Propositional tableaux

method to prove the unsatisfiability of a set of formulae

principle:

break each formula into its components up to the simplest ones, where contradiction is easy to spot

create a tree structure called *tableau*
(plural: tableaux)

Propositional tableau: example

to prove the unsatisfiability of the set

$\{ a \land c, (\neg a \lor b) \land (\neg b \lor \neg c) \}$

first place the formulae in column:

$$\begin{array}{c} a \land c \\ | \\ (\neg a \lor b) \land (\neg b \lor \neg c) \end{array}$$

Example, second step

since we have $a \land c$, place a and c below the other formulae:

$$\begin{array}{c} a \land c \\ | \\ (\neg a \lor b) \land (\neg b \lor \neg c) \\ | \\ a \\ | \\ c \end{array}$$

Example, third step

same as before, but for $(\neg a \lor b) \land (\neg b \lor \neg c)$

still a conjunction: place formulae below the other ones:

$$\begin{array}{c} a \land c \\ | \\ (\neg a \lor b) \land (\neg b \lor \neg c) \\ | \\ a \\ | \\ c \\ | \\ \neg a \lor b \\ | \\ \neg b \lor \neg c \end{array}$$

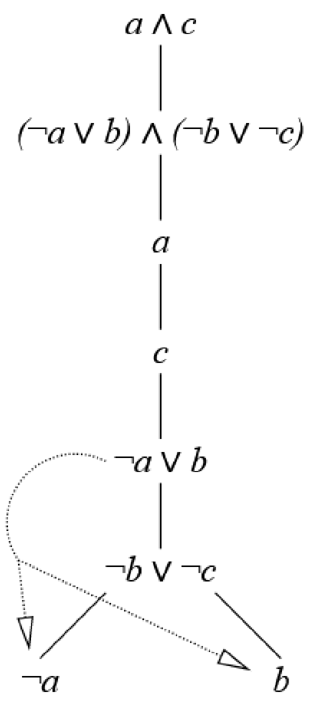
Example, fourth step

we already broke the two conjuctions $a \wedge b$ and $(\neg a \vee b) \wedge (\neg b \vee \neg c)$ into their components

do the same for the disjunctions

but, for disjunction make two branches

for $\neg a \vee b$:

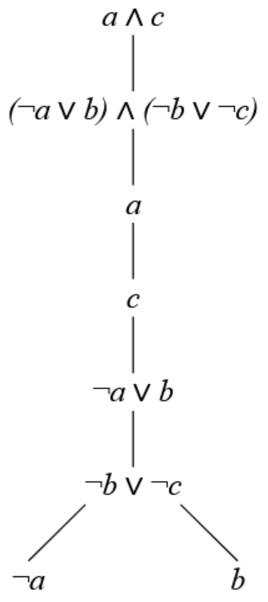


what did we do, so far?

Meaning of branches

each branch (path from root to a leaf) is a different way to satisfy the formulae in the original set

in this case:



two ways to satisfy the set:

- make true all formulae in the branch $a \wedge c$ ... $\neg a$
- make true all formulae in the branch $a \wedge c$ ... b

Example, fifth step

a and $\neg\, a$ in the same branch

contradiction

Rules of expansion

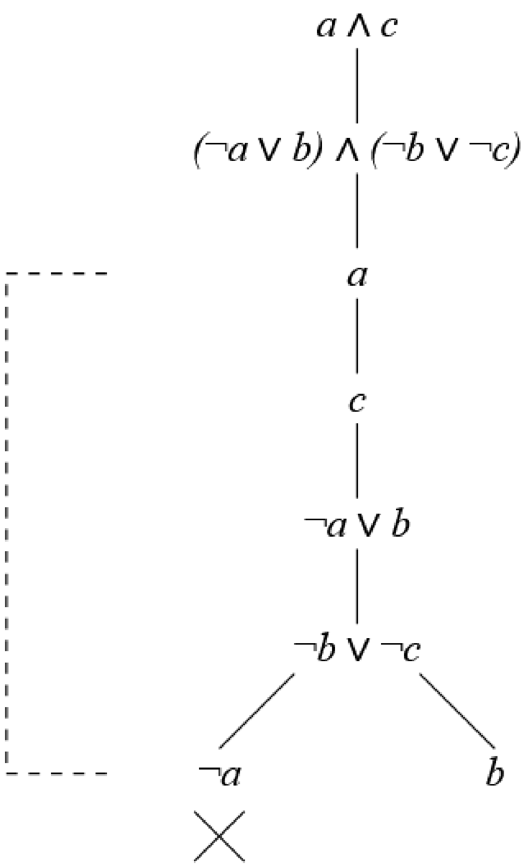
to satisfy $A \wedge B$ we have to satisfy both A and B

we place both A and B below

to satisfy $A \vee B$, we have to satisfy either A or B

two different ways to satisfy the same formula

we make a branch for A and one for B



Contradiction

each branch is a way to satisfy the set

a and $\neg\, a$ in the same branch

this possibility is not viable

close the branch (mark it with X)

what does X mean?

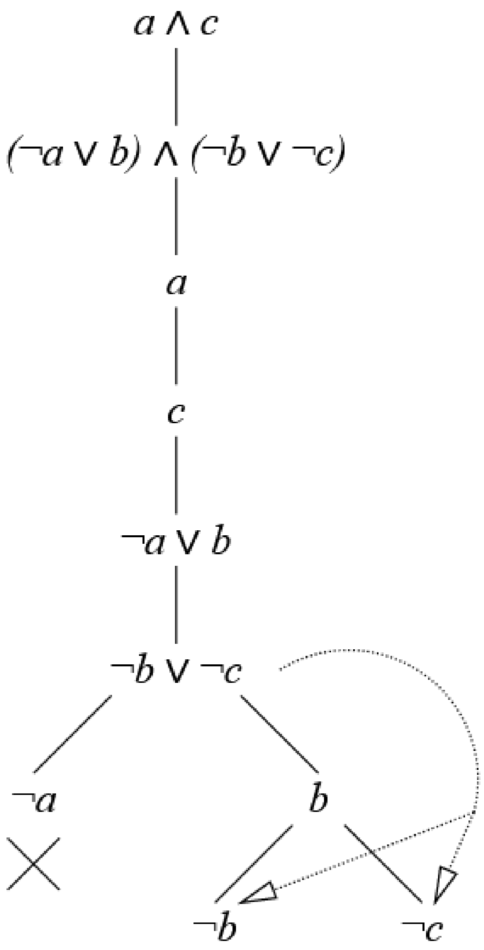
Example, sixth step

expand $\neg b \lor \neg c$

the first branch is closed

we already excluded it as a possible way to satisfy the set

go in the other possibility (the other branch)

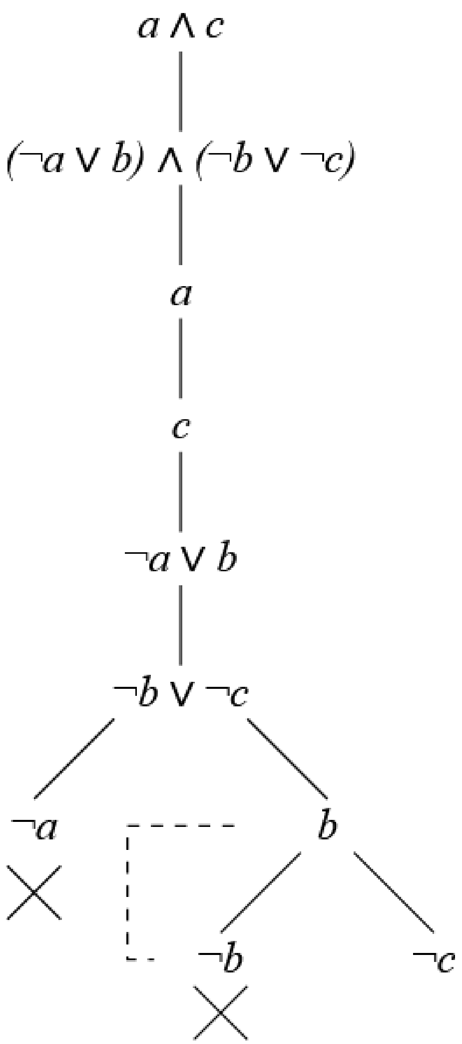


Example, seventh step

b and $\neg b$ in the same branch

(branch from $a \land c$ to $\neg b$)

close branch



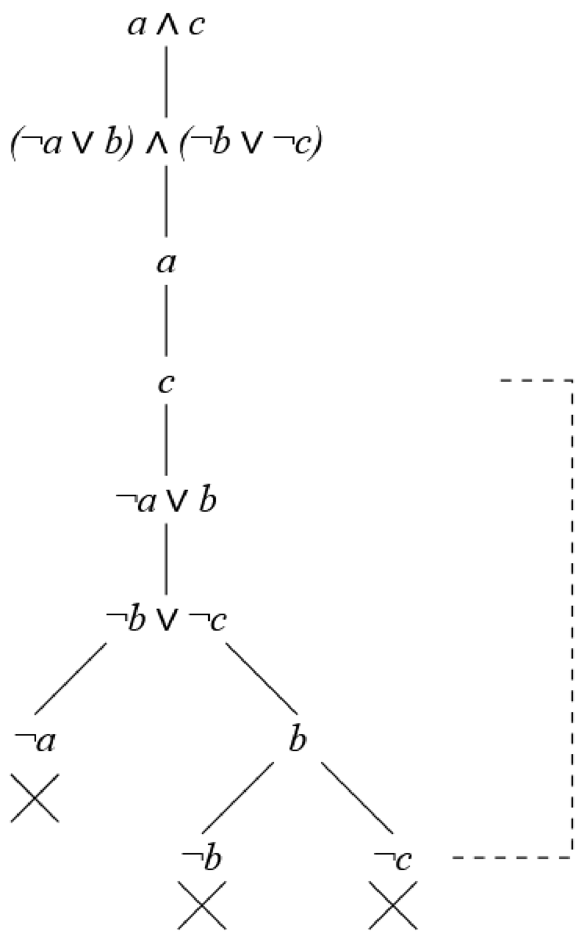
Example, eighth step

c and $\neg c$ in the same branch

note that each branch is a full path from root to a leaf

in this case, from $a \land c$ to $\neg c$

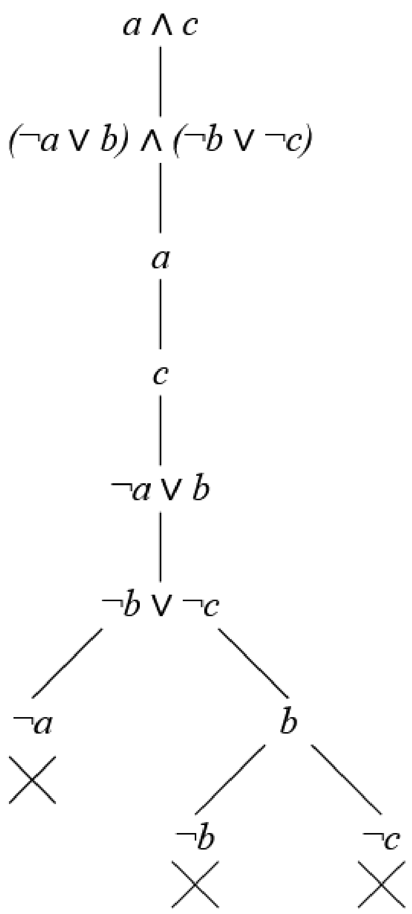
close branch



conclusion?

Example: conclusion

final tableau:



each branch is a possible way to satisfy the set

closed branch = possibility excluded because of contradiction

all closed = no way to satisfy the set

conclusion: $\{a \wedge c, (\neg a \vee b) \wedge (\neg b \vee \neg c)\}$ is **unsatisfiable**

Propositional tableau: rules

- place formulae in a line
- expand according to the following rules:

$$\frac{A \wedge B}{\begin{array}{c} A \\ B \end{array}} \qquad \frac{A \vee B}{A \mid B}$$

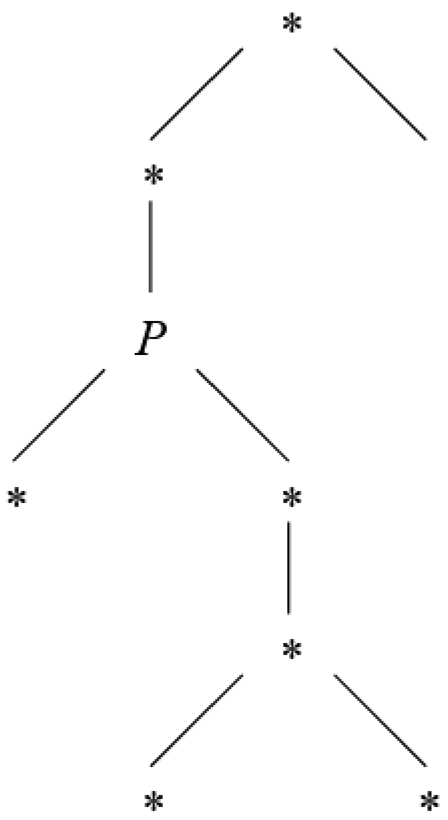
- if a branch contains complementary literals (e.g., x and $\neg x$), close the branch

addition: do not add formulae to closed branches

logics different from propositional logic require other rules

Application of rules

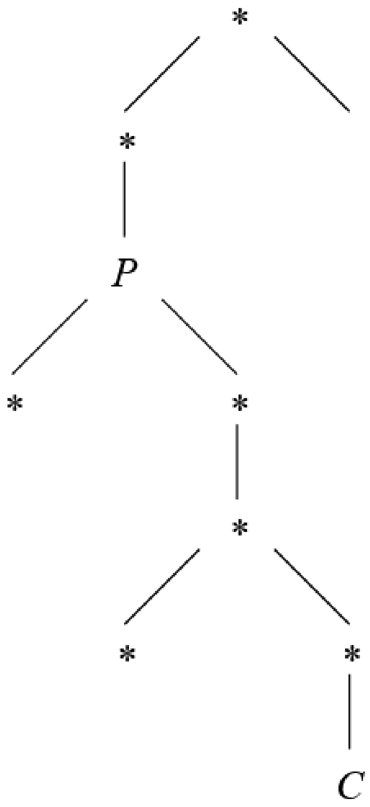
in this situation:



we can apply

$$\frac{P}{C}$$

getting:



pretty obvious in this case

not so for tableau for other logics (e.g., first-order logic)

Another example

$\{(a \vee b) \wedge c,\ \neg b \vee \neg c,\ \neg a\}$

first step: place formulae in a line

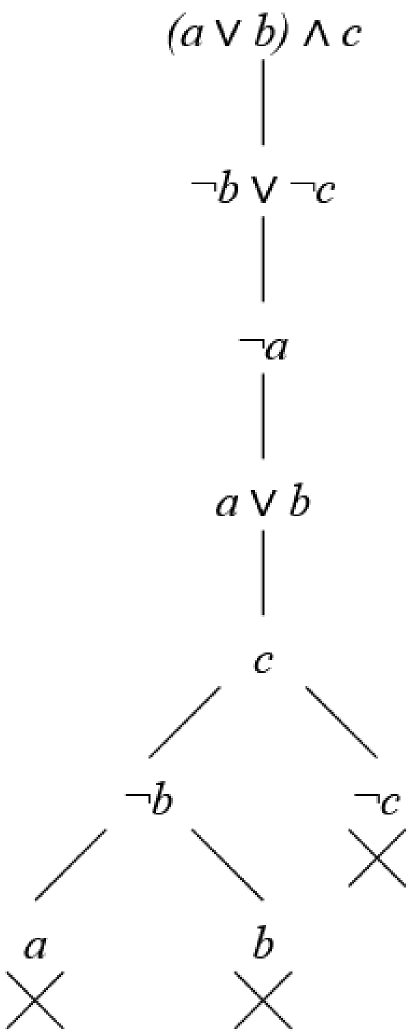
$(a \vee b) \wedge c$

$\neg b \vee \neg c$

$\neg a$

follow rules of expansion

Solution

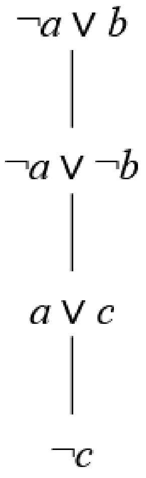


all branches close ⇒ set is unsatisfiable

what if some branches do not close?

Open branches: multiple application

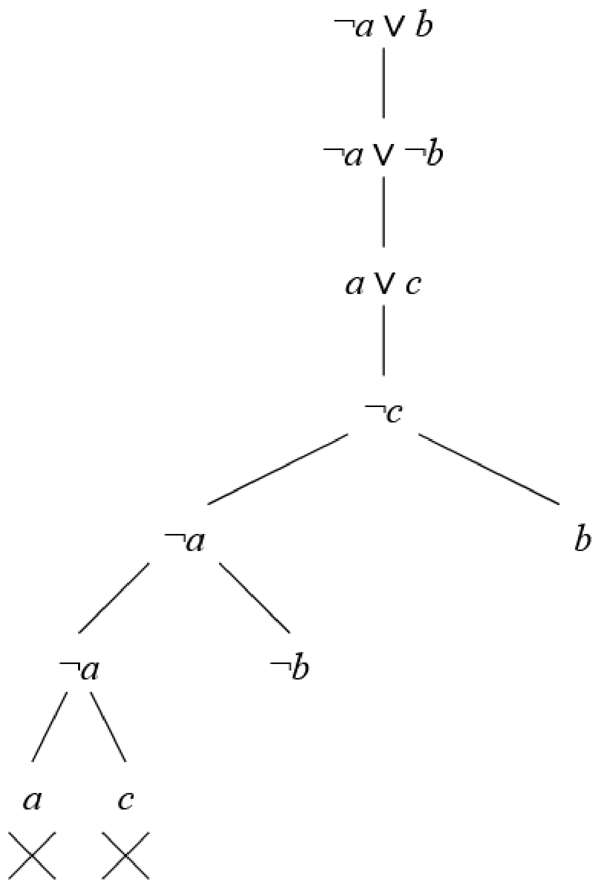
$\{\neg a \lor b, \neg a \lor \neg b, a \lor c, \neg c\}$



expand every formula once

enough?

After expanding every formula once



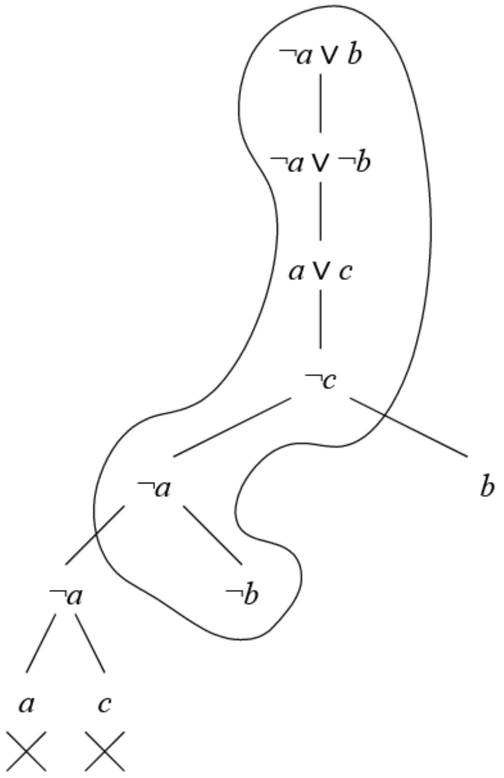
every formula has been expanded at least once

tableau not closed

yet, set is unsatisfiable

The choice not taken

consider the branch ending in b



each branch is a different way to satisfy the set

the formulae in this branch are: $\neg a \lor b$, $\neg a \lor \neg b$, $a \lor c$, $\neg c$, $\neg a$, $\neg b$

for $\neg a \lor b$ we took $\neg a$

for $\neg a \lor \neg b$ we took $\neg b$

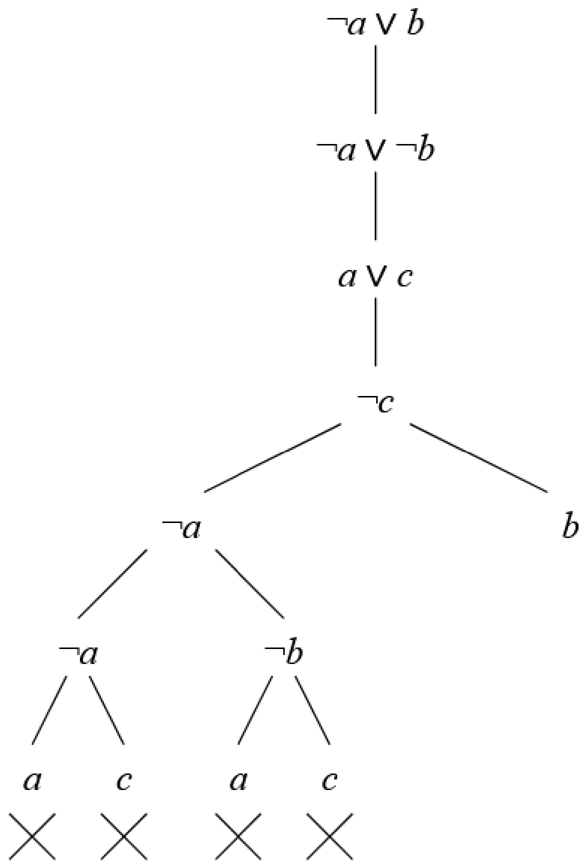
no choice have been made for $a \lor c$

choose either a or c

in terms of tableaux?

Multiple applications

in terms of tableaux: expand $a \vee c$

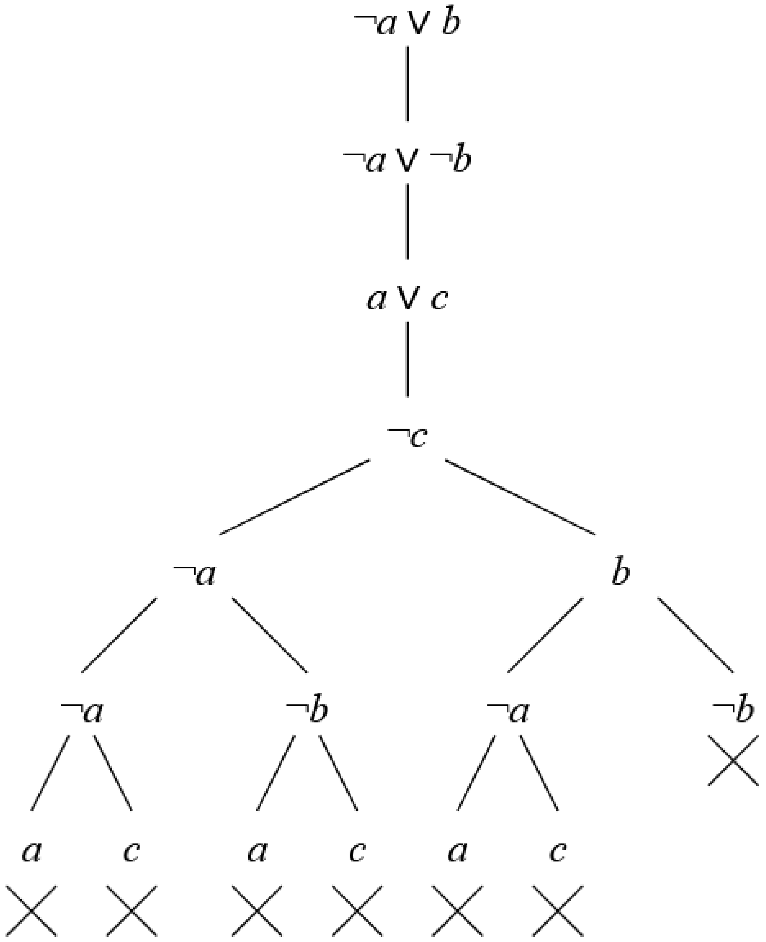


principles:

- each branch is a possible way to satisfy the formulae
- for every possibility, break formulae into their smallest components

in the example, there are still unbroken formulae in the branch ending in b

Multiple application: final tableau



general rule:

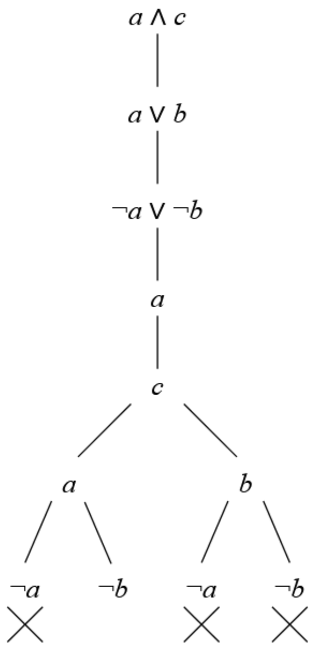
in every branch, every formula has to be expanded once

Open branches: satisfiability

$\{a \wedge c,\ a \vee b,\ \neg a \vee \neg b\}$

expand every formula in every branch

Satisfiability: final tableau



in the second branch ($a \wedge c$ ... $\neg b$) every formula has been expanded once:

$a \wedge c$
: taken both a and c

$a \vee b$
: chosen a

$\neg a \vee \neg b$
: chosen $\neg b$

this is a way to satisfy the set that does not lead to contradiction

the set is satisfiable

model: take the literals in the branch

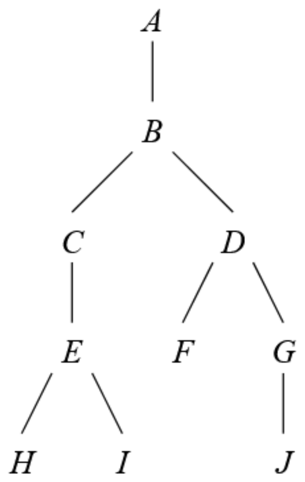
$a,\ c,\ a,\ \neg b$

model: $\{a{=}true,\ b{=}false,\ c{=}true\}$

Semantics of tableau

given a tableau, its semantics is a formula:

- the semantics of a branch is the conjunction of the formulae in the branch
- the semantics of the tableau is the disjunction of the formulae of all its branches



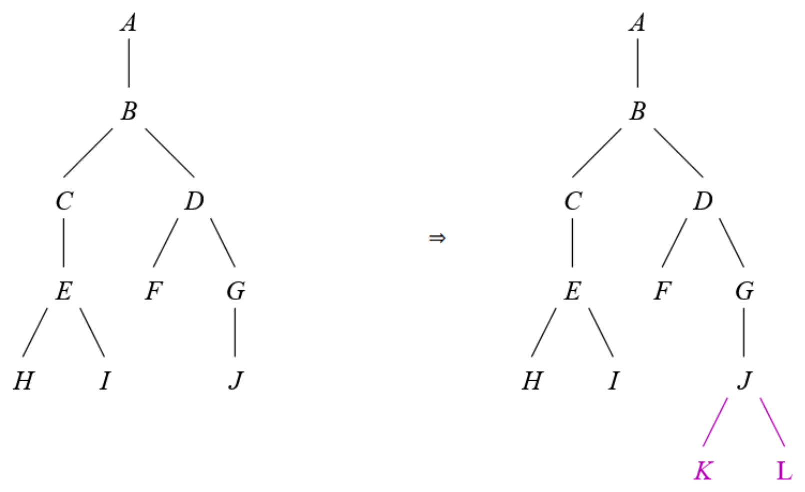
in this example, four branches

- $B_1 = A \wedge B \wedge C \wedge E \wedge H$
- $B_2 = A \wedge B \wedge C \wedge E \wedge I$
- $B_3 = A \wedge B \wedge D \wedge F$
- $B_4 = A \wedge B \wedge D \wedge G \wedge J$

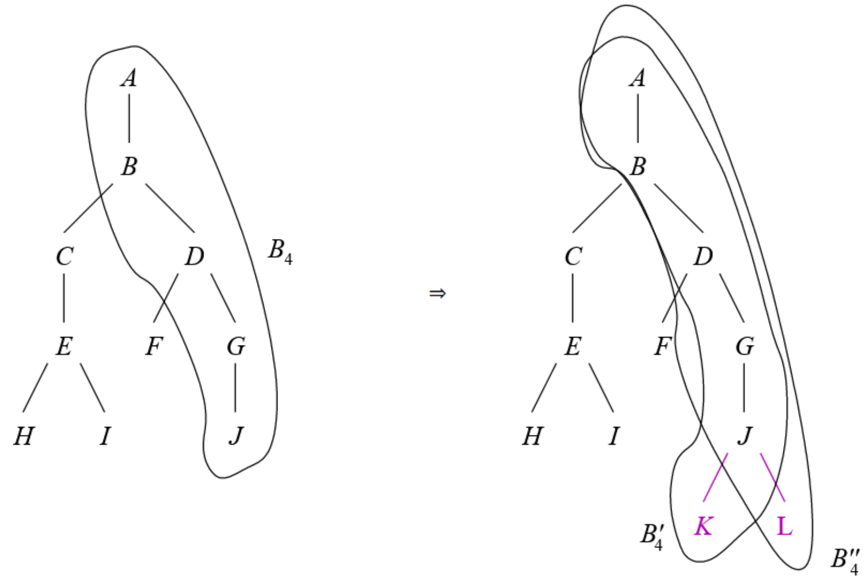
semantics of the tableau is $B_1 \vee B_3 \vee B_3 \vee B_4$

Rules of expansion

let $D=K \vee L$, expand it on J



the semantics changes: the last branch is made two



Semantic change, in formulae

old semantics: $B_1 \vee B_3 \vee B_3 \vee B_4$

B_4 is replaced by two new formulae:

$$B_4 = A \wedge B \wedge D \wedge G \wedge J$$
$$B'_4 = A \wedge B \wedge D \wedge G \wedge J \wedge K \;=\; B_4 \wedge K$$
$$B''_4 = A \wedge B \wedge D \wedge G \wedge J \wedge L \;=\; B_4 \wedge L$$

new semantics: $B_1 \vee B_2 \vee B_3 \vee B'_4 \vee B''_4$

equivalent to: $B_1 \vee B_2 \vee B_3 \vee (B_4 \wedge K) \vee (B_4 \wedge L)$

equivalent to: $B_1 \vee B_2 \vee B_3 \vee (B_4 \wedge (K \vee L))$

since B_4 contains $K \vee L$, this is equivalent to the original formula

a similar fact holds for conjunctions: expanding a tableau creates a new one with an equivalent semantics

How tableaux work

given $\{A, B, C\}$, place them in a line

the formula of this tableau is $A \wedge B \wedge C$

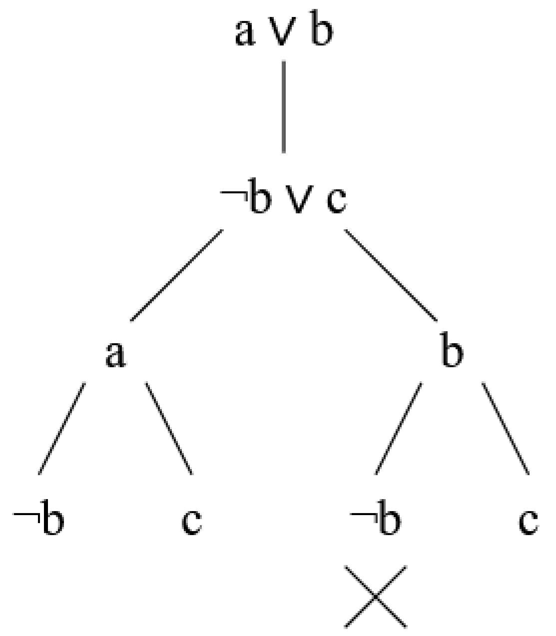
expand formulae, which means:

- creating simpler formulae from complex ones
- still maintaining equivalence with the original set

Partial models

a tableau for a satisfiable set detects a number of partial models of the formula covering all models of the formula

example: *{a ∨ b, ¬b ∨ c}*



the three unclosed branches lead to a partial model each:

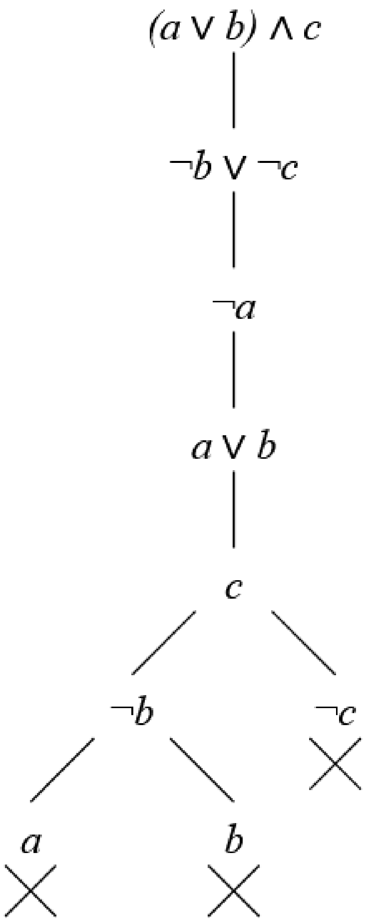
- *{a=true, b=false}*
- *{a=true, c=true}*
- *{b=true, c=true}*

every model of the set can be obtained by setting the unassigned variable to an arbitrary value in one of these three partial models

Propositional tableaux: policy

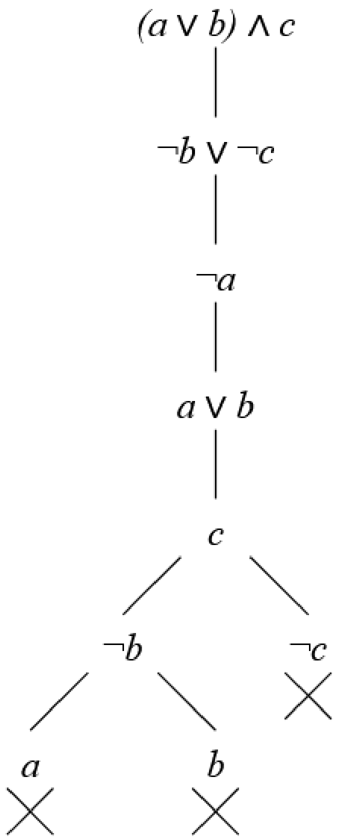
consider again the set $\{(a \vee b) \wedge c, \neg b, \vee \neg c, \neg a\}$

expanding first conjunctions and then disjunctions, we get:

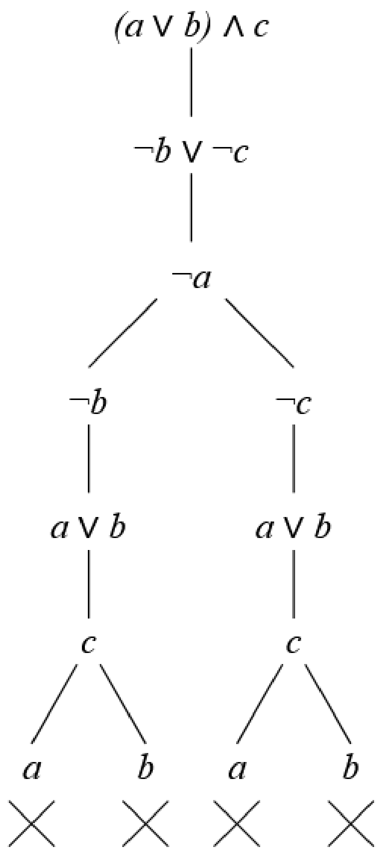


what happens if we do the opposite? (first disjunctions then conjunctions)

Conjunctions first vs. disjunctions first



conjunctions first



disjunctions first

expanding disjunctions creates new branches

conjunctions may need to be expanded in all of them

better expand conjunctions first

Effects of wrong policies on semantics

using the wrong policy (e.g., expanding *disjunctions* first) leads to an increase of size of the table, which leads to an increase of time

yet, unsatisfiability is still proved if set is unsatisfiable

this is not the case for other logics, where applying the wrong policy may inhibit proving unsatisfiability of some unsatisfiable sets

Refutation and entailment

the method of tableaux is a system for *refutation*

it can prove that a set is unsatisfiable

we can use it to prove entailment:

$A_1,\ldots, A_n \Rightarrow B$ if and only if $\{A_1,\ldots, A_n, \neg B\}$ is unsatisfiable