Informatics 1

Lecture 6 Satisfiability

Michael Fourman

Ordering

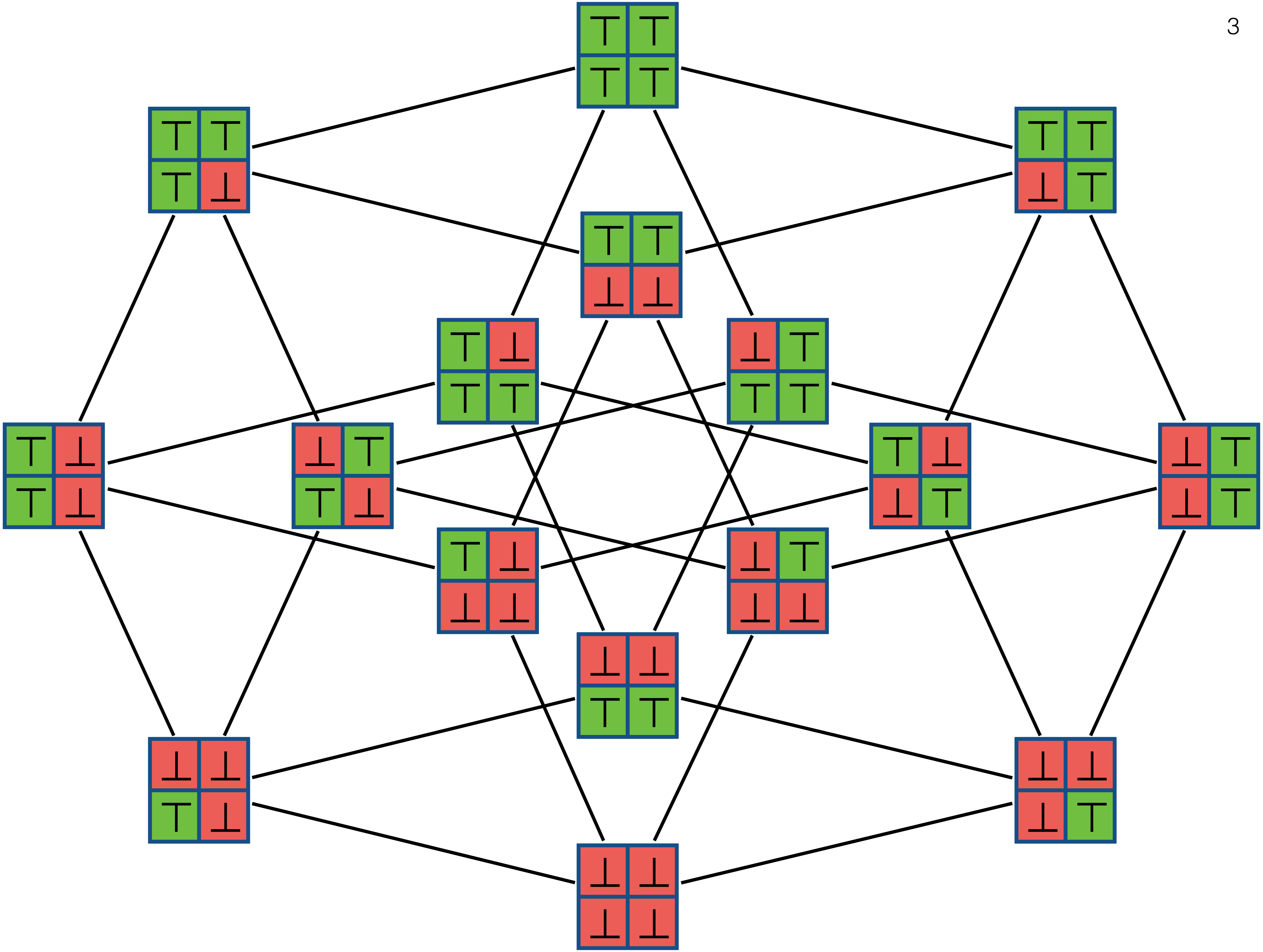
$A \to B$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\top$

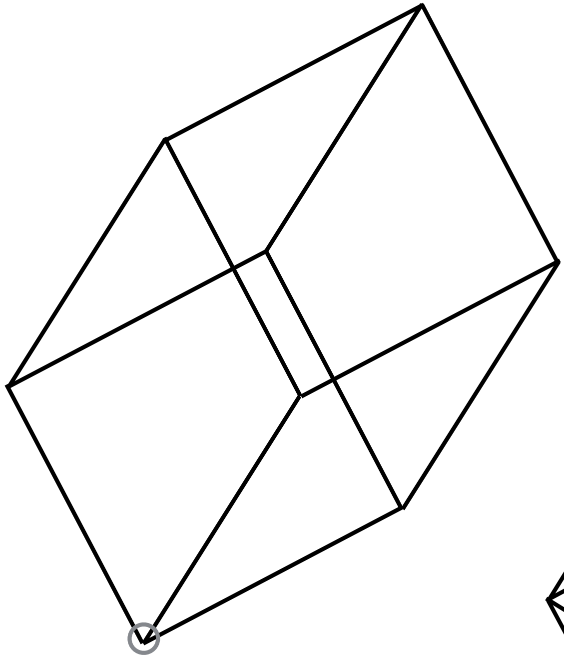
for 0-1 truth values,
$A \to B = \top$ iff
$A \leq B$

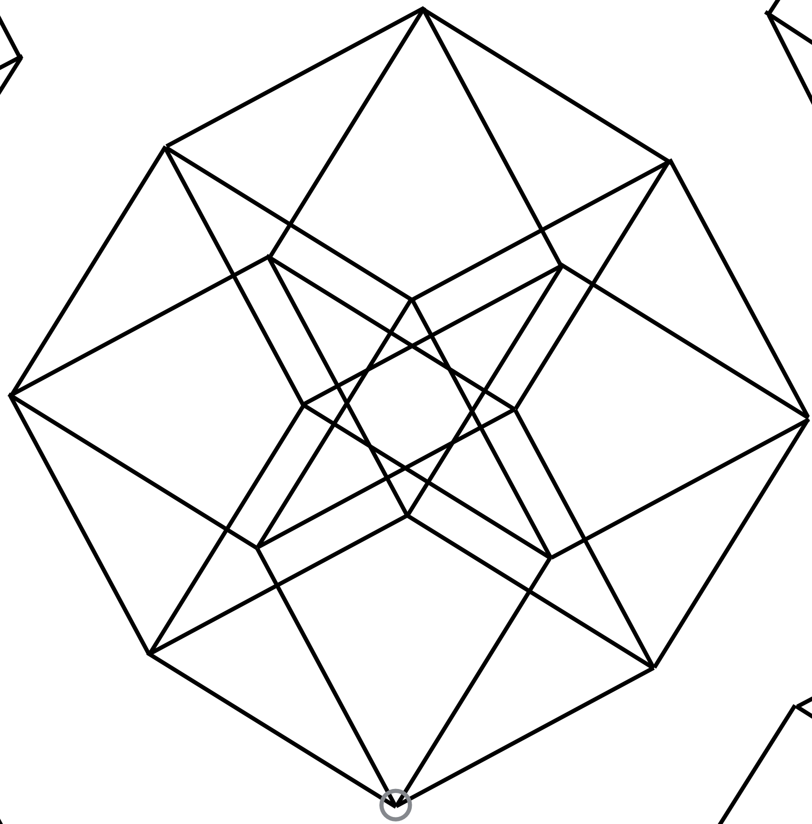
if $A \to B = \top$ then
$\{ x \mid A \} \subseteq \{ x \mid B \}$

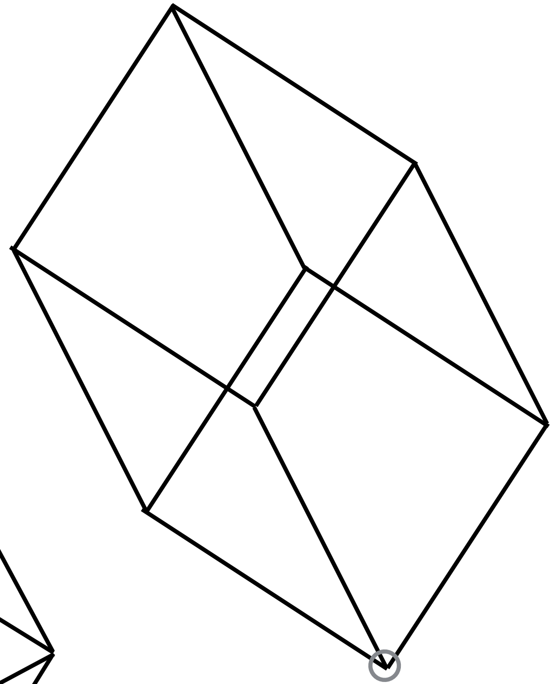
In any Boolean algebra, we define

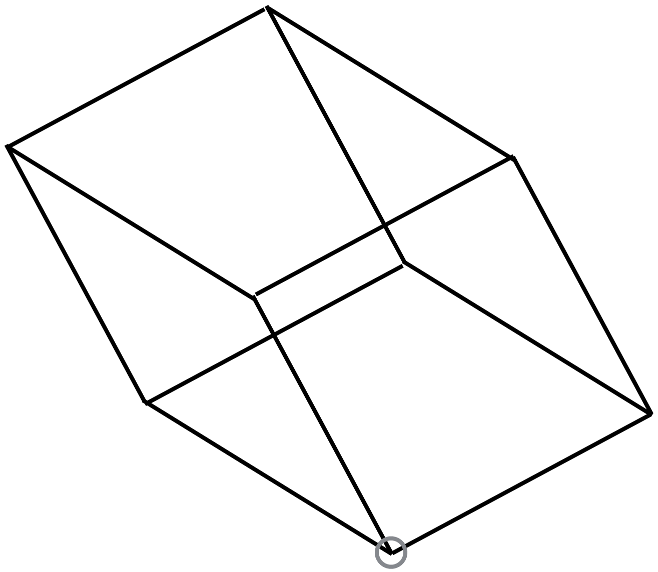
$$x \leq y \;\equiv\; x \wedge y = x \;\equiv\; x \vee y = y \;\equiv\; x \to y = \top$$

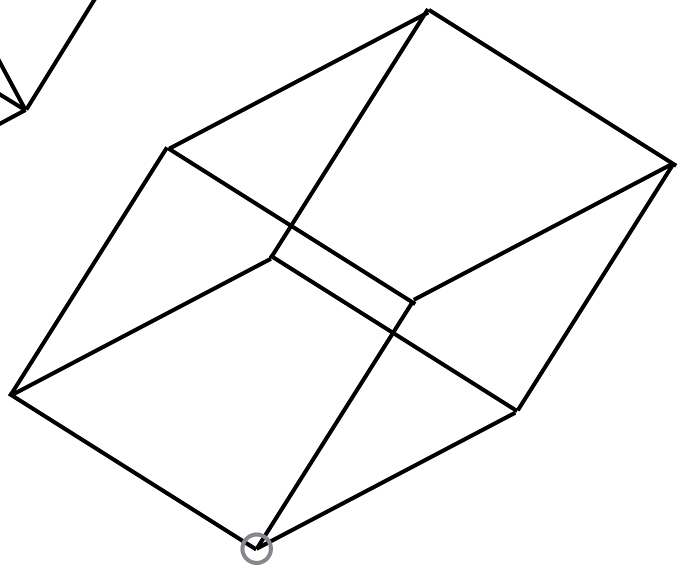


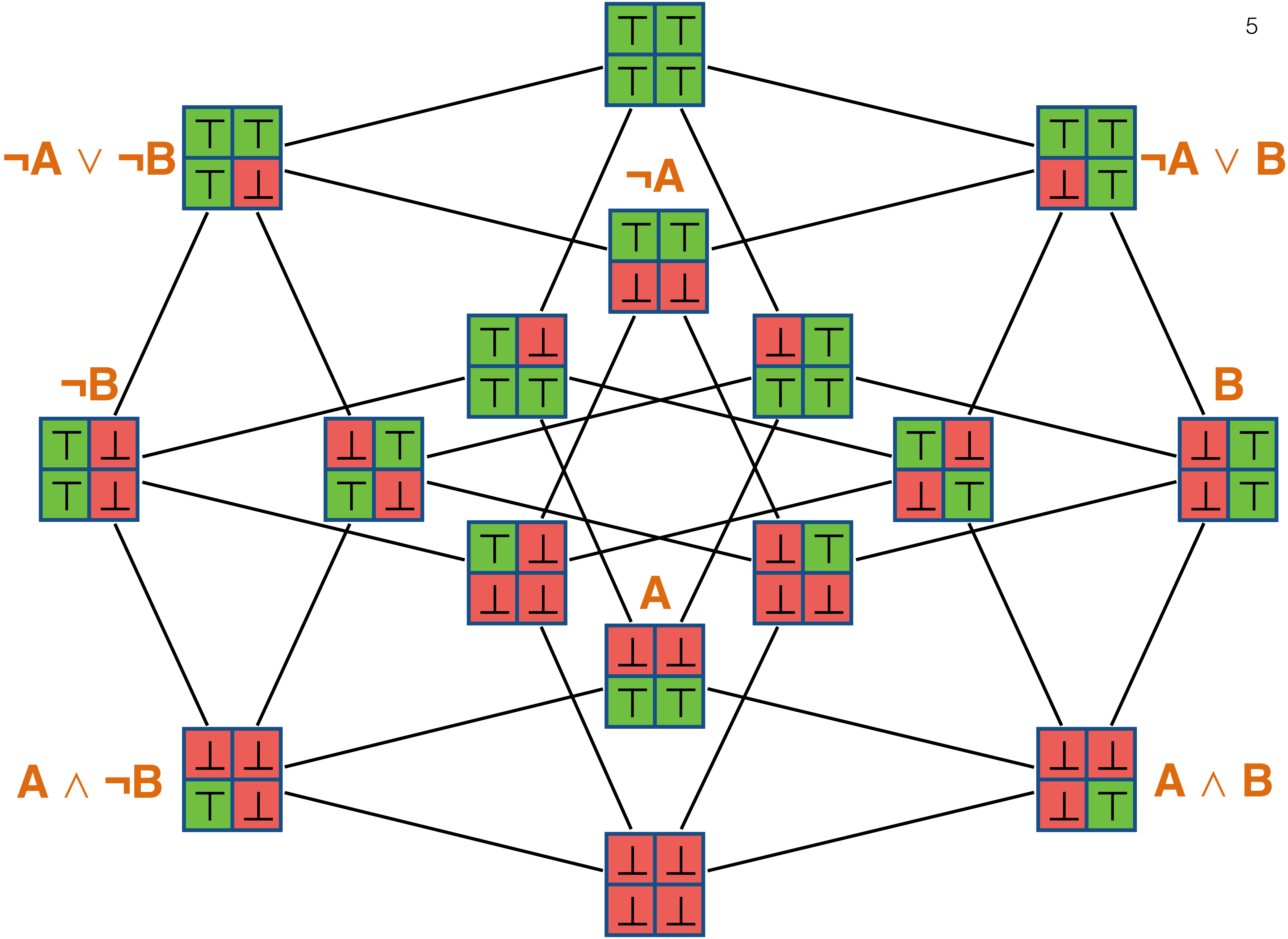




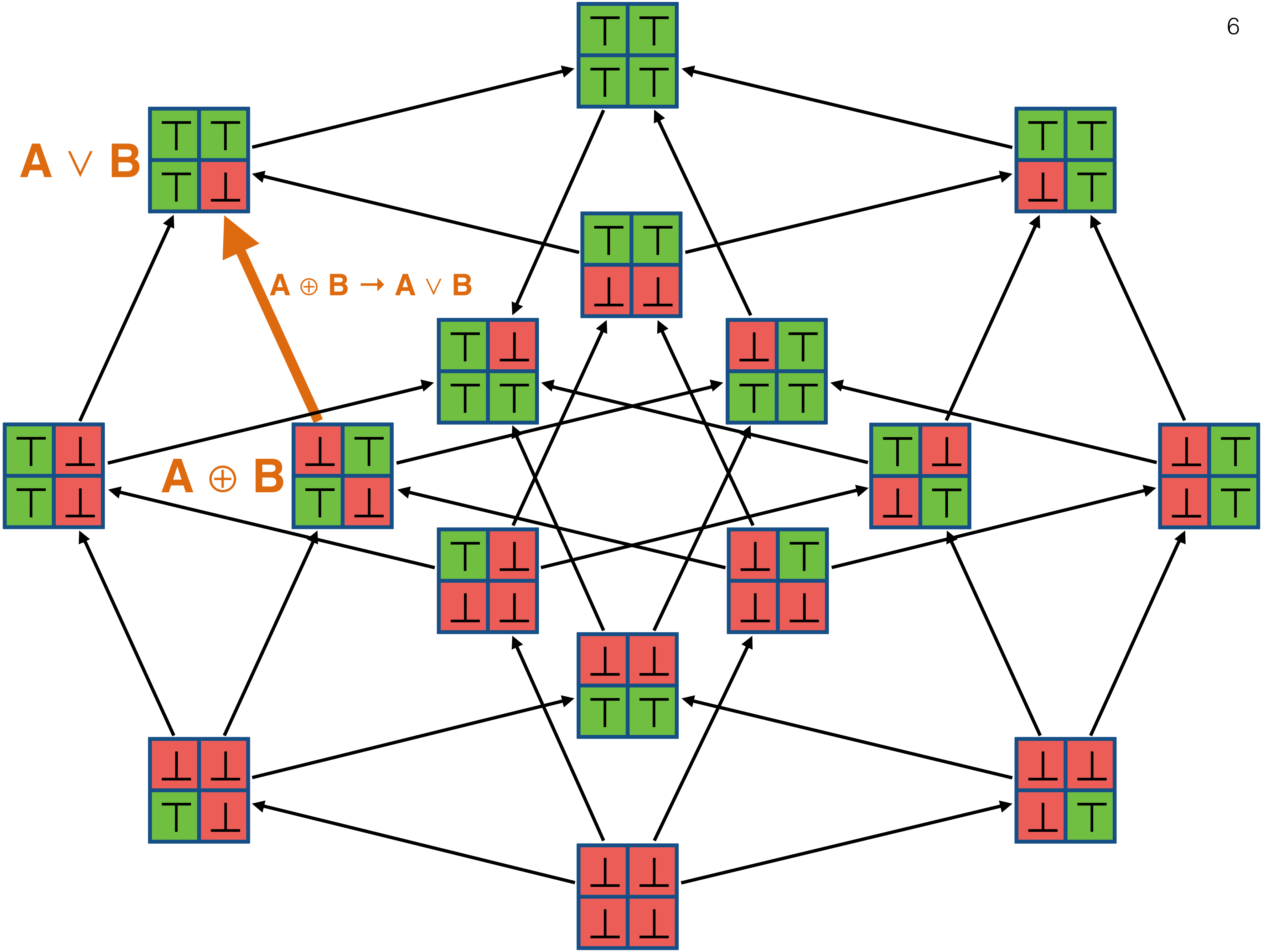


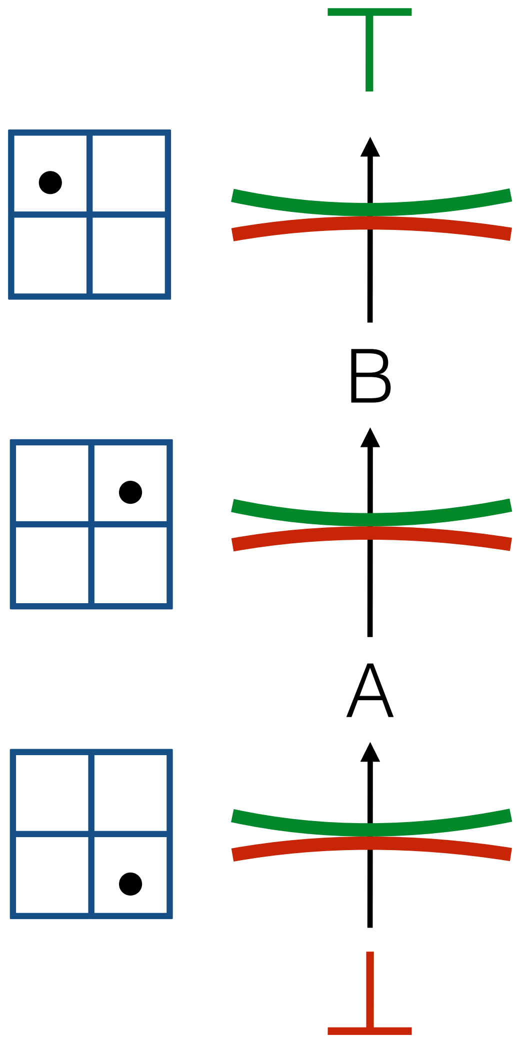






¬A ∨ ¬B
¬A ∨ B
¬A
¬B
B
A
A ∧ ¬B
A ∧ B


A ∨ B
A ⊕ B → A ∨ B
A ⊕ B



Propositions are ordered

by $x \leq y$ iff $x \rightarrow y = \top$

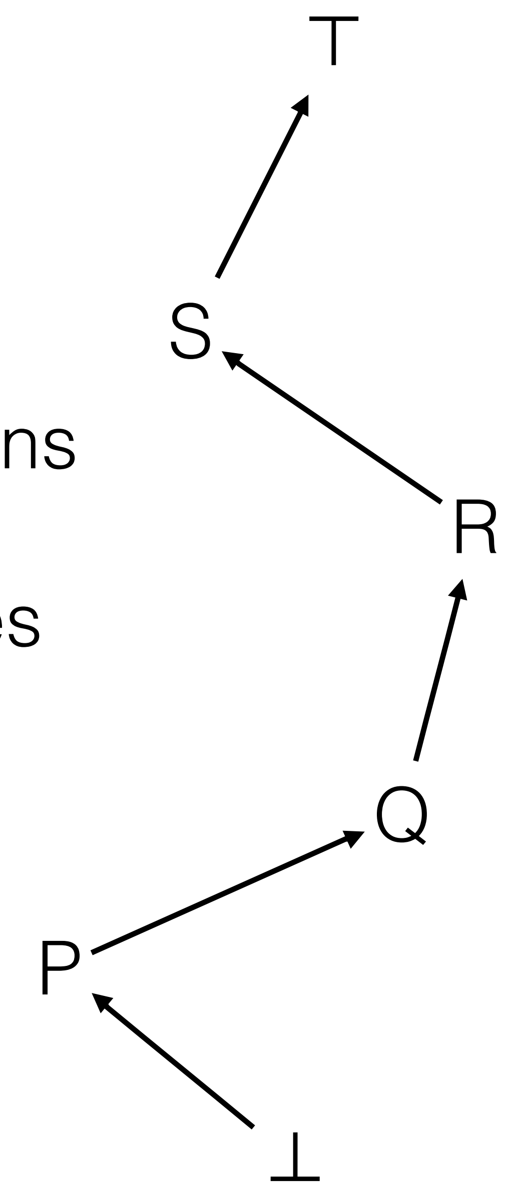
Any valid truth assignment

must draw a line

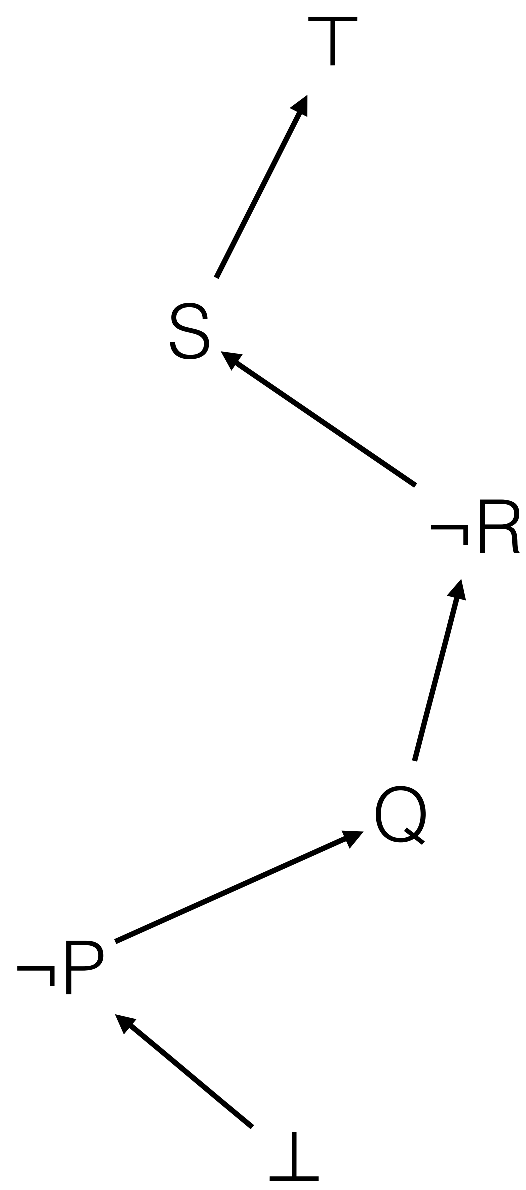
between $\bot$ and $\top$

We make a variable true when it falls above the line and false when if falls below.

If we have a chain of n-1 implications
between n variables
we can draw the line in n+1 places
making any number, from 0 to n,
of these variables true.



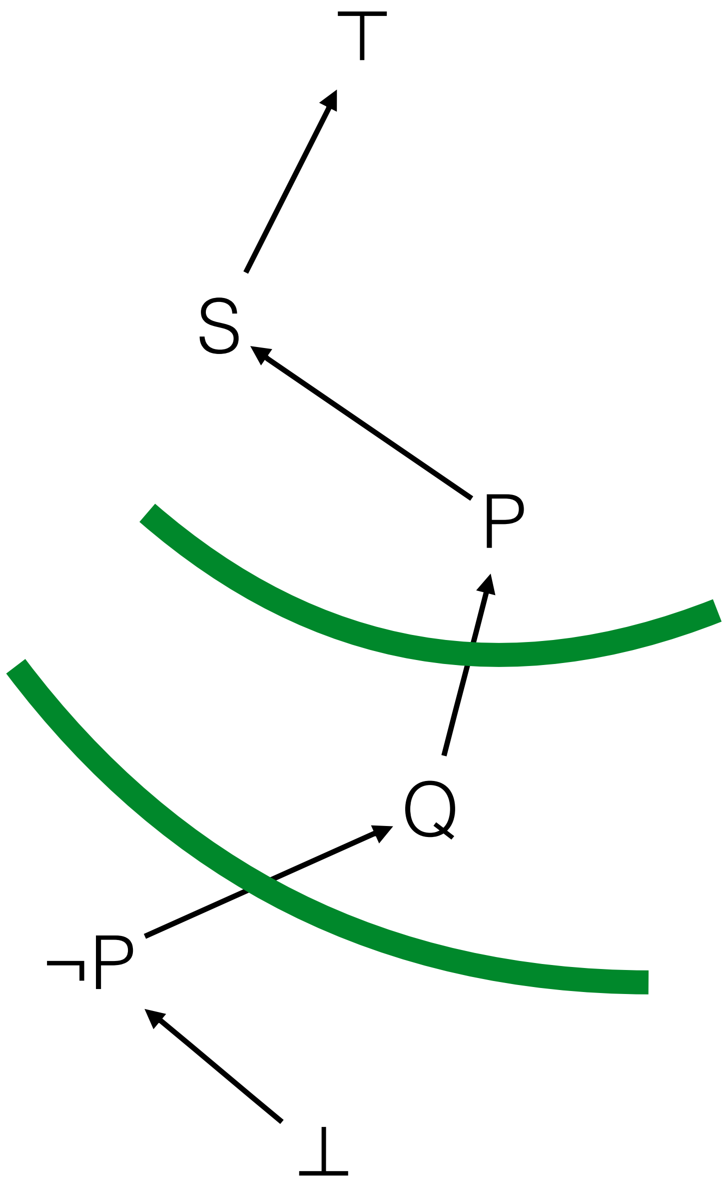
If some of the variables are negated we can do the same (but making the negated variables false when they fall above the line and true when they fall below)



If a variable appears together with its negation, we have to draw the line between them.

Here, P must be true.

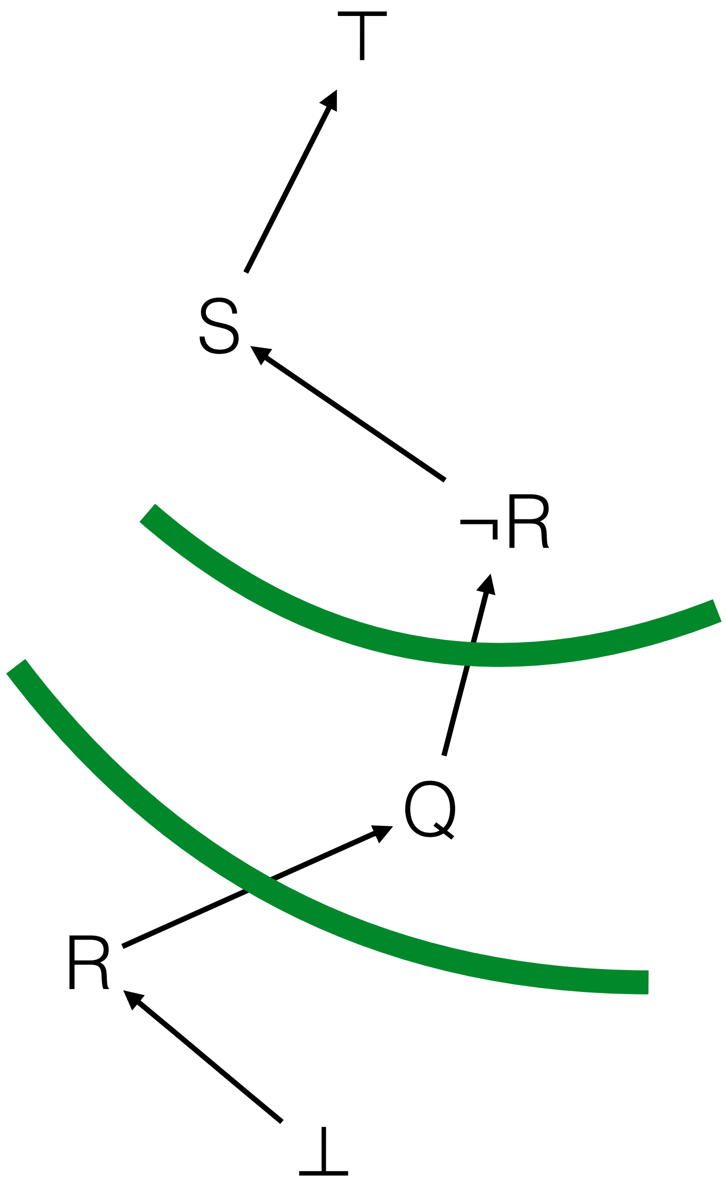
$(\neg P \rightarrow P) \rightarrow P$
is a tautology



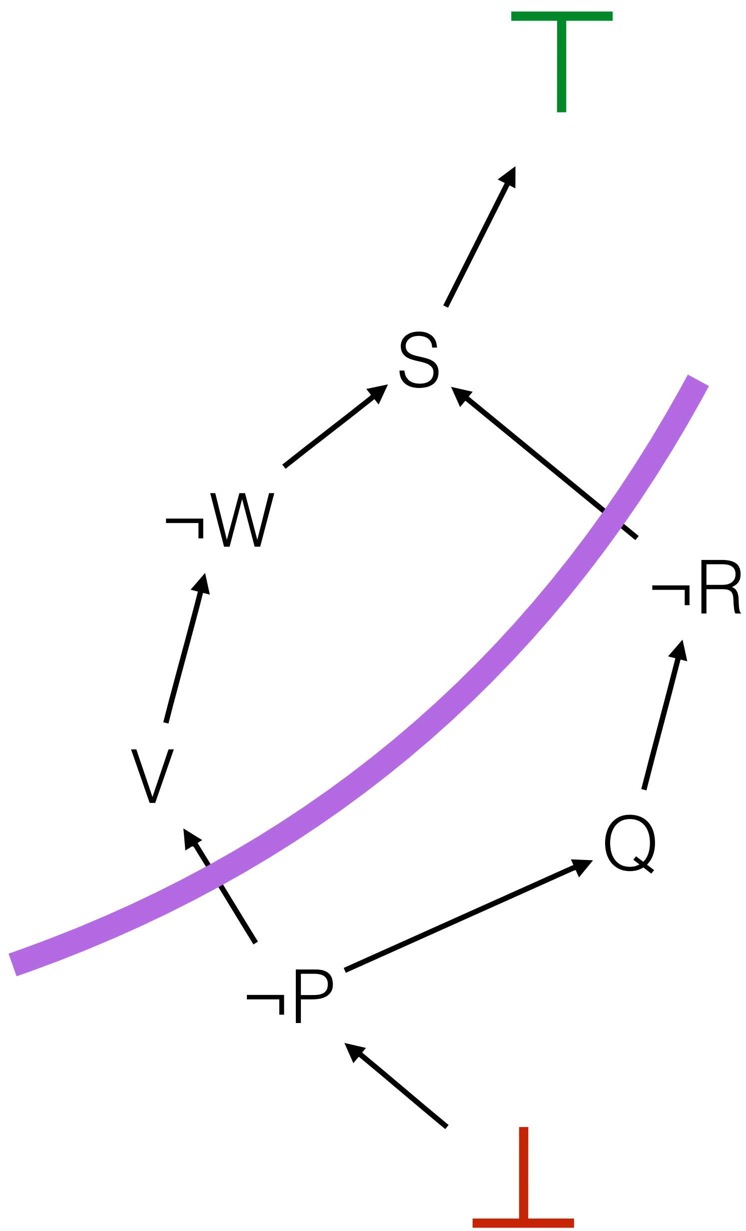
If a variable appears together with its negation, we have to draw the line between them.

Here, R must be false.

$(R \to \neg R) \to \neg R$
is a tautology



The same trick works if our implications form a partial order.
But we have more options since we can draw a wavy line.

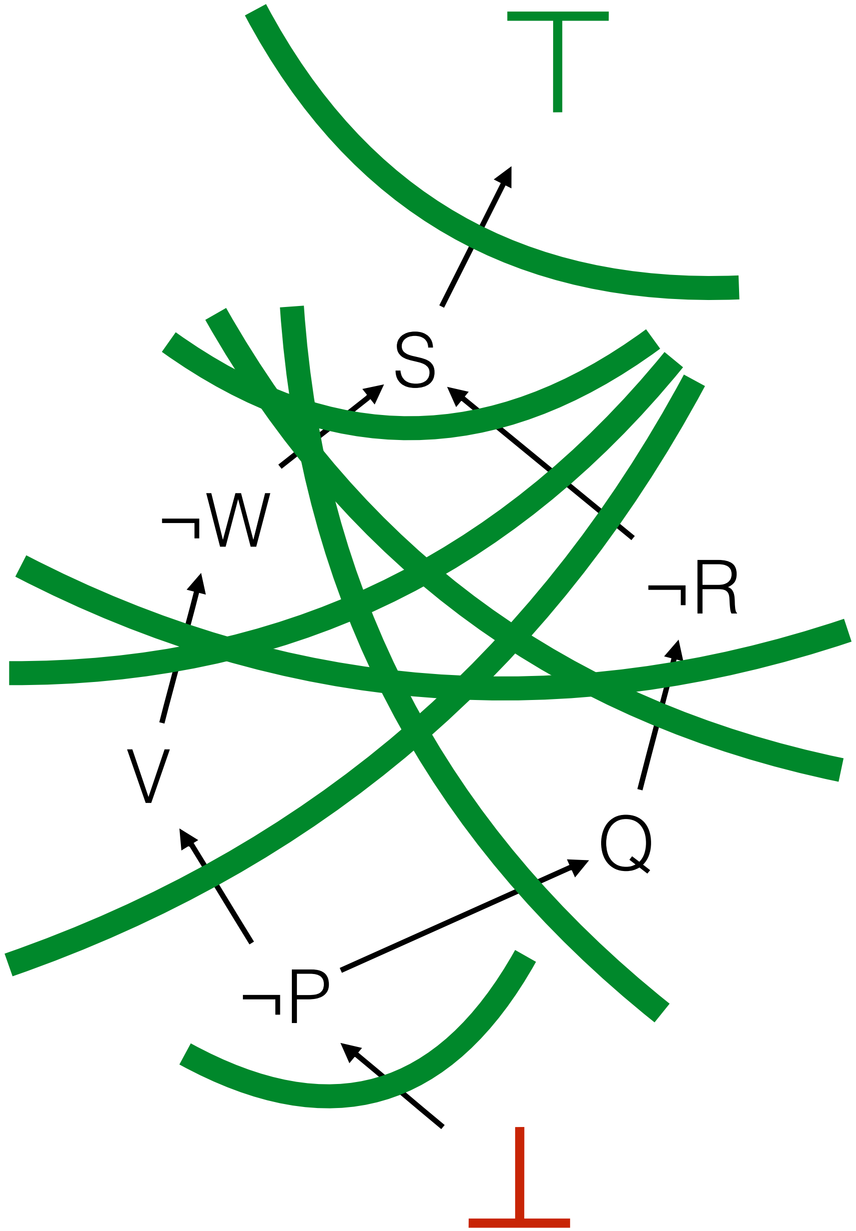


The **arrow rule** says that, whenever our line cuts an arrow, then the head must be on the side of true and the tail on the side of false.

The same trick works if our implications form a partial order.
But we have more options since we can draw a wavy line.

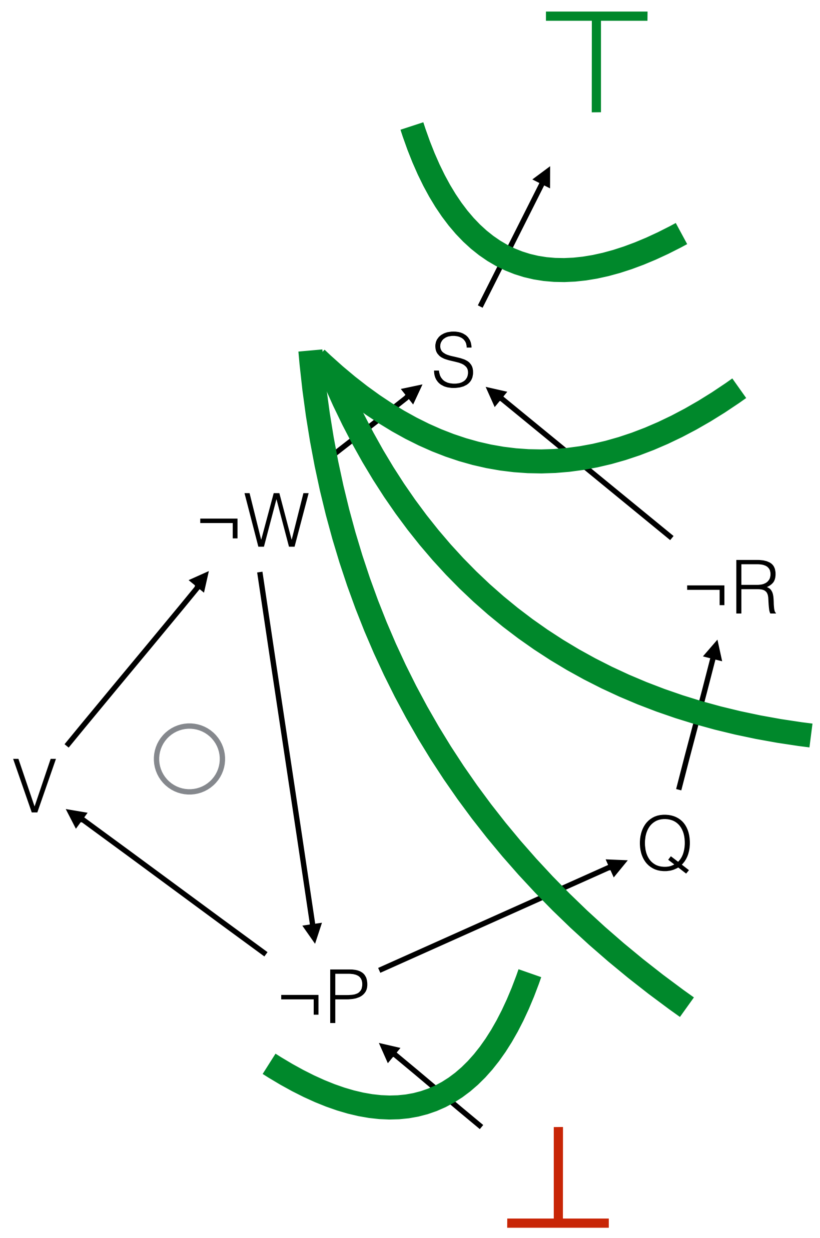
Not all of the valid truth assignments are represented in this diagram.

How many are missing?



If our constraints include cycles (loops), then our lines must not cut them.

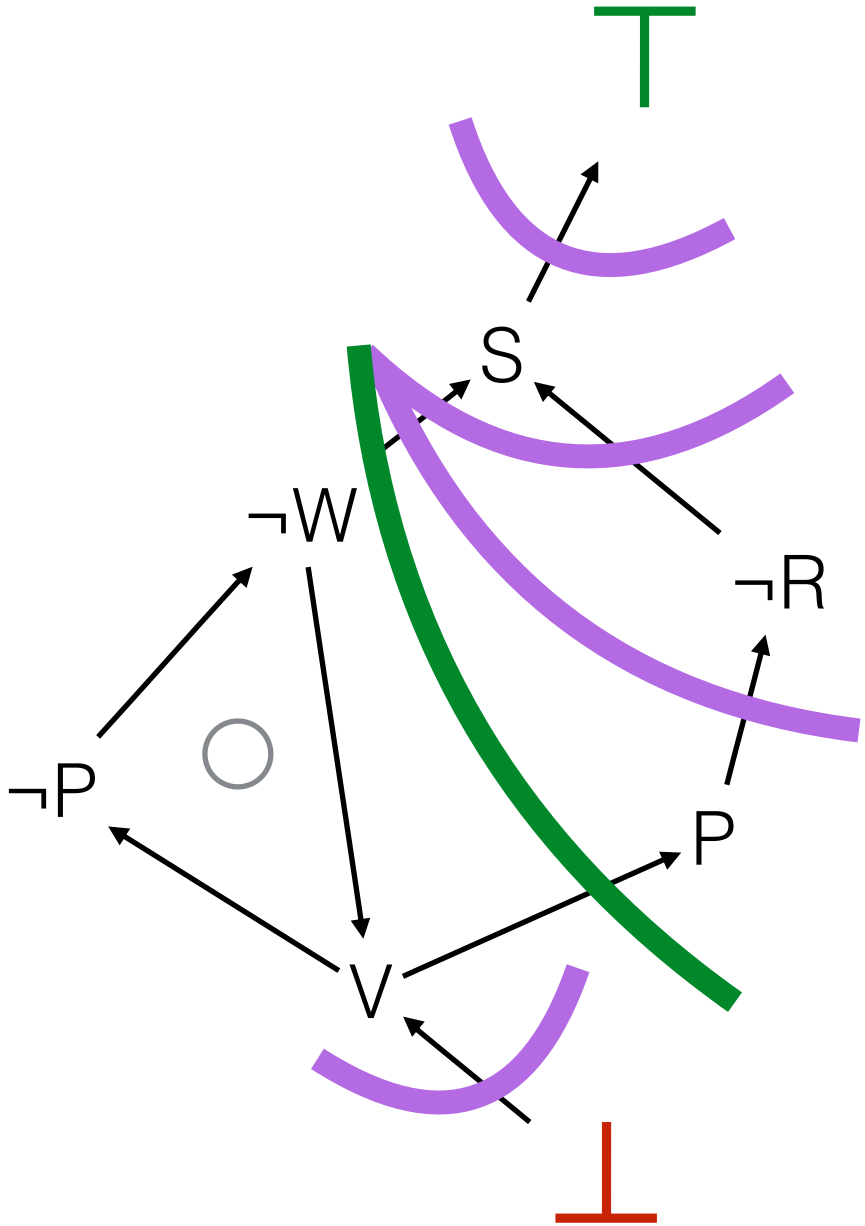
(because of the arrow rule)



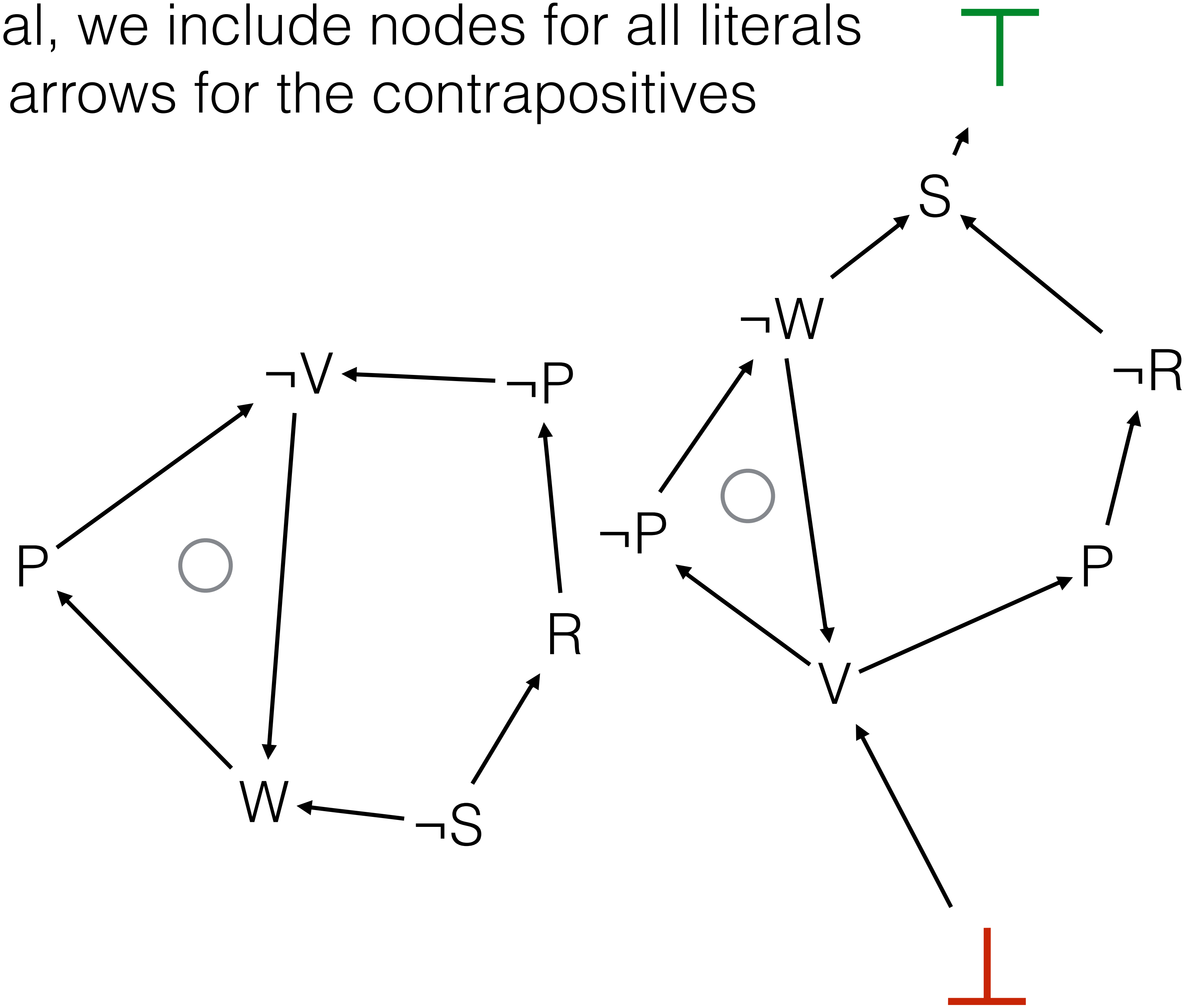
Here is a different example.

We have to draw the line between true and false, and between a variable and its negation

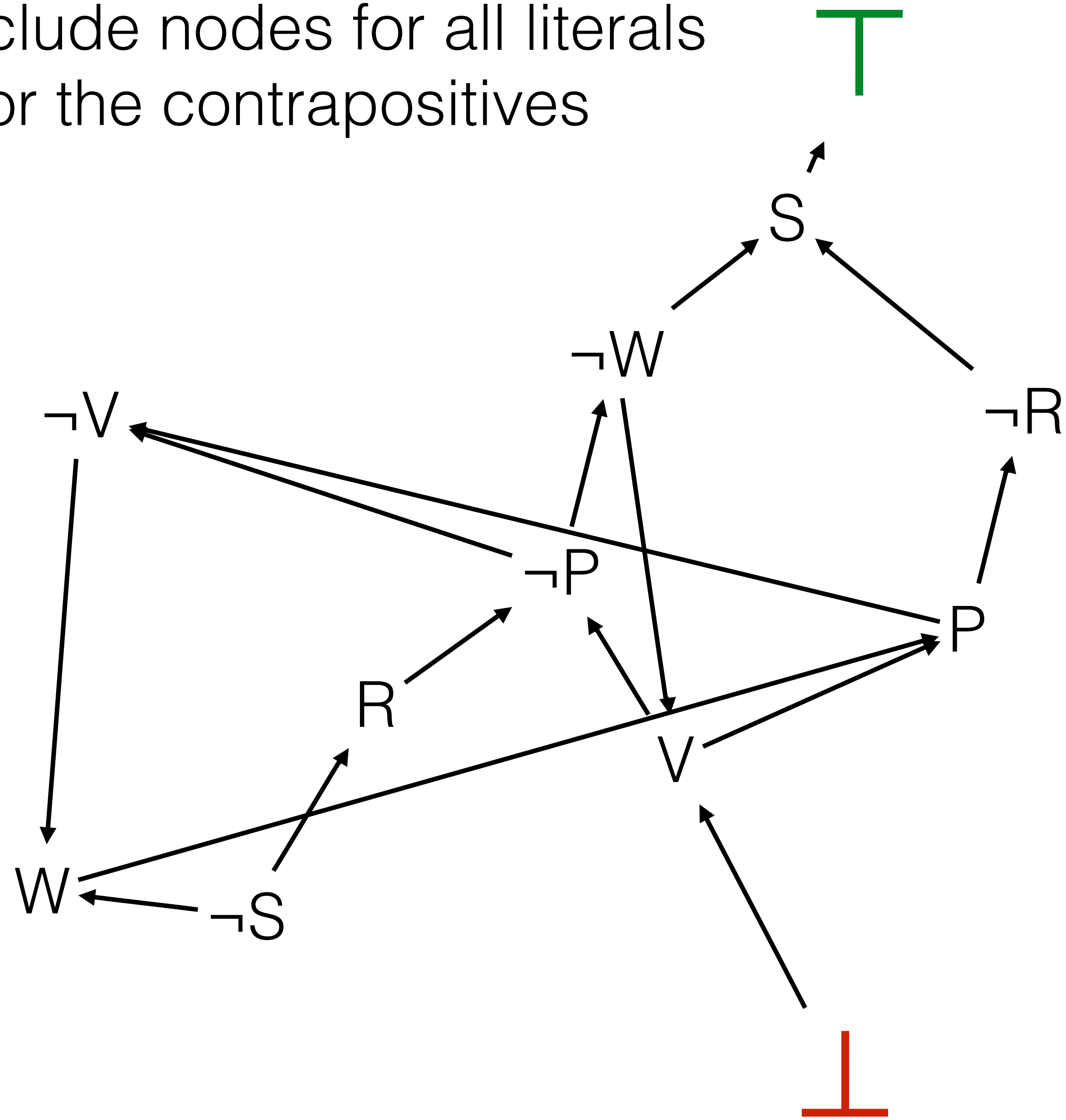
They cannot both fall on the same side of the line



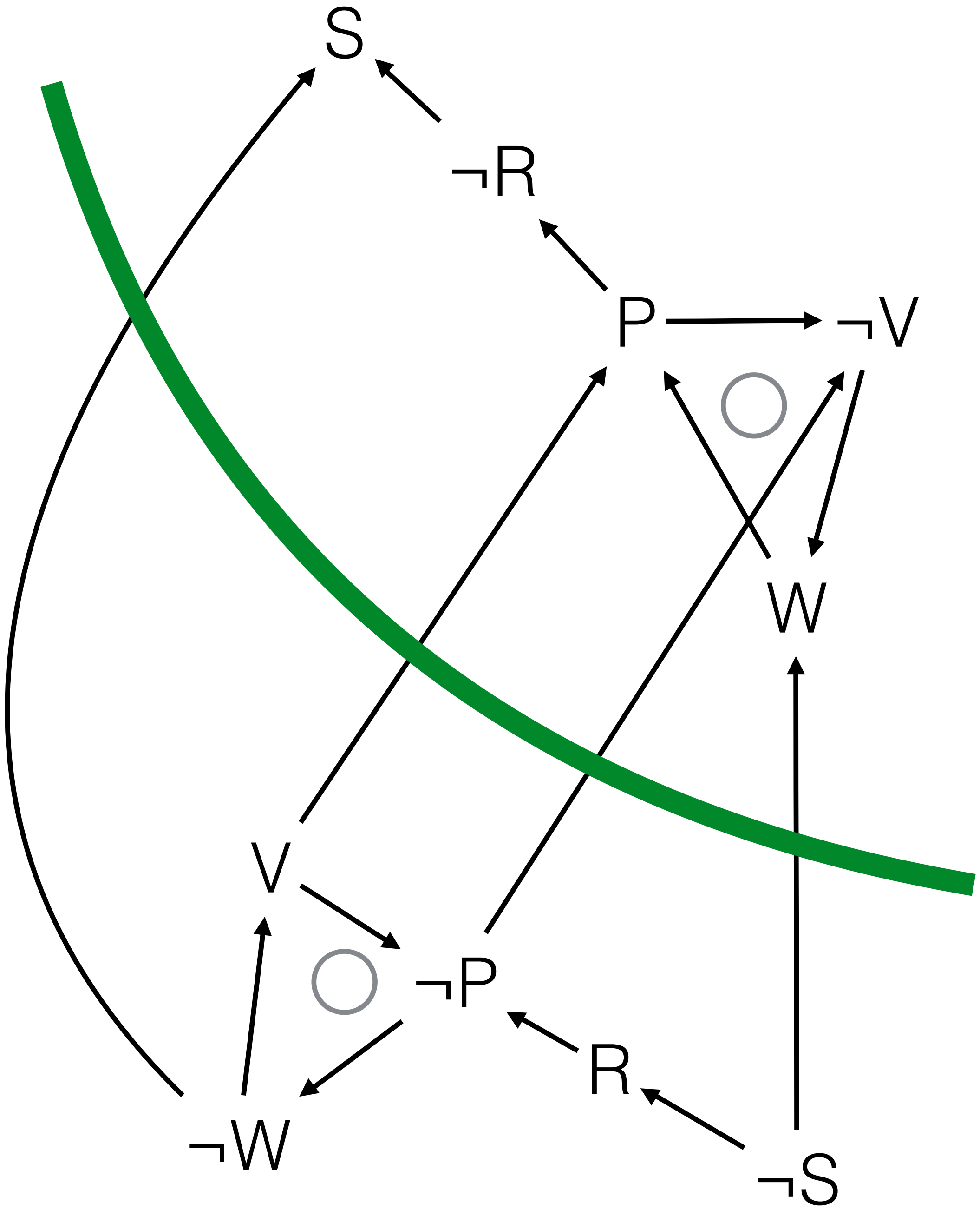
In general, we include nodes for all literals and arrows for the contrapositives



In general, we include nodes for all literals and arrows for the contrapositives



By dragging things around, we get



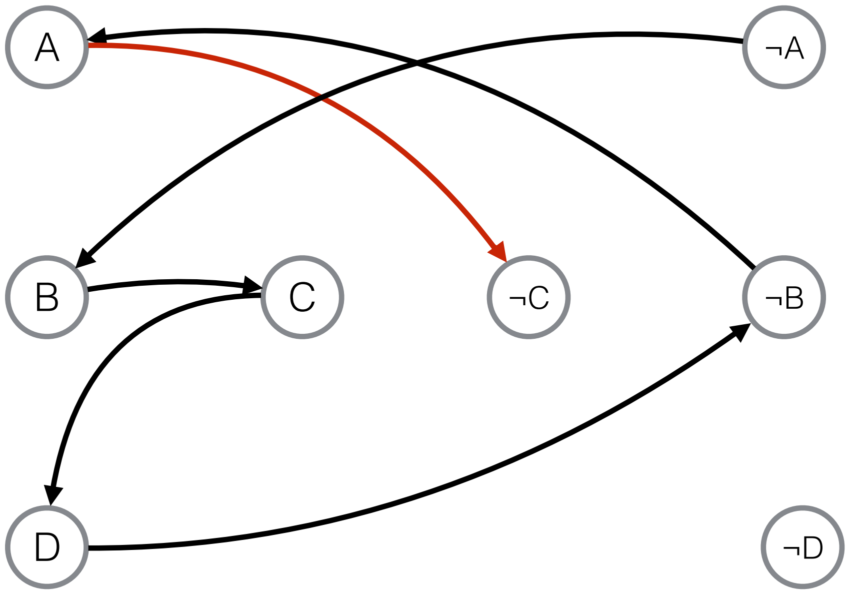
Binary constraints

You may not take both Archeology and Chemistry
If you take Biology you must take Chemistry
You must take Biology or Archeology
If you take Chemistry you must take Divinity
You may not take both Divinity and Biology

$$(\neg A \vee \neg C) \wedge (\neg B \vee C) \wedge (B \vee A) \wedge (\neg C \vee D) \wedge (\neg D \vee \neg B)$$

$$(A \rightarrow \neg C) \wedge (B \rightarrow C) \wedge (\neg B \rightarrow A) \wedge (C \rightarrow D) \wedge (D \rightarrow \neg B)$$

$(\neg A \vee \neg C) \wedge (\neg B \vee C) \wedge (B \vee A) \wedge (\neg C \vee D) \wedge (\neg D \vee \neg B)$

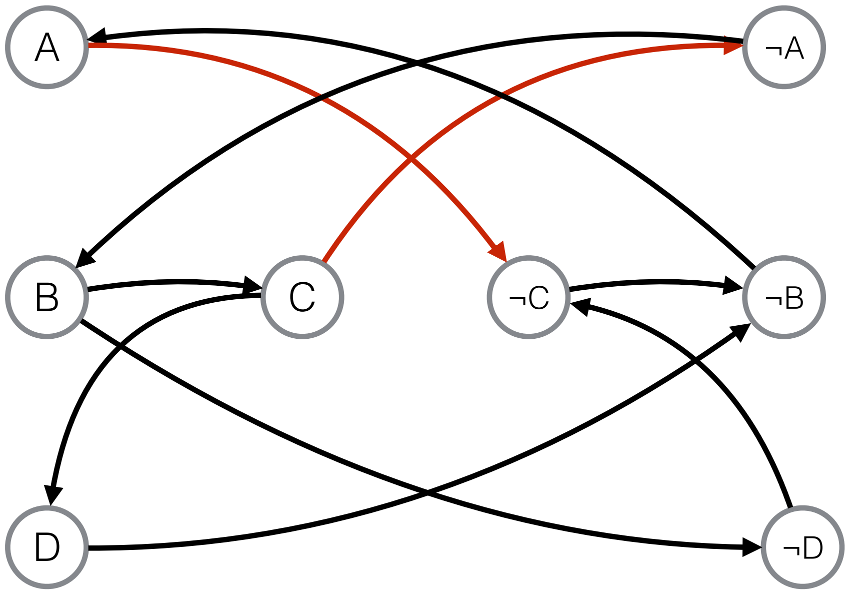


$$(\neg A \vee \neg C) \wedge (\neg B \vee C) \wedge (B \vee A) \wedge (\neg C \vee D) \wedge (\neg D \vee \neg B)$$

$$\equiv$$

$$(A \rightarrow \neg C) \wedge (B \rightarrow C) \wedge (\neg B \rightarrow A) \wedge (C \rightarrow D) \wedge (D \rightarrow \neg B)$$

$$(\neg A \lor \neg C) \land (\neg B \lor C) \land (B \lor A) \land (\neg C \lor D) \land (\neg D \lor \neg B)$$



$$(\neg A \lor \neg C) \land (\neg B \lor C) \land (B \lor A) \land (\neg C \lor D) \land (\neg D \lor \neg B)$$

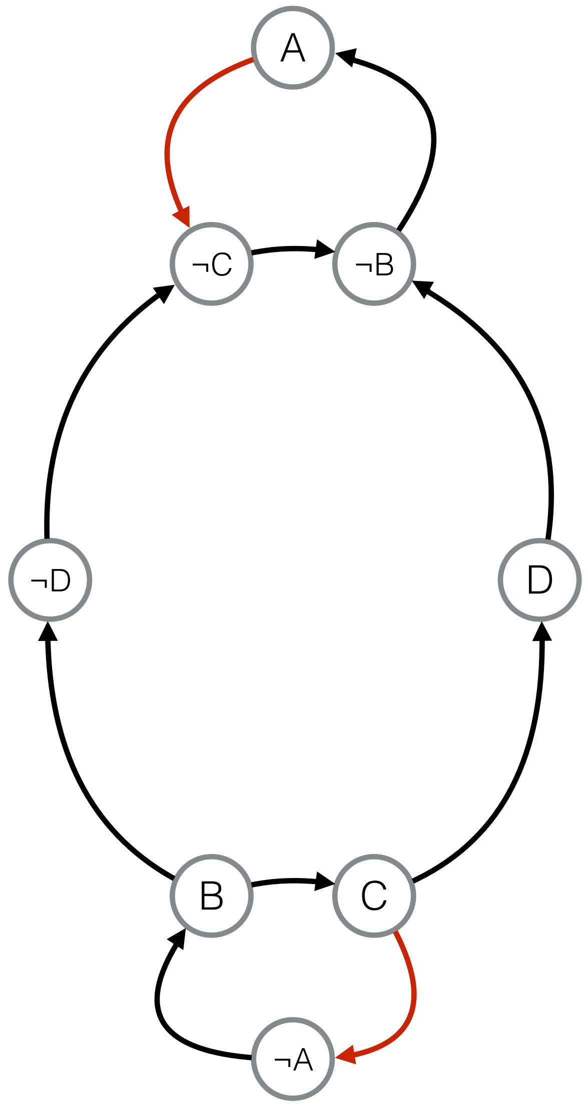
$$\equiv$$

$$(A \to \neg C) \land (B \to C) \land (\neg B \to A) \land (C \to D) \land (D \to \neg B)$$

Everyone must take Archeology

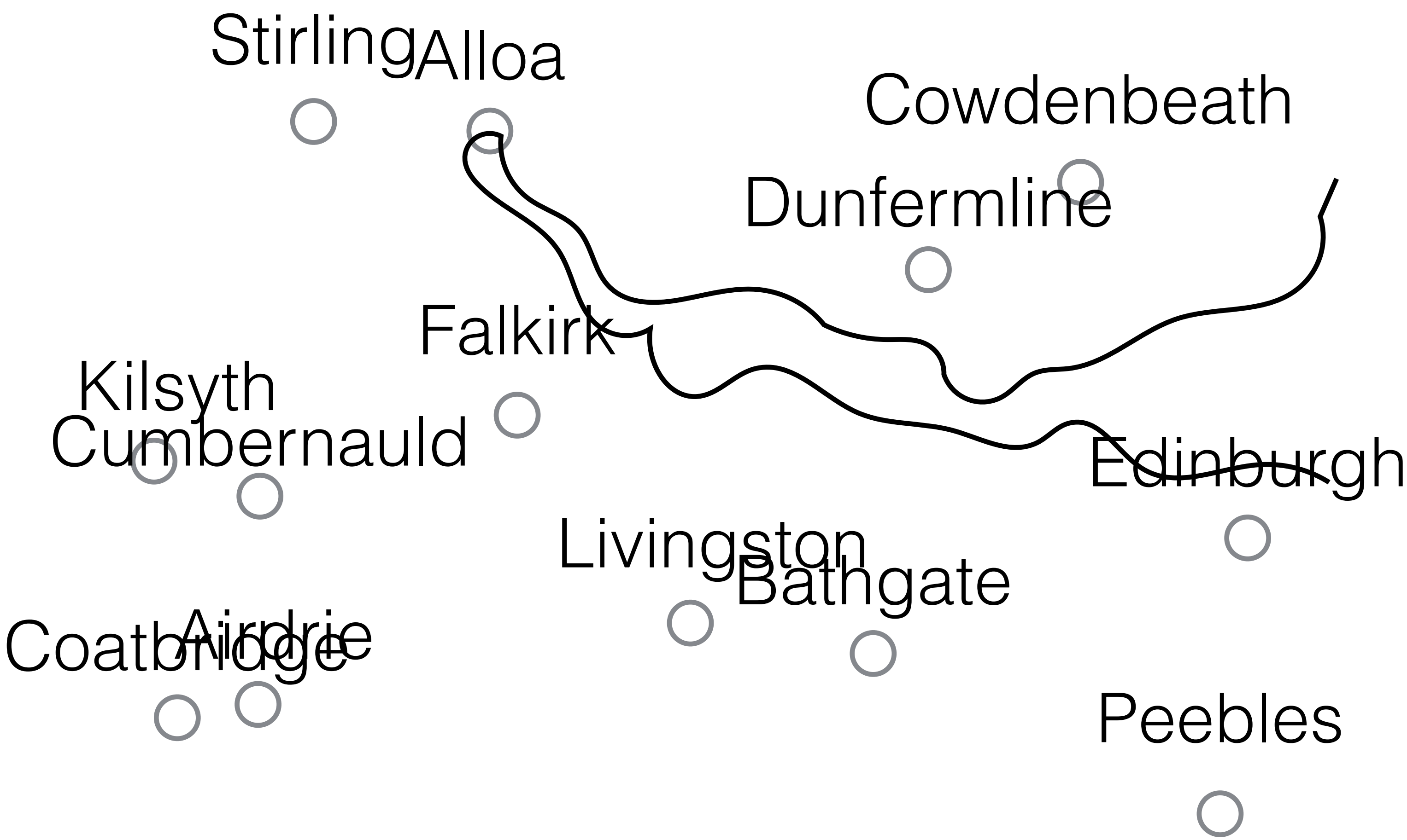
No one can take
Chemistry or Biology

Divinity is optional



labelling maps

labelling maps



labelling maps

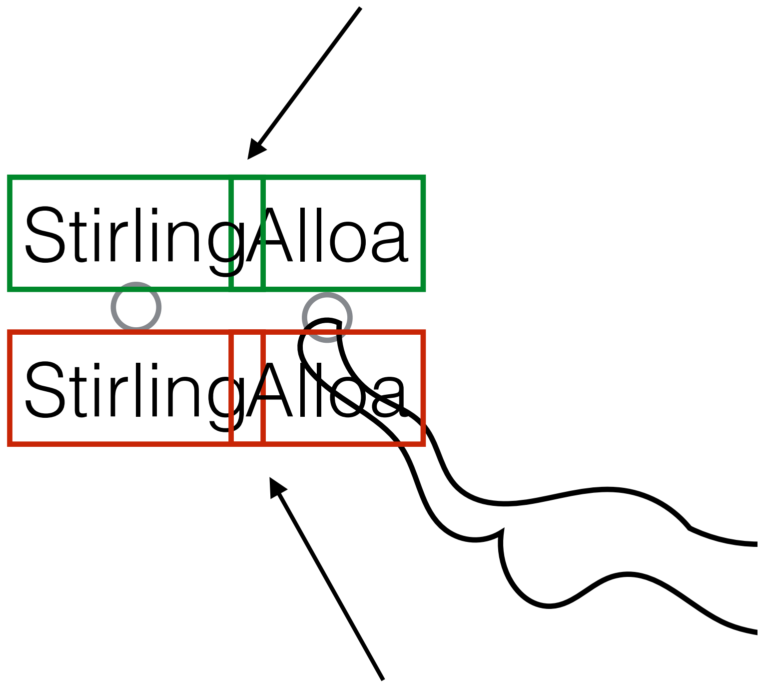
Suppose each label can be in one of two positions: top ($\top$) or bottom($\bot$).

We represent these by boolean variables S (Stirling) and A (Alloa)

We have a clash if both labels are at above their town, or if both are below.

We introduce two constraints to avoid these two possibilities.

$\neg(S \wedge A)$



$\neg(\neg S \wedge \neg A)$

$(\neg S \vee \neg A) \wedge (S \vee A)$

Solving all the constraints gives us a map where the labels don't clash.