

PY1003 Introduction to Logic
Lecture 6

Last time we looked at five of the rules for constructing truth trees:

- THE RULE FOR ' \wedge '
- THE RULE FOR ' \vee '
- THE RULE FOR ' $\neg\neg$ '
- THE RULE FOR ' \rightarrow '
- THE RULE FOR ' \leftrightarrow '

Consistency of a set of sentences:

A set Σ of sentences of Sentential Logic is consistent iff there is an interpretation (an assignment of truth-values to the atomic sentences) which makes them all true together. If there is no such interpretation, Σ is inconsistent.

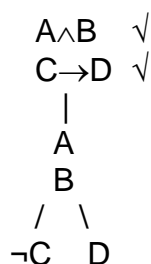
$\{A, B \wedge C\}$ is a consistent set of sentences. If A is true and B is true and C is true, then all the sentences in the set $\{A, B \wedge C\}$ come out true.

$\{A, \neg A\}$ is an inconsistent set. Any interpretation which makes A true makes $\neg A$ false. So there is no interpretation on which both members of this set are true.

A *model* of a set of sentences is an interpretation which makes them all true together. Consistent sets have a model; inconsistent sets do not.

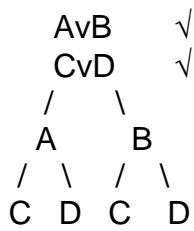
Starting a tree with more than one sentence

Example:



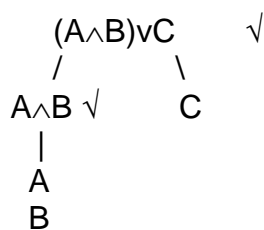
When applying the rule for a sentence, you **MUST** apply it to every branch on which that sentence appears (unless the branch is finished or 'closed', in a sense to be described shortly).

Example:



But this doesn't just mean "apply it to every branch".

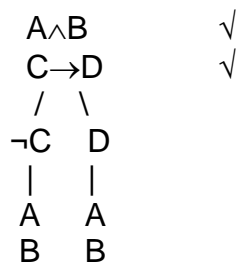
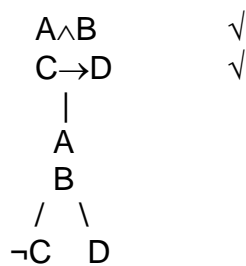
Example:



We don't apply the rule for ' $A \wedge B$ ' to the right hand branch, because the sentence ' $A \wedge B$ ' does not appear on that branch.

Hint: If faced with the option of applying a branching rule or a non-branching rule, choose the non-branching rule first!

Compare:



What do the branches represent?

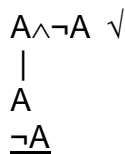
- Each branch represents a way for the starting sentence(s) to be true.
- When you've finished your tree, follow a branch downwards. When you get to the end of the branch, you know that one way for your starting sentence(s) to be true is for all the sentences you've come across on that branch to be true.

EXCEPT THAT:

- If the branch "closes", it does *not* represent a way for the starting sentence(s) to be true.

A closed branch is one that contains both a sentence and its negation.

We display a branch closure as follows:

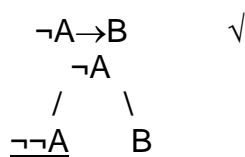
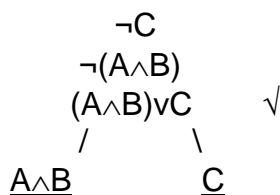


We do not write anything further on a branch once it has closed.

A branch that has not closed is called an open branch.

Closures occur whenever we have a sentence and its negation on the same branch; the sentence in question needn't be atomic.

Examples:



If every branch on a tree is closed

then there is no way for the set of sentences you started with all to be true,
i.e. there is no interpretation which makes them all true,
i.e. the set has no model.

If you have applied all the rules you can and a branch is still open

then there is a way for the set of sentences you started with all to be true,
i.e. there is an interpretation which makes them all true,
i.e. the set has a model.

It is often important to **show** that you have applied as many rules as you can, in order to show that there is a way for your starting sentences all to be true. That is why it is important to tick off used-up sentences. If you have ticked off every sentence that a rule can be applied to, you've shown that you have applied as many rules as you can.

FOUR MORE TREE RULES

THE RULE FOR NEGATED '∧'

$$\begin{array}{c} \neg(A \wedge B) \\ / \quad \backslash \\ \neg A \quad \neg B \end{array}$$

THE RULE FOR NEGATED '∨'

$$\begin{array}{c} \neg(A \vee B) \\ | \\ \neg A \\ \neg B \end{array}$$

THE RULE FOR NEGATED '→'

$$\begin{array}{c} \neg(A \rightarrow B) \\ | \\ A \\ \neg B \end{array}$$

THE RULE FOR NEGATED '↔'

$$\begin{array}{c} \neg(A \leftrightarrow B) \\ / \quad \backslash \\ A \quad \neg A \\ \neg B \quad B \end{array}$$