

305 Lecture 4.1 - Using Truth Trees

Brian Weatherson

Plan

Showing how we use truth trees to check for logical truth and validity.

Boxes and Diamonds, section 2.4.

1. $\top \neg A$
2. $\bot A \quad \neg\top, 1$

1. $\bot \neg A$
2. $\top A \quad \neg\bot, 1$

1. $\top A \wedge B$
2. $\top A \quad \wedge\top, 1$
3. $\top B \quad \wedge\top, 1$


1. $\bot A \wedge B$

2. $\bot A \quad \bot B \quad \wedge\bot, 1$

1. $\top A \vee B$

2. $\top A \quad \top B \quad \vee\top, 1$

1. $\bot A \vee B$
2. $\bot A \quad \vee\bot, 1$
3. $\bot B \quad \vee\bot, 1$

1. $\top A \rightarrow B$

2. $\bot A \quad \top B \quad \rightarrow\top, 1$

1. $\bot A \rightarrow B$
2. $\top A \quad \rightarrow\bot, 1$
3. $\bot B \quad \rightarrow\bot, 1$

Closure

- A tree without branches closes if there is a sentence on the tree that is marked as both true and false.
- Here is an example. The first line is stipulated, the rest are derived.

1.	$\top A \wedge \neg A$	
2.	$\top A$	$\wedge \top, 1$
3.	$\top \neg A$	$\wedge \top, 1$
4.	$\bot A$	$\neg \top, 3$
	\times	


Interpreting Closure

- If a tree is closed, it means the initial assumptions can't be true.
- So this tree means that the initial assumption $A \wedge \neg A$ can't be true.

1.	$\top A \wedge \neg A$	
2.	$\top A$	$\wedge \top, 1$
3.	$\top \neg A$	$\wedge \top, 1$
4.	$\bot A$	$\neg \top, 3$
	\times	

Closure with Branches

- A branching tree closes if every branch closes.
- The next slide has an example, with in this case the top two lines stipulated.

1. $\top \neg A \wedge \neg B$
2. $\top A \vee B$
3. $\top \neg A$ $\wedge \top, 1$
4. $\top \neg B$ $\wedge \top, 1$
5. $\text{F } A$ $\neg \top, 3$
6. $\text{F } B$ $\neg \top, 4$
- 
7. $\top A$ $\top B$ $\vee \top, 2$

x x

Closure with Branches

- A tree with any open branches is open, i.e., not closed.
- If a tree has some open branches and some closed branches, it is **open**.
- All that matters is if all branches are closed.
- The next slide is an example of an open tree.

1. $\top \neg A \wedge B$
 2. $\top A \vee B$
 3. $\top \neg A$ $\wedge \top, 1$
 4. $\top B$ $\wedge \top, 1$
 5. $\bot A$ $\neg \top, 3$
 6. $\top A \quad \top B$ $\vee \top, 2$
- x

Logical Truth

Here is the algorithm for seeing whether a sentence is a logical truth.

1. Start a tree by saying at line 1 that the sentence is **False**.

Logical Truth

Here is the algorithm for seeing whether a sentence is a logical truth.

1. Start a tree by saying at line 1 that the sentence is **False**.
2. If the tree closes, it is a logical truth.

Logical Truth

Here is the algorithm for seeing whether a sentence is a logical truth.

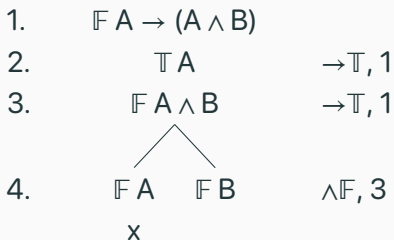
1. Start a tree by saying at line 1 that the sentence is **False**.
2. If the tree closes, it is a logical truth.
3. If the tree does not close, it is not a logical truth (of propositional logic).

Looking for Counterexamples (Tables)

- Truth tables didn't just tell us that something failed to be a logical truth (of propositional logic).
- It told us where the failure was.
- You didn't just know that there was an **F** in the relevant column, you knew which row it was on.
- And that told you where to look for counterexamples.

Looking for Counterexamples (Trees)

- The same thing happens with trees.
- By reading off the open branch, you can see where the sentence fails.
- The trick is to focus on the **atomic** sentences on the branch.
- These are the ones with no connectives at all.



- The right hand branch doesn't close.
- The atomics on that branch are that A is \mathbb{T} and B is \mathbb{F} .
- So that's the line on the truth table where $A \rightarrow (A \wedge B)$ is \mathbb{F} .

Validity

Here is the algorithm for seeing whether an argument is valid (in propositional logic).

1. Start a tree with one line for each premise, saying that the premise is **True**.

Validity

Here is the algorithm for seeing whether an argument is valid (in propositional logic).

1. Start a tree with one line for each premise, saying that the premise is **True**.
2. Then have a line that says the conclusion is **False**.

Validity

Here is the algorithm for seeing whether an argument is valid (in propositional logic).

1. Start a tree with one line for each premise, saying that the premise is **True**.
2. Then have a line that says the conclusion is **False**.
3. If the tree closes, the argument is valid.

Validity

Here is the algorithm for seeing whether an argument is valid (in propositional logic).

1. Start a tree with one line for each premise, saying that the premise is **True**.
2. Then have a line that says the conclusion is **False**.
3. If the tree closes, the argument is valid.
4. If the tree does not close, the argument is not valid (in propositional logic).

For Next Time

We will illustrate this algorithm.