The Completeness of Tableau

1. Recall that to be complete, Tableau must meet the following condition:

   \[(C) \text{ If } \Gamma \models \varphi, \text{ then } \Gamma \models \neg \varphi\]

That is, if \(\Gamma\) entails \(\varphi\) then it will have a closed tableau. Once again, we show that this condition holds by turning it around and proving \((C)\)’s contrapositive:

\[(C^*) \text{ If not } \Gamma \models \neg \varphi, \text{ then not } \Gamma \models \varphi\]

Essentially this means that if the tableau for \(\Gamma\) on the left and \(\varphi\) on the right cannot be closed, then \(\Gamma\) doesn’t entail \(\varphi\) – that there is an interpretation \(I\) in which all of \(\Gamma\) is true and \(\varphi\) is false.

2. Our overall strategy will be to consider an open path for a fully developed tableau, and then construct from it an interpretation for the formulas along that path that will make all the formulas on the left true and the ones on the right false. It follows, then, that this will be an interpretation that shows that \(\Gamma\) does not entail \(\varphi\).

3. There are, however, a couple of wrinkles we will have to consider, since not all tableaus can be fully developed. Specifically, when the set \(\Gamma\) has an infinite number of formulas, then if its tableau doesn’t close, it will stretch on without end. For that reason we shall, for the moment, restrict our consideration to finite tableaus, which can be developed fully. We shall lift this restriction to finite tableau in the set of notes on compactness.

4. In a basic logic course which taught you how to develop “truth trees”, you might have learned how to construct a counterexample to an argument form an open branch of its tree. That’s just what we’re going to do here.

5. So with the restriction above in mind, let’s consider a fully developed tableau which cannot be closed. In particular, let’s attend to an open path along that tableau. From that open path, we shall construct an interpretation (call it \(I\)) for all the formulas along that path. Since we are not dealing with quantifier formulas, an interpretation consists simply in an assignment of truth values to the proposition letters that appear in the formulas on the path. Our procedure will be quite simple and straightforward:

   (1) If an atomic formula (single proposition letter) appears on the left of the path, assign that proposition letter the value True.
   (2) If an atomic formula appears on the right of the path, assign that proposition letter the value False.
   (3) Assign any other proposition letters arbitrary truth values. This rule is required only to ensure that we have a complete interpretation. We will not need to appeal to it later.

We know that since the path we are considering does not close, we can be assured that
this procedure will not assign the values true and false to the same proposition letter.

As an example, consider the fully developed trees for the following:

\[((R \& S) → \neg T), ((P \lor Q) → R) |-/- (P → \neg T)\]

\[(P → (\neg R \& Q)), \neg(P → Q) |-/- ((P \lor Q) → R)\]

Now pick open paths along these trees and follow the procedure described above for constructing interpretations. Now think about how that procedure is bound to generate a counterexample for the corresponding entailment.

6. The claim then is that this interpretation I is one that makes every formula on the left of the path (including Γ) true and every formula on the right of the path (including φ) false. And the demonstrated existence of such an interpretation will clearly show that Γ cannot entail φ.

7. Our demonstration will be an argument by induction on the length of formulas occurring along our path. Our inductive hypothesis will be that every formula shorter than an arbitrary one, χ, will be true if it is on the left and false if it is on the right.

(i) Base Case; χ is atomic (and the inductive hypothesis is moot):

Trivially, our procedure simply stipulates that χ will be true on I if it’s on the left and that χ will be false on I if it’s on the right, which is precisely what we’re aiming to show.

(ii) χ is a negation: \(\neg \phi\)

Suppose χ is on the left of our path. By our rules of development, then, \(\phi\) must be on the right of the path. But \(\phi\) is clearly shorter than χ, and by our inductive hypothesis, it must then be false on I. So I must make χ true.

Parallel reasoning applies if χ is on the right.

(iii) χ is a conjunction: \(\phi \& \psi\)

First suppose that χ is on the left of the path. By our rules of development, both \(\phi\) and \(\psi\) must also lie on the left. But both of these formulas are clearly shorter than χ, so our inductive hypothesis holds, and thus they both must be true on I. But this means that their conjunction, which is χ, must also be true on I.

Now suppose that χ is on the right side of the path. By our rules of development, our open path must be one in which either \(\phi\) or \(\psi\) is on the right of the path. But both of these formulas are clearly shorter than χ, so our inductive hypothesis holds, and thus one or the other must be false on I. But this dictates that their conjunction, which is χ, must also be false on I.
So if \( \chi \) is on the left, it must be true, and if \( \chi \) is on the right, it must be false, which is exactly what we want to show.

(iv) \( \chi \) is a disjunction: \( \varphi \vee \psi \)

I leave this to the reader. The justification is very similar to that above.

(v) \( \chi \) is a conditional: \( \varphi \rightarrow \psi \)

I also leave this to the reader. Its justification is only slightly more complex.

Thus completes our demonstration of basic completeness for truth-functional logic: if a fully developed tableau is open, then the sequent it represents must not be correct.

At this point, another very nice exercise would be for one to come up with appropriate tableau rules for either the stroke (\( n/\text{and} \)) or dagger (\( n/\#\text{or} \)) operations, and then to prove both the soundness and the completeness of those rules.