non-empty set of *nodes*, and R is a binary relation on N, and ρ is a function that assigns each member of N a natural number, its *rank*, such that

1. There exists an $r \in N$ – the *root* – such that there is no $x \in N$ such that $R(r, x)$, and $\rho(r) = 0$.
2. If the rank of x is n, and $R(x, y)$, then the rank of y is $n + 1$.

A *branch* B is a subset of N such that if $y \in B$ and $R(x, y)$, then $x \in B$.

That is the general idea of a tree; a tableaux is a very special type of tree, to wit:

Definition 8 (Finite Tableaux again). A tableaux generated by a finite set Γ is any tree $\langle N, R, \rho \rangle$ where

1. All the members of N are $\mathscr{L}_1$ sentences;
2. N is finite;
3. For any $\phi, \psi \in N$, $R(\phi, \psi)$ only if either (i) ψ comes from an application of one of the tableau rules in Figure 1, either to ϕ, or to some ξ occurring on the same branch as ϕ such that $\rho(\xi) < \rho(\phi)$; or (ii) in some enumeration of Γ, $\phi = \gamma_n$ and $\psi = \gamma_{n+1}$.

Our definitions of closed, open, and finished are as before.

We prove, as promised, a Lemma. A *descendent* of a node n is any node n' which appears on a branch passing through n, such that $\rho(n') > \rho(n)$; i.e., on the same branch but with a higher rank.

Lemma 7 (König's Lemma). *If a tableau has infinitely many wffs appearing on it, then it has an infinite branch.*

Proof. Obviously a tableau has a branch which only contains nodes having infinitely many descendents iff that branch is infinitely long. We prove by induction on rank in the tableau T that an infinite tableau has a branch on which every node has infinitely many descendents.

Basis: If T has infinitely many nodes, then since every node is a descendent of the root, the root node of T has infinitely many descendents; and the root has rank 0.

Induction: The tableau rules ensure that any node n on T has either one or two immediate successors. Assume that n has infinitely many descendents; we show that at least one immediate successor n^+ (i.e., a descendent such that $\rho(n^+) = \rho(n) + 1$) of n also has infinitely many descendents.

- If we applied a *list rule* to n, and n has infinitely many descendents, it is obvious that any immediate successor n^+ has infinitely many descendents (since having one fewer than infinitely many descendents still involves having infinitely many descendents).
- If we applied a *branch rule* to n, then if both of n's immediate successors didn't have infinitely many descendents (i.e., each has only finitely many descendents), then n would not have infinitely many descendents, since its descedents are just those that occur on both branches stemming from it; so at least one of n's immediate successors n^+ must have infinitely many descendents. □

For a more general proof, for trees with tree rules that permit any finite number of branches to stem from a given node (whereas in tableau, at most 2 branches stem from any node), see Beall and van Fraassen (2003: 152).

B. Tableaux Completeness for Arguments with Infinitely Many Premises

Extending the proof of completeness to arguments with infinitely many premises is trickier. Obviously, given the Compactness Theorem, whenever $\Gamma \vdash \phi$, there is some finite subset of Γ which entails ϕ, and we can apply Theorem 8. But to prove it directly, without invoking compactness, is trickier.[6]

We need to modify our definition of a tableau slightly. Up to now, we've defined a tableau generated by Σ to have every member of Σ as the initial members of every branch on the tableau. We will now modify our definition slightly.

Definition 9 (Tableaux Revised). A *tableau* generated by $\Gamma \cup \{\neg\phi\}$ contains $\neg\phi$ at the root node, with one or more branches descending from that root, such that every node on any branch is either a member of Γ, or results from the application of a tableau rule to some sentence in a node on that branch of lower rank.

This definition is not explicitly restricted to finite generating sets, unlike the definition above, and so can handle arguments with infinitely many premises.[7] We also alter our definition of a finished branch/tableau:

[6] You might wish to contrast the proof of the same result (25.11) in Hodges (2001: 119–20), or that given in Bostock (1997: §4.6). Harris' lecture notes Harris (2008: esp. lectures 13–15) are clear and helpful.

[7] If you prefer the more formal style of the appendix, I offer this definition:

Definition 11 (Finished Revised). A branch on a tableau generated by $\Gamma \cup \{\neg\phi\}$ is *finished* if either (i) it is closed, or (ii) every member of Γ occurs on the branch, and for every sentence occurring on the branch, if a list rule applies to it, any resulting sentences also occur on the branch, and if a branch rule applies to it, at least one of the resulting sentences also occurs on the branch.

As expected, $\Gamma \vdash \phi$ under these new definitions iff there is a finished closed tableau generated by $\Gamma \cup \{\neg\phi\}$.

Here is the key new idea: If $\tau_0, \tau_1, \ldots$ is a finite or infinite sequence of finite tableaux, each generated by $\Gamma \cup \{\neg\phi\}$ such that τ_{n+1} results from τ_n by either applying some tableaux rule to a sentence on some branches of τ_n, or by adding some $\gamma \in \Gamma$ to the end of some branches in τ_n, then $\tau = \bigcup_i \tau_i$ is a tableau too. That is, the *limit* of a monotonically increasing sequence of tableau generated by $\Gamma \cup \{\neg\phi\}$ is also a tableau generated by $\Gamma \cup \{\neg\phi\}$.

We prove a lemma:

Lemma 8. *If Γ is a (possibly infinite) set of propositions, there is a finished tableau generated by Γ.*

Proof. Let Γ be enumerated $\gamma_0, \gamma_1, \ldots, \gamma_n, \ldots$. Define τ_0 to be the degenerate 'tree' with a single instance of γ_0 at the root.

Suppose we have constructed $\tau_0, \ldots, \tau_n$.

- If n is even (or zero), recursively apply the appropriate tableau rule to $\gamma_{n/2}$ wherever it occurs on any branch of τ_n, and to the results of that application of the tableau rule, and to the results of that further application, etc. (Since every sentence has finite complexity, and the tableau rules reduce complexity, this process terminates after finitely many steps.) Let the resulting tableau be τ_{n+1}. (If $\gamma_{n/2}$ is a literal, let $\tau_{n+1} = \tau_n$.)
- If n is odd, add $\gamma_{(n+1)/2}$ to the bottom of every unfinished branch in τ_n to construct τ_{n+1}.

Let $\tau = \bigcup_i \tau_i$. τ is a tableau generated by Γ, since by construction every sentence which occurs in any node on τ is either a member of Γ, or results from some number of applications of the tableau rules to a member of Γ. And τ is finished, since every

Definition 10 (Tableaux yet again). A tableau generated by Γ is any tree $\langle N, R, \rho\rangle$ where

1. All the members of N are $\mathscr{L}_1$ sentences;
2. For any $\phi, \psi \in N$, $R(\phi, \psi)$ only if either (i) ψ comes from an application of one of the tableau rules either to ϕ, or to some ξ occurring on the same branch as ϕ such that $\rho(\xi) < \rho(\phi)$; or (ii) ϕ is the last added result of successively applying the tableau rules to γ_n and its tableau-consequences, and ϕ is γ_{n+1}.

sentence that is ever added at some stage n has all appropriate applications of the tableau rules made to it at stage $n + 1$. □

If Γ is infinite, the finished tableau τ generated by Γ in accordance with Lemma 8 is also infinite.

Now we can prove completeness:

Theorem 9 (General Completeness). *If $\Gamma \vDash \phi$, then $\Gamma \vdash \phi$.*

Proof. We prove the contrapositive. Assume that $\Gamma \nvdash \phi$. Then, by our revised definition of a tableau, and Lemma 8, there is a finished open (possibly infinite) tableau generated by $\Gamma \cup \{\neg\phi\}$. (We let $\neg\phi$ be the root of τ_0, and then successively add the γ_is.) By the construction of Lemma 8, each member of $\Gamma \cup \{\neg\phi\}$ is added to every unfinished branch at some stage τ_i. So if τ is finished and open, any open branch B in τ must contain every member of $\Gamma \cup \{\neg\phi\}$. By Hintikka's Lemma (Lemma 6), any open branch on a finished tableau is such that the set of sentences occurring in nodes on that branch is satisfiable. Putting these results together, $\Gamma \cup \{\neg\phi\}$ is satisfiable, i.e., $\Gamma \nvDash \phi$. □

C. Compactness Revisited via Tableaux

Theorem 10 (Finite Tableaux Derivations). *If there is a tableau derivation establishing that $\Gamma \vdash \phi$, then there is a finite tableau derivation of ϕ using only finitely many members of Γ.*

Proof. Let τ be a tableau derivation of ϕ from Γ. Every branch on τ is finished and closed. That is, for each branch B, for some ψ, both ψ and $\neg\psi$ occur on B. But because τ is a tableau, each sentence on it occurs in a node of some *rank*; and each rank is finite. (There is no upper bound to ranks, but each rank is a natural number, by the construction of a tree.) So $\rho(\psi)$ and $\rho(\neg\psi)$ are both finite. For each such branch, the *pruned* branch B^- is the result of cutting any nodes from B of rank greater than $\max(\rho(\psi), \rho(\neg\psi))$. The *repaired* branch $B^\dagger$ is the result of applying any tableaux rules to sentences on the pruned branch B^- where the pruning has made that branch unfinished. Since each pruned branch is finite, each repaired branch is also finite. Let τ^- be the result of pruning and repairing every branch on τ. By König's Lemma (7), τ^- is finite, since it is a binary branching tableau with only finite branches. Thus only finitely many members of Γ appear on τ^-; gather them together into a set Σ. τ^- is a finished closed tableau generated by $\Sigma \cup \{\neg\phi\}$, i.e., showing $\Sigma \vdash \phi$ where Σ is a finite subset of Γ. □

Theorem 11 (Compactness). *If Γ is finitely satisfiable, Γ is satisfiable.*

Proof. We prove the contrapositive. Assuming without loss of generality that Γ is non-empty, then $\Gamma = \Delta \cup \{\phi\}$, and assume that $\Gamma = \Delta \cup \{\phi\}$ is unsatisfiable. That is, $\Delta \vDash \neg\phi$. By General Completeness (Theorem 9), $\Delta \vdash \neg\phi$. By Theorem 10, there is a finite tableau derivation of $\neg\phi$ using only finitely many members of Δ. Thus, some finite subset of Δ, Δ_0, is such that $\Delta_0 \vdash \neg\phi$. By soundness, $\Delta_0 \vDash \neg\phi$; i.e., $\Delta_0 \cup \{\phi\}$ is unsatisfiable. But $\Delta_0 \cup \{\phi\}$ is a finite subset of Γ, so Γ is not finitely satisfiable, as needed to be shown. □

References

Beall, JC and van Fraassen, Bas C. (2003) *Possibilities and Paradox*. Oxford: Oxford University Press.

Bostock, David (1997) *Intermediate Logic*. Oxford: Oxford University Press.

Harris, Kenneth (2008), Lecture notes for *Math 481: Mathematical Logic*, University of Michigan, Fall 2008, kaharris.org/teaching/481/lectures/index.html.

Hodges, Wilfrid (2001) *Logic*, 2nd ed. London: Penguin.

Jeffrey, Richard C. (2006) *Formal Logic: Its Scope and Limits*, 4th ed. Indianapolis: Hackett.

Smith, Nicholas J. J. (2012) *Logic: The Laws of Truth*. Princeton: Princeton University Press.