A simple proof system for propositional logic.

We shall now present a method of testing the satisfiability of a finite set of propositional formulas.

This is done by using a tree analysis to decompose a formula into its basic constituents, which in this case is literals, i.e., propositional formulas or negations of such.

We work with formulas using all the logical connectives, $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$.

Note; If a formula A is not a literal it has one of the following forms:

(1) $A = \neg\neg B$ (2) $A = (B \wedge C)$ (3) $A = \neg(B \wedge C)$

(4) $A = (B \vee C)$ (5) $A = \neg(B \vee C)$

(6) $A = (B \rightarrow C)$ (7) $A = \neg(B \rightarrow C)$

(8) $A = (B \leftrightarrow C)$ (9) $A = \neg(B \leftrightarrow C)$

Derivation rules in tableaux form:

(1) $\neg\neg B$ ✓

B

(2) $(B \wedge C)$ ✓

B
C

(3) $\neg(B \wedge C)$ ✓

$\neg B$ | $\neg C$

(4) $(B \vee C)$ ✓

B | C

(5) $\neg(B \vee C)$ ✓

$\neg B$
$\neg C$

(6) $(B \rightarrow C)$ ✓

$\neg B$ | C

(7) $\neg(B \rightarrow C)$ ✓

B
$\neg C$

(8) $(B \leftrightarrow C)$ ✓

B C | $\neg B$ $\neg C$

(9) $\neg(B \leftrightarrow C)$ ✓

B $\neg C$ | $\neg B$ C

Given a finite set of formulas $A_1, A_2, \dots, A_n$, a <u>sentence tableau</u> for $A_1, \dots, A_n$ is a tree of formulas constructed as follows

- Begin by listing the formulas vertically:

A_1
A_2
$\vdots$
A_n

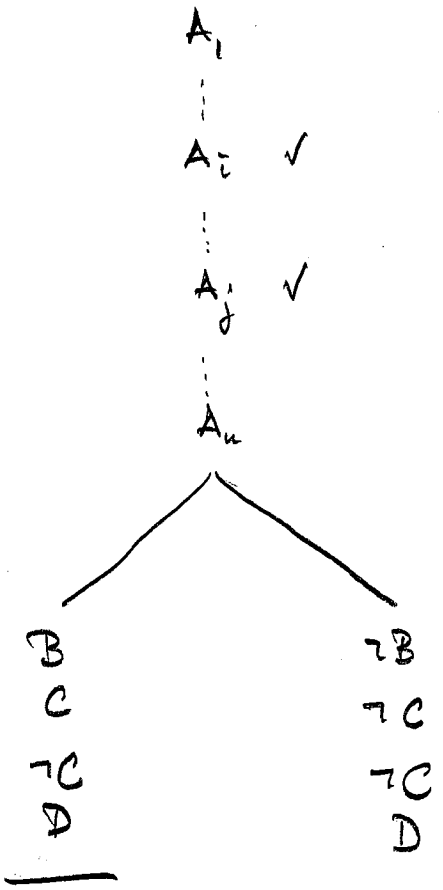
- Now, if A_i is the first formula on the list that is <u>not</u> a literal, apply the appropriate derivation rule to A_i; say $A_i = (B \leftrightarrow C)$:

A_1
$\vdots$
A_i ✓
$\vdots$
A_n

B
C

$\neg B$
$\neg C$

- Now along each branch of the tree constructed hitherto, we repeat the previous step; say $A_j = (\neg C \wedge D)$



- After each step, we check for satisfiability along branches. Thus the left-most branch contains a formula and its negation, namely C and ¬C. Therefore, the left-most branch is not satisfiable. So we can <u>close off</u> this branch and we will not analyse this further (indicated by a bar).

- Note that each analysed formula gets checked to indicate that we will not apply further derivations rules to it.

- We apply the previous steps along each branch that is not yet closed off until either

  (i) every branch is closed, or

  (ii) all non-literals along each unclosed branch is checked.

Then, in case (i), the initial set $A_1, \dots, A_n$ is not satisfiable. On the other hand, if (i) fails, i.e., some branch is not closed off, but (ii) holds, then the set of formulas $A_1, \dots, A_n$ is satisfiable.

Example Consider the set

$$\Delta = \{(P \wedge \neg Q) \rightarrow S \;,\; \neg Q \vee \neg R \;,\; P \wedge T\}.$$

A tableau for Δ:

$(P \wedge \neg Q) \rightarrow S$ ✓

$\neg Q \vee \neg R$ ✓

$P \wedge T$ ✓

P

T

$\neg Q$ | $\neg R$

$\neg(P \wedge \neg Q)$ ✓ | S (1) | $\neg(P \wedge \neg Q)$ ✓ | S (4)

$\neg P$ | $\neg\neg Q$ | $\neg P$ (2) | $\neg\neg Q$ ✓

Q (3)

Here there are 6 branches in total, 2 of which get closed off and 4 of which remain open.

To see that an unclosed branch that has been fully analysed, i.e., all of whose unchecked formulas are literals, implies satisfiability, it suffices to define a valuation v by letting

$$v(P) = \begin{cases} F & \text{if } \neg P \text{ appears along the branch,} \\ T & \text{if not.} \end{cases}$$

For example, if we use the open branch (1), we get

$$v(P) = v(T) = v(S) = v(R) = T$$

$$v(Q) = F.$$

Thus

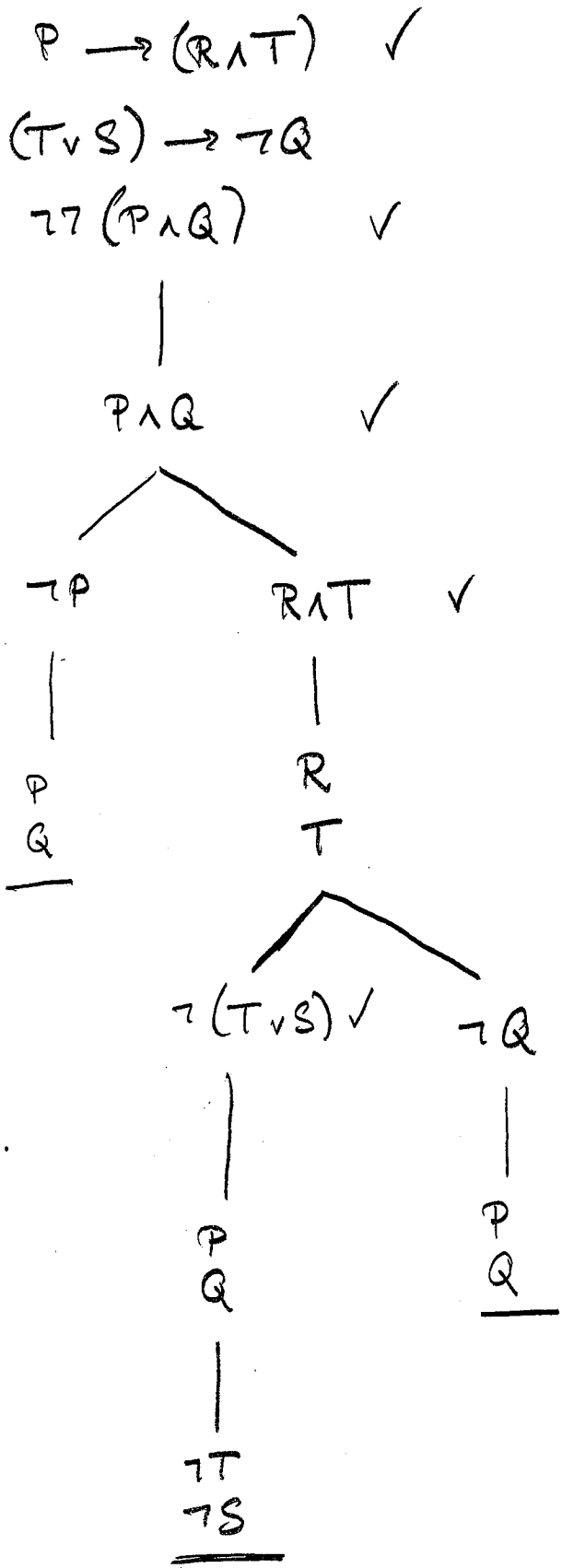
$$v((P \wedge \neg Q) \rightarrow S) = T$$

$$v(\neg Q \vee \neg R) = T$$

$$v(P \wedge T) = T$$

Note that the propositional variables not occurring in a literal along the branch can be assigned any value T or F.

Example $P \to (R \wedge T),\ (T \vee S) \to \neg Q,\ \neg\neg(P \wedge Q)$:



Since every branch closes, the set is not satisfiable.

Definition

An argument

$$\frac{\begin{array}{l} A_1 \\ A_2 \\ \vdots \\ A_n \end{array}}{B}$$

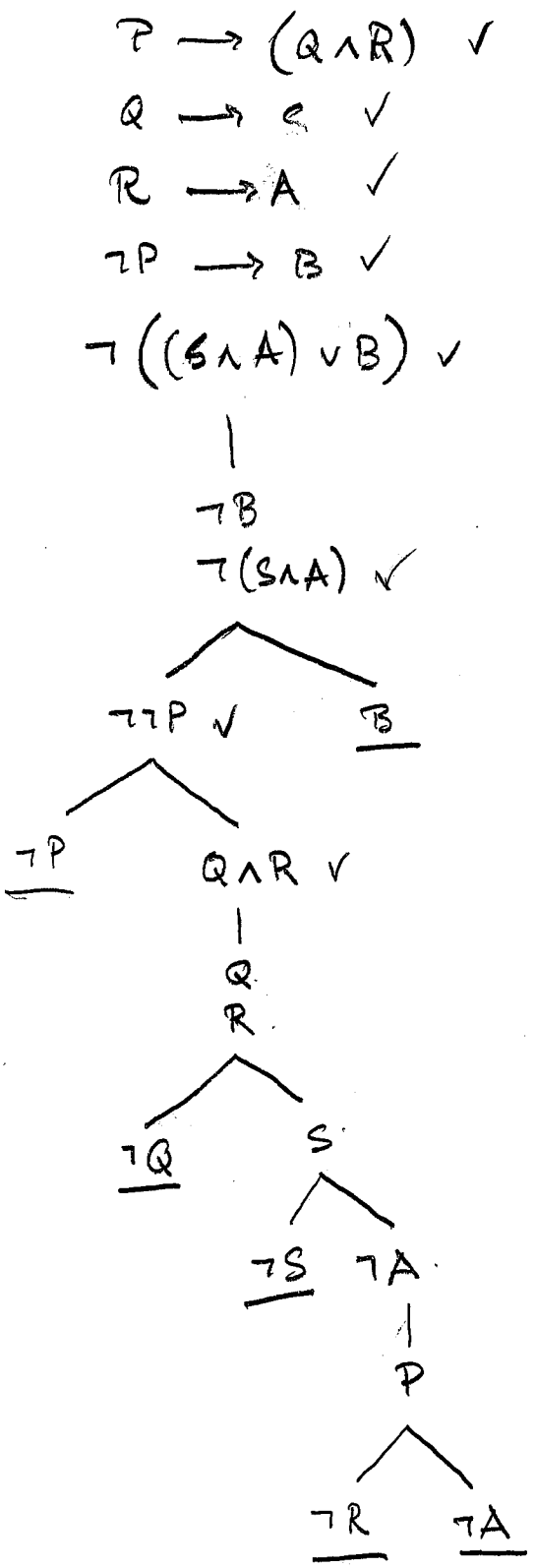
is said to be valid if the conclusion B is a tautological consequence of the premises $A_1, \ldots, A_n$, i.e., $\{A_1, A_2, \ldots, A_n\} \models B$.

Example

$$\frac{\begin{array}{l} P \longrightarrow (Q \wedge R) \\ Q \longrightarrow S \\ R \longrightarrow A \\ \neg P \longrightarrow B \end{array}}{(S \wedge A) \vee B}$$

This argument is valid if and only if for any valuation v satisfying the hypotheses, v also satisfies the conclusion. Or in other words, if and only if the hypotheses plus the negation of the conclusion is not satisfiable.

So we check for satisfiability:



Every branch closes and hence the argument is valid.