

Symbolic Logic: Truth trees

David W. Agler

October 5, 2026

Why trees?

Truth trees are an alternative method for checking validity and other properties, e.g., consistency.

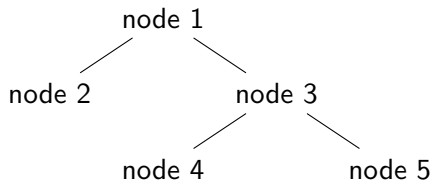
- ▶ They can be used on other logical languages.
- ▶ They are "less complex" (require less mechanical work).

What is a tree

Definition (Tree)

A tree is a collections of nodes and branches.

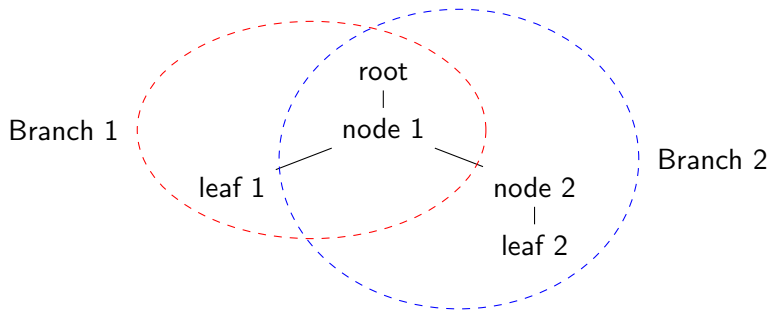
Here is a tree:



Branches

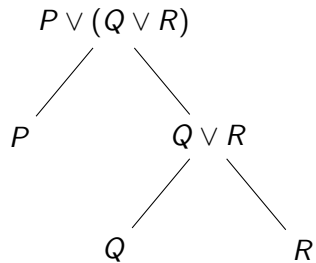
Definition (branch)

A branch in a tree consists of all of the nodes starting from a leaf node and moving upward to the root node.



Two branches in the tree. Leftmost branch consists of leaf 1, node 1, and the root node. Rightmost branch consists of leaf 2, node 2, node 1, and the root node.

Truth tree



Open and Closed Branches

Branches are either

1. Open
2. Closed

Open and Closed Branches

Definition (Closed Branch)

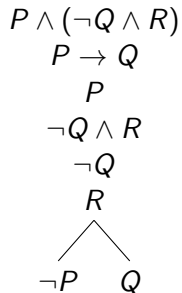
A branch is closed when it contains a wff ϕ and its literal negation $\neg(\phi)$. A closed branch is represented by writing $\otimes$ under the bottom node in the closed branch.

1. $P \wedge Q$
2. $\neg P \vee Q$
3. P
4. Q
5. $\neg P \quad Q$
6. $\otimes$

Open Branches

Definition (Open Branch)

An open branch is a branch that is not closed. That is, a branch that does not contain a wff ϕ and its literal negation $\neg(\phi)$.



decomposition

Definition (decomposition)

A **decomposition** of a wff ϕ is a representation of what wff or wffs are true if ϕ is true.

Example of AND- decomposition

Example: “Tek is evil and Tek is rich.”

1. Tek is evil.
2. Tek is rich.

Example of PL AND-decomposition

$E \wedge R$

E

R

By writing E and R directly under $E \wedge R$, we are expressing the decomposition that if $E \wedge R$ is true, then

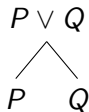
1. E is true *and*
2. R is true.

Example of OR-decomposition

Example: “Tek is evil or Tek is rich.”

1. Tek is evil, OR
2. Tek is rich.

Example of PL OR-decomposition



By branching E and R under $E \vee R$, we are expressing the decomposition that if $E \vee R$ is true, then

1. E is true *or*
2. R is true.

Decomposition rule idea

1. If ϕ is true just in one case, then *stack*
2. If ϕ is true in three cases, then *branch*
3. If ϕ is true in two cases, then *stack and branch*

What about the conditional?

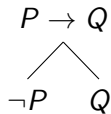
Notice that $P \rightarrow Q$ is true in any of the following three cases:

1. P is false
2. Q is true
3. P is false and Q is true

In other words, $P \rightarrow Q$ is true iff P is F **or** Q is T.

Example of PL Conditional-decomposition

$P \rightarrow Q$ is true iff P is F **or** Q is T.



Descending decomposition rule

Definition (Descending decomposition rule)

Whenever a wff ϕ is decomposed, it should be decomposed under every leaf node in every open branch that *descends* from ϕ .

Let's illustrate this with some examples.

1. $(P \wedge Q) \vee (R \wedge S)$
2. $(P \vee Q) \wedge (R \vee S)$

Decomposition rules

- ▶ Reasoning what wff(s) are T when ϕ is T and whether you should stack, branch, or stack-branch is exhausting.
- ▶ Simpler approach is to identify a set of rules and apply them to wff types

Conjunction decomposition

$$\begin{array}{l} \phi \wedge \psi \\ \phi \quad \wedge D \\ \psi \quad \wedge D \end{array}$$

Conjunction decomposition illustrated

$$\neg A \wedge \neg B$$

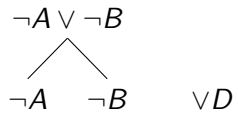
$$\neg A \quad \wedge D$$

$$\neg B \quad \wedge D$$

Disjunction decomposition

$$\begin{array}{c} \phi \vee \psi \\ \wedge \\ \phi \quad \psi \end{array} \quad \vee D$$

Disjunction decomposition illustrated



How many rules?

There are nine decomposition rules:

1. conjunctions: $\phi \wedge \psi$
2. disjunctions: $\phi \vee \psi$
3. conditionals: $\phi \rightarrow \psi$
4. biconditionals: $\phi \leftrightarrow \psi$
5. negated conjunctions: $\neg(\phi \wedge \psi)$
6. negated disjunctions: $\neg(\phi \vee \psi)$
7. negated conditionals: $\neg(\phi \rightarrow \psi)$
8. negated biconditionals: $\neg(\phi \leftrightarrow \psi)$
9. double-negations: $\neg(\neg(\phi))$

How many rules?

- ▶ The back of your book (pp.358-359) shows all of these rules.
- ▶ Be familiar with these pages.
- ▶ This will be provided for you on the exam.

Conjunction-D and Negation-Conjunction-D

$$\phi \wedge \psi$$

$$\phi$$

$$\wedge D$$

$$\psi$$

$$\wedge D$$

$$\neg(\phi \wedge \psi)$$

$$\neg(\phi) \quad \neg(\psi)$$

$$\neg \wedge D$$

Notice that conjunction decomposition is a stacking rule, negated conjunction decomposition is a branching.

Disjunction-D and Negated-Disjunction-D

$$\begin{array}{c} \phi \vee \psi \\ \swarrow \searrow \\ \phi \quad \psi \end{array} \quad \vee D$$

$$\begin{array}{c} \neg(\phi \vee \psi) \\ \neg(\phi) \quad \neg \vee D \\ \neg(\psi) \quad \neg \vee D \end{array}$$

While the disjunction decomposition rule branches, the negated disjunction decomposition is a stacking rule.

Exercise

Book, Ex. 4.42, 1-5

Truth tree columns

To make trees clearer, we will add a few things:

1. Leftmost column: for numbering the wffs,
2. Middle column: for nodes and branches,
3. Rightmost column: for justification of wffs

Example

$$1. \quad (R \vee P) \wedge (C \wedge D) \quad P$$

$$1. \quad (R \vee P) \wedge (C \wedge D) \quad P$$

$$2. \quad R \vee P \quad 1 \wedge D$$

$$3. \quad C \wedge D \quad 1 \wedge D$$

$$1. \quad (R \vee P) \wedge (C \wedge D) \quad P$$

$$2. \quad R \vee P \quad 1 \wedge D$$

$$3. \quad C \wedge D \quad 1 \wedge D$$

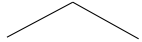
$$4. \quad \begin{array}{cc} & \wedge \\ R & P \end{array} \quad 2 \vee D$$

Fully decomposed branch

Definition (Fully decomposed branch)

A PL-branch is fully decomposed when every non-literal wff in the branch has had a decomposition rule applied to it. In other words, when every wff that can be decomposed has been decomposed.

Fully decomposed?

1.	$(P \rightarrow S) \rightarrow Q \checkmark$	P
		
2.	$\neg(P \rightarrow S) \checkmark$	$1 \rightarrow D$
3.	P	$2 \neg \rightarrow D$
4.	$\neg S$	$2 \neg \rightarrow D$

Fully decomposed?

1.	$(R \vee P) \wedge (C \wedge D)$	P
2.	$R \vee P$	$1 \wedge D$
3.	$C \wedge D$	$1 \wedge D$
		
4.	$R \quad P$	$2 \vee D$

It was not. It is now.

1.	$(R \vee P) \wedge (C \wedge D)$	P
2.	$R \vee P$	$1 \wedge D$
3.	$C \wedge D$	$1 \wedge D$
		
4.	R P	$2 \vee D$
5.	$\boxed{C}$ $\boxed{C}$	$3 \wedge D$
6.	$\boxed{D}$ $\boxed{D}$	$3 \wedge D$

- ▶ Notice that we applied the descending decomposition rule
- ▶ $C \wedge D$ needs to be decomposed under every open node that descends from $C \wedge D$

Tree decomposition

Exercise 4.43, 1.

Trees for sets of wffs

- ▶ So far, we've decomposed single wffs.
- ▶ Now let's decompose sets of wffs.

Trees for sets of wffs: process

The process is straightforward. Suppose you have three wffs: ϕ, ψ, χ .

- ▶ Stack the wffs, number them, justify with P
- ▶ Then decompose the tree

Trees for sets of wffs: Example

$\{P \wedge Q, A \vee B\},$

1. $P \wedge Q$ P

2. $A \vee B$ P

Now decompose them.

Tree decomposition

Exercise 4.44, 2.

Completed open branch

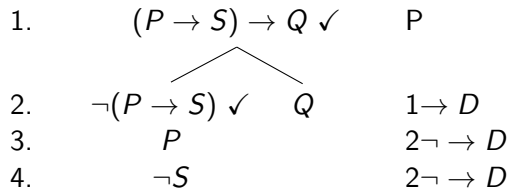
To use trees for testing various properties, e.g., validity, we will need the property of a branch being a **completed open branch**.

Definition (Completed open branch)

A completed open branch (COB) is a fully decomposed branch that is not closed. That is, it is a fully-decomposed branch that does not contain a wff ϕ and its literal negation $\neg(\phi)$.

COB: Example 1

Does this tree have a completed open branch?



COB: Example 2

Does this tree have a completed open branch?

1.	$P \wedge R \checkmark$	P
2.	$M \wedge \neg P \checkmark$	P
3.	P	$1 \wedge D$
4.	R	$1 \wedge D$
5.	M	$2 \wedge D$
6.	$\neg P$	$2 \wedge D$
	$\otimes$	
	3,6	

Open and closed trees

The definition of a completed open branch is used to define a *completed open tree*.

Definition (Completed Open Tree)

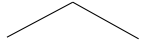
A tree is a completed open tree iff it has at least one completed open branch. That is, a tree is a completed open tree iff it contains at least one fully decomposed branch that is not closed. A completed open tree is a tree where there is at least one branch that has an **O** under it.

This contrasts with a closed tree.

Definition (Closed Tree)

A tree is closed when all of the tree's branches are closed. A closed tree will have an $\oplus$ under every branch.

Open or closed?

1.	$(P \rightarrow S) \rightarrow Q \checkmark$	P
		
2.	$\neg(P \rightarrow S) \checkmark \quad Q$	$1 \rightarrow D$
3.	P	$2 \neg \rightarrow D$
4.	$\neg S$	$2 \neg \rightarrow D$

Open or closed?

1.	$P \wedge R \checkmark$	P
2.	$M \wedge \neg P \checkmark$	P
3.	P	$1 \wedge D$
4.	R	$1 \wedge D$
5.	M	$2 \wedge D$
6.	$\neg P$	$2 \wedge D$
	$\otimes$	
	3,6	

Open or closed?

Generally, you have a wff ϕ or set of wffs Γ you want to test. The process is:

1. Set up the tree
2. Decompose the tree
3. Determine if it is a completed open tree or a closed tree

Exercise

Exercise 4.45, 4

Summary

At this point:

1. Know how to decompose individual wffs: $\phi \wedge \psi$ decomposed using $\wedge D$
2. Know how to construct a tree through decomposition (and use associated rules), e.g., descending decomposition rule
3. Know key terminology: open branch, closed branch, completed open branch, completed open tree, closed tree

1. Use only the decomposition rules you need to determine if the tree is a completed open tree or a closed tree.
2. Use rules that close branches.
3. Use stacking rules before branching rules.
4. Maybe: decompose more complex wffs before simpler wffs

- | | | |
|----|---|--------------|
| 1. | $((A \vee B) \wedge (R \vee \neg S)) \wedge (P \wedge \neg P) \checkmark$ | P |
| 2. | $(A \vee B) \wedge (R \vee \neg S)$ | $1 \wedge D$ |
| 3. | $P \wedge \neg P$ | $1 \wedge D$ |

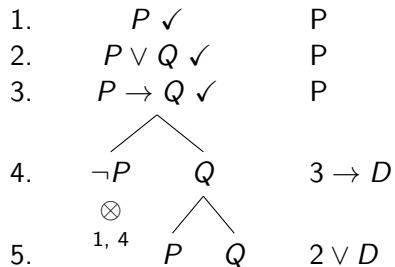
Makes sense to decompose line (3). It will close the main branch.

1.	$((A \vee B) \wedge (R \vee \neg S)) \wedge (P \wedge \neg P) \checkmark$	P
2.	$(A \vee B) \wedge (R \vee \neg S) \checkmark$	$1 \wedge D$
3.	$P \wedge \neg P$	$1 \wedge D$
4.	P	$3 \wedge D$
5.	$\neg P$	$3 \wedge D$

- ▶ Tree is closed.
- ▶ Further decomposition will not yield an open tree.
- ▶ Tree is done.

1.	$P \checkmark$	P
2.	$P \vee Q \checkmark$	P
3.	$P \rightarrow Q$	P
	$\swarrow \quad \searrow$	
4.	P	Q
	$\swarrow \quad \searrow \quad \swarrow \quad \searrow$	
5.	$\neg P \quad Q$	$\neg P \quad Q$
	$3 \rightarrow D$	

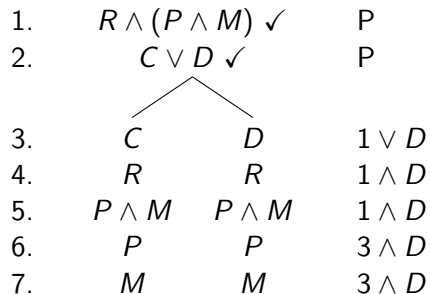
In this example, line (2) was decomposed instead of line (3).



Same tree but decomposing (3) first yields a simpler tree.

Branch before you stack?

In this tree, we branch before we stack. Notice the two long branches (12 nodes).



Stack before you branch

In this tree, we stack before we branch (there are 8 nodes).

1.	$R \wedge (P \wedge M) \checkmark$	P
2.	$C \vee D \checkmark$	P
3.	R	$1 \wedge D$
4.	$P \wedge M$	$1 \wedge D$
5.	P	$3 \wedge D$
6.	M	$3 \wedge D$
		
7.	$C \quad D$	$2 \vee D$

Exercise

Exercise 4.46, 3, 4.

Analysis

Truth trees have three steps:

1. Setup
2. Decomposition
3. Analysis

Part of the Analysis involves understanding what a *completed open branch* tells us.

COB and interpretation

A *completed open branch* tells us that there is an interpretation that makes all the wffs in the branch true.

Recovering an interpretation

1.	$P \wedge R \checkmark$	P
2.	$\neg M \wedge P \checkmark$	P
3.	P	$1 \wedge D$
4.	R	$1 \wedge D$
5.	$\neg M$	$2 \wedge D$
6.	P	$2 \wedge D$

There is a completed open branch. We can get an interpretation that makes all the wffs in the branch true.

Recovering an interpretation: method

To recover an interpretation from a completed open branch (COB):

1. Write down each literal in the COB
2. If it is a letter, e.g., P , assign it a value of T to the letter, e.g. $\mathcal{I}(P) = T$
3. If it is a negated letter, e.g., $\neg P$, assign it a value of F to the letter, e.g., $\mathcal{I}(P) = F$

Recovering an interpretation: method

1. $P \wedge R \checkmark$ P
2. $\neg M \wedge P \checkmark$ P
3. P $1 \wedge D$
4. R $1 \wedge D$
5. $\neg M$ $2 \wedge D$
6. P $2 \wedge D$

1. $P, R, \neg M, P$

2. Interpretation: $\mathcal{I}(P) = T, \mathcal{I}(R) = T, \mathcal{I}(M) = F$

P	R	M	$P \wedge R$	$\neg M \wedge P$
T	T	F	T	T

Recovering an interpretation - another example

1.	$(P \wedge R) \wedge (\neg M \vee \neg P) \checkmark$	P
2.	$P \wedge R \checkmark$	$1 \wedge D$
3.	$\neg M \vee \neg P \checkmark$	$1 \wedge D$
4.	P	$2 \wedge D$
5.	R	$2 \wedge D$
6.	$\neg M$ $\neg P$ $\otimes$ 4,6	$3 \vee D$

Exercise

It is extremely likely you will need to know how to recover an interpretation on the exam. Let's practice.

Exercise 4.47, 6

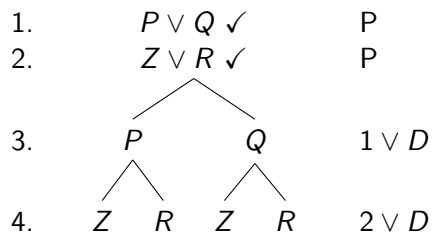
Consistency

Definition (Truth-Tree Test for Consistency)

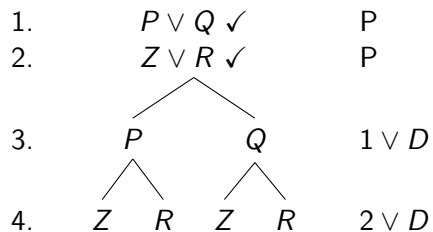
A truth tree shows that a set of wffs $\{\phi_1, \phi_2, \dots, \phi_n\}$ is PL-consistent if there is at least one completed open branch for a tree whose root wffs are: $\phi_1, \phi_2, \dots, \phi_n$. If such a tree is closed, then $\{\phi_1, \phi_2, \dots, \phi_n\}$ is inconsistent.

1. Stack the wffs
2. Create the tree via decomposition
3. If COB, then consistent
4. If Closed Tree, then inconsistent

Is it consistent?



Is it consistent?



1. Yes.
2. You can recover an interpretation from a COB.

Is it consistent?

1.	$\neg(S \rightarrow T) \checkmark$	P
2.	$\neg\neg(\neg S \vee T) \checkmark$	P
3.	$\neg S \vee T \checkmark$	$2 \neg\neg D$
4.	S	$1 \neg \rightarrow D$
5.	$\neg T$	$1 \neg \rightarrow D$
		
6.	$\neg S$	$3 \vee D$
		
	4,6	
	T	
		
	5,6	

Is it consistent?

1.	$\neg(S \rightarrow T) \checkmark$	P
2.	$\neg\neg(\neg S \vee T) \checkmark$	P
3.	$\neg S \vee T \checkmark$	$2 \neg\neg D$
4.	S	$1 \neg \rightarrow D$
5.	$\neg T$	$1 \neg \rightarrow D$
		
6.	$\neg S \quad T$	$3 \vee D$
	 	
	4,6 5,6	

1. No. All branches are closed.
2. There is no interpretation that makes the wffs true.

Exercise

Ex. 4.48, 5, 9

Tautology, Contradiction, Contingency

Definition (Truth-Tree Test for Contradiction)

A truth tree shows that ϕ is a PL-contradiction if a tree of the stack of ϕ determines a closed tree.

Definition (Truth-Tree Test for Tautology)

A truth tree shows that ϕ is a PL-tautology if a tree of the stack of $\neg(\phi)$ determines a closed tree.

Definition (Truth-Tree Test for Contingency)

A truth tree shows that ϕ is a contingency if a tree of ϕ does not determine a closed tree and a tree of $\neg(\phi)$ does not determine a closed tree.

Tautology, Contradiction, Contingency

Suppose you have $P \rightarrow Q$.

1. If you test for contradiction, your tree is $P \rightarrow Q$.
 - ▶ If closes: contradiction.
 - ▶ If not: not a contradiction
2. If you test for tautology, your tree is $\neg(P \rightarrow Q)$.
 - ▶ If closes: tautology.
 - ▶ If not: not a tautology

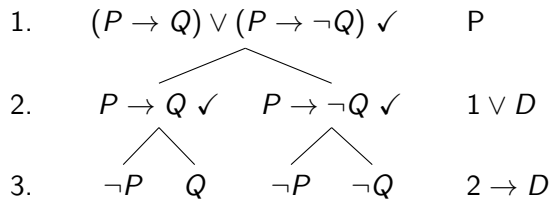
Contingency is determined by a process of elimination: not a contradiction and not a tautology.

Example

Let's check what $P \vee \neg Q$ is:

$$\begin{array}{lcl} 1. & P \vee \neg Q \checkmark & P \\ & \swarrow \quad \searrow & \\ 2. & P \quad \neg Q & 1 \vee D \end{array}$$

$$\begin{array}{lcl} 1. & \neg(P \vee \neg Q) \checkmark & P \\ 2. & \neg P & 1 \neg \vee D \\ 3. & Q & 1 \neg \vee D \end{array}$$



$(P \rightarrow Q) \vee (P \rightarrow \neg Q)$ is not a contradiction.

Since it is not a contradiction, test whether it is a tautology by negating it and decomposing it.

1.	$\neg((P \rightarrow Q) \vee (P \rightarrow \neg Q)) \checkmark$	P
2.	$\neg(P \rightarrow Q) \checkmark$	$1 \neg \vee D$
3.	$\neg(P \rightarrow \neg Q) \checkmark$	$1 \neg \vee D$
4.	P	$2 \neg \rightarrow D$
5.	$\neg Q$	$2 \neg \rightarrow D$
6.	P	$3 \neg \rightarrow D$
7.	$\neg \neg Q$	$3 \neg \rightarrow D$
	$\otimes$	
	5,7	

Notice that it closes, so $(P \rightarrow Q) \vee (P \rightarrow \neg Q)$ is a tautology

Exercise

Ex. 4.49, 7, pp.172

Equivalence

1. Skip equivalence.
2. You won't have to do an equivalence tree test on the exam.
3. You can also skip Ex. 4.50 in the homework.

Semantic entailment

Definition (Truth-Tree Test for Semantic Entailment)

A truth tree shows that $\Gamma \models \phi$ if a tree consisting of the members of Γ stacked with the literal negation of ϕ determines a closed tree. Otherwise, the tree shows that $\Gamma \not\models \phi$.

Simple example

Consider the following argument:

1. P1: If Tek is happy, then he's smiling.
2. P2: Tek is happy.
3. C: Therefore, he is smiling.

Translation: $H \rightarrow S, H \models S$

Simple example

1. Take this $H \rightarrow S, H \models S$
2. And setup the tree by listing the premises and the **negation** of the conclusion
3. So this: $H \rightarrow S, H, \neg S$

Simple example

1. Take this $H \rightarrow S, H \models S$
2. And setup the tree by listing the premises and the **negation** of the conclusion
3. So this: $H \rightarrow S, H, \neg S$

- | | | |
|----|-------------------|---|
| 1. | $H \rightarrow S$ | P |
| 2. | H | P |
| 3. | $\neg(S)$ | P |

Now decompose

Simple example

1.	$H \rightarrow S$	P
2.	H	P
3.	$\neg(S)$	P
		
4.	$\neg H \quad S$	$2 \rightarrow D$
	$\otimes \quad \otimes$	
	2,4 3,4	

Since all branches close, the argument is valid. It is the case that: $H \rightarrow S, H \models S$

What to remember

1. Take the argument $\Gamma \models \phi$ and
2. Setup the tree by stacking the premises Γ and the negation of the conclusion ϕ
3. Decompose
4. If the tree closes, then $\Gamma \models \phi$.
5. If it is a completed open tree, then $\Gamma \not\models \phi$

Invalid

If the tree shows $\Gamma \not\models \phi$, then you can recover an interpretation that shows:

- ▶ $\Gamma \not\models \phi$
- ▶ In other words: how the premises can be T and conclusion F.
- ▶ In other words: how the argument is invalid
- ▶ In other words: how the conclusion does not necessarily follow from the premises

Example

Here is a simple example from Ex. 4.51, pp.177, #2: $A \vee B \models B$

Exercise

Let's try a more difficult one: Ex. 4.51, pp.177, #8: $P \rightarrow Q, P \wedge W \models Q \vee R$

Exercise

On the exam, you might have something more difficult, so let's try something much harder:

$$A \wedge (B \wedge C), \neg(\neg A \vee D), A \leftrightarrow B \models D \wedge C$$