PY1003 Introduction to Logic
Lecture 6: Trees for Sentential Logic III

Last time we looked at some more tree rules. We now have a complete set of nine rules for Sentential truth trees. These are the **only** legitimate rules for tree construction.

A point to note about branch closures

```
   A→B      √
   /  \
 ¬A    B
```

Does the left branch close because it has ¬A at the bottom and A further up (as part of A→B)?

No! In order for the branch to close, we would need A to appear on the branch *as a stand-alone atomic sentence*, rather than as a part of a compound sentence.

Which rule do I use?

Remember that in order to know which rule to apply for a sentence we must identify the main connective. When the main connective is '∧', 'v', '→' or '↔', we apply the rule for that connective.

But when the main connective is '¬', we must work a bit harder in order to know which rule to apply.

The rule to use depends on what *would* be the main connective if you took away the '¬' with the widest scope.

If the main connective would then be another '¬', use the rule for '¬¬'.
If the main connective would be '∧', use the rule for negated '∧'.
If the main connective would be 'v', use the rule for negated 'v'.
If the main connective would be '→', use the rule for negated '→'.
If the main connective would be '↔', use the rule for negated '↔'.

Notice that the rule for '∧' is non-branching but the rule for negated '∧' is branching.
Conversely, the rule for 'v' is branching but the rule for negated 'v' is non-branching.
Similarly, the rule for '→' is branching but the rule for negated '→' is non-branching.
But notice that the rule for '↔' and the rule for negated '↔' are both branching rules.

Proving whether or not a sentence is a contradiction

Each **open** branch on a tree represents a way for the starting sentence(s) to be true. A contradiction is a sentence which is false on every interpretation; there is **no** way for a contradiction to be true. So when you draw the tree for a contradiction, there will be no open branches. All the branches will eventually close.

```
¬(Av¬A)      √
   |
  ¬A
 ¬¬A
```

Every branch has closed, so ¬(Av¬A) is a contradiction.

```
(AvB)∧(¬A∧¬B)    √
      |
     AvB         √
   ¬A∧¬B         √
      |
     ¬A
     ¬B
    /   \
   A     B
```

Every branch has closed, so (AvB)∧(¬A∧¬B) is a contradiction.

A sentence which is not a contradiction (a 'consistent' sentence) is true on at least one interpretation. There **is** a way for a non-contradictory sentence to be true. So when you draw the tree for a non-contradictory sentence, at least one branch will remain open.

```
        (AvB)↔(B→A)       √
           /        \
 √    AvB          ¬(AvB)   √
 √   (B→A)         ¬(B→A)   √
    /      \           |
  ¬B        A         ¬A
 /  \      /  \       ¬B
A    B    A    B       |
                       B
                      ¬A
```

This tree has open branches and we have applied all the rules we can. So (AvB)↔(B→A) is not a contradiction.

Proving whether or not a sentence is a tautology

A sentence is a tautology if and only if its negation is a contradiction. So we can test whether or not a sentence is a tautology by testing whether or not its negation is a contradiction.

1. Is (A∧B)→A a tautology?

```
¬((A∧B)→A)      √
      |
    A∧B          √
     ¬A
      |
      A
      B
```

This tree has no open branches, so ¬((A∧B)→A) is a contradiction. So (A∧B)→A is a tautology.

2. Is A∧(Bv¬B) a tautology?

```
¬(A∧(Bv¬B))       √
   /       \
 ¬A       ¬(Bv¬B)  √
              |
             ¬B
            ¬¬B
```

This tree has an open branch, so ¬(A∧(Bv¬B)) is not a contradiction. So A∧(Bv¬B) is not a tautology.

Query:
If a tree has only open branches, does that mean the starting sentence was a tautology? Answer: NO! Consider the tree for the sentence A∧B:

```
A∧B   √
 |
 A
 B
```

This tree has only open branches, but A∧B is not a tautology.

All we know is that **if** there is a way for our starting sentence to be true, then there will be an open branch representing that way. But there may **also** be ways for the sentence to be false which are **not** represented on the tree by anything. And if there are ways for the sentence to be false it is not a tautology. We can't always tell from looking at a tree whether or not there are ways for the starting sentence to be false.

The **only** way to test whether a sentence is a tautology is to test whether its negation is a contradiction.

Proving whether or not a set of sentences is consistent

1. Is the set {A∧B, ¬A} consistent?

```
 A∧B   √
  ¬A
   |
   A
   B
```

Every branch is closed, so there is no way for all the sentences we started with to be true together. The set is inconsistent; it has no model.

2. Is the set {AvB, ¬A} consistent?

```
 AvB   √
  ¬A
 /   \
A     B
```

There is an open branch and we have applied all the rules we can. So there is a way for all the sentences we started with to be true together. The set is consistent; it has a model.

Exhibiting a model

When you've finished drawing a tree, if you have any open branches, follow one of them 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.

Pay particular attention to the atomic sentences and negated atomic sentences on the branch. We will be able to use them to 'exhibit a model' for our starting set.

Remember that a model for a set of sentences is an interpretation (assignment of truth-values to atomic sentences) which makes all the sentences in the set come out true.

```
 AvB   √
  ¬A
  /  \
A     B
```

Look at the right hand branch. It tells use that the interpretation which makes ¬A true and B true, i.e. makes A false and B true, will make every member of {AvB, ¬A} true.

We express this by saying that the following is a model for this set:
I(A)=F
I(B)=T

Sometimes a consistent set has more than one model.

```
 Av(Bv¬C)    √
   A→C       √
     A
   /   \
 ¬A     C
      /   \
     A   (Bv¬C)    √
          /    \
         B      ¬C
```

The tree has open branches, so the set is consistent. One model for this set is given by the rightmost open branch:
I(A)=T
I(B)=T
I(C)=T

But what about the branch on which A and C appear but neither B nor its negation does? The existence of this branch tells us that *any* interpretation where A and C are true is a model for the set. *It doesn't matter* what value B has. So another model is:
I(A)=T
I(B)=F
I(C)=T

We could have got this result using a truth table:

	A	B	C	Av(Bv¬C)	A→C	A
1	T	T	T	T	T	T
2	T	T	F	T	F	T
3	T	F	T	T	T	T
4	T	F	F	T	F	T
5	F	T	T	T	T	F
6	F	T	F	T	T	F
7	F	F	T	F	T	F
8	F	F	F	T	T	F

Lines 1 and 3 represent the two models for the set {Av(Bv¬C), A→C, A} described above.