Planning and Reasoning

Sveva Pepe

Monday 21st December, 2020

1 Propositional Natural Deduction

Natural deduction is a mechanism not for satisfiability, not for contraddiction, not like GSAT or DPLL. It still start with propositional version but it is not a mechanism for satisfiability, not directly because we can establish satisfiability because the satisfiability is the coverse of validity and validity can be establish by deduction. But the point is not directly like this, it is not like we look for a model, and we aren't trying to derive a contraddiction, not necessarly.
Natural deduction is basically a method for deduction, meaning that it is based on the principle that we start from some set of formula and we deduce things from them, so we draw conclusions.

$$\frac{A \quad B}{A \land B}(\land I) \qquad \frac{A \land B}{A}(\land E_1)$$

$$\frac{A \land B}{B}(\land E_2) \qquad \frac{A}{A \lor B}(\lor I_1)$$

There are severals rules, like 20 rules, it is not so simple like the tableau method, but the advantage w.r.t tableau method is that deduction is always a "line", never a tree. For instance, if we derive A and we derive B then we can also derive $A \land B$. And, of course, if we derive $A \land B$ we can derive A, but we can also derive B. Given A we can conclude $A \lor B$.

Some of the rules are kind of very simple but we need them anyway, we need to list them, we enumerate all of them. Each rule has a name, the name is inside the parenthesis.

Example

prove that $x \land y \vDash y \land (x \lor z)$

$x \land y$

This is a very simple example because there is one things that will need in Natural deduction which are assumptions. In the simplify versione there is no assumption. $x \land y$ is the premise and we want to prove that $x \land y$ implies $y \land (x \lor z)$.
We start with $x \land y$ and we use the rules to obtain some other formula.

$$\begin{array}{l} x \land y \\ x \end{array} \qquad (\land E_1)$$

In the figure we apply the rule $(\land E_1)$, the rule of conjunction elimination. It is a rule by which given a conjunction we can derive one of the two elements of the conjunction. In this case we derive x. It is not like we have every possible implication in propositional logic, the rules are just a very simple subset of them. One thing that we know for certain is that $A \land B$ implies A, and there is a rule that sayas that given A and B we also have A and we apply that. This is called elimination of conjunction because we start with conjunction and we obtain a formula where conjunction disappears.

$$\begin{array}{l} x \land y \\ x \\ x \lor z \end{array} \qquad (\lor I_1)$$

Another rule is the disjunction introduction. It is a rule saying that if we have a formula we a formula we have the disjunction, so given x we have $x \lor z$.

$x \wedge y$

x

$x \vee z$

y $(\wedge E_2)$

Again we apply the second rule for conjunction elimination, so given $x \wedge y$ we derive y.

$x \wedge y$	$x \wedge y$
x	x
$x \vee z$	$x \vee z$
$(\wedge I)$ y	y
$y \wedge (x \vee z)$	$y \wedge (x \vee z)$

Rule of introduction of conjunction that says that everything looks very simple at this point but the complication will be later. According to this rule if we have two formulas we have also the conjunction, then the proof is completed, see the right figure. As you can see we have formula written on a single line and not a tree, and, everytime we just derive consequences. The conclusion is that the first formula implies the last.

We can see that we have set of rules and some rules break the formula and others build the formula. For example in conjunction we break the formula and we have two separate rules for two separate parts of conjunction, instead in the tableau method we have immediately two part separate when we apply the conjunction. In this case, if you want the first part of the conjunction we apply a rule and if you want the other part you apply different rule. We have also rules for building formula given their components.

rules

all in the form:

$$\frac{\textit{formula}, \ldots, \textit{formula}}{\textit{formula}}$$

inference

to prove $A_1, \ldots, A_n \models B$

- place preconditions in a line:

  A_1

  $\vdots$

  A_n
- if there is a rule like

  $$\frac{\textit{formula}, \ldots, \textit{formula}}{\textit{formula}}$$

  and *formula, ... , formula* are all in the line add *formula* at the bottom
- if B is obtained, proof is successful

The general method is like this, so, we have a set of rules and all rules are in the form: *given a set of formula we have another formula.* This is also different from the tableau method because in the **tableau method** we have a *single formula as a premise and return a set of formula.*

In some cases of deduction we can have 1 premise and 1 consequence, but generally we have a set of premises and a single conclusion. We want to prove that a set of formula implies another formula and we start with precondition on the line and if we have set of formula and the formulas are already in the line then we can obtain the consequence, so we can add the formula in the both of the rule that we decide to apply. The proof works, deduction is proved if we have the formula that we want to prove at the end of the line.

Since we add only one formula at time, so all rules have a single consequence, we can just check final formula of the line, we do not need anything in the middle.

$$\frac{A \quad B}{A \wedge B}(\wedge I) \qquad \frac{A \wedge B}{A}(\wedge E_1) \qquad \frac{A \wedge B}{B}(\wedge E_2)$$

$$\frac{A}{A \vee B}(\vee I_1) \qquad \frac{B}{A \vee B}(\vee I_2) \qquad \frac{A \vee B \quad A \to C \quad B \to C}{C}(\vee E)$$

$$\frac{???}{???}(\to I) \qquad \frac{A \quad A \to B}{B}(\to E)$$

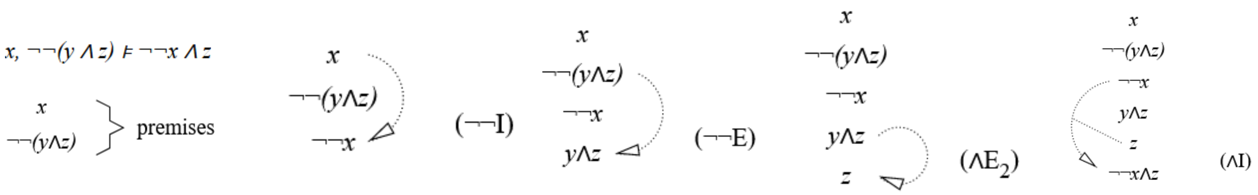
$$\frac{A \to \bot}{\neg A}(\neg I) \qquad \frac{A \quad \neg A}{\bot}(\neg E) \text{ or } (\bot I) \qquad \frac{\bot}{A}(\bot E)$$

$$\frac{A}{\neg\neg A}(\neg\neg I) \qquad \frac{\neg\neg A}{A}(\neg\neg E)$$

$$\frac{\neg A \to \bot}{A}(RA) \qquad \frac{A \to B \quad \neg B}{\neg A}(MT)$$

There are many rules but not all af them are necessary. In the first line there are rules for conjunction, build a formula or break the formula. Second line we have disjunctions, in the first two rules we build a formula and in the third we remove disjunction, it is little bit complicated because we need the disjunction and both element of disjunction should imply the same element, in this case $A \to C$ and $B \to C$, so we can deduce C. In third line, in the second rule we have modus ponens, so we remove implication by modus ponens. In the last line, in the second rule we have modus tollens in which an element implies another and we have the negation of one that we implies, in this case $\neg B$, so we derive the negation of the premise of the implication, in this case $\neg A$.

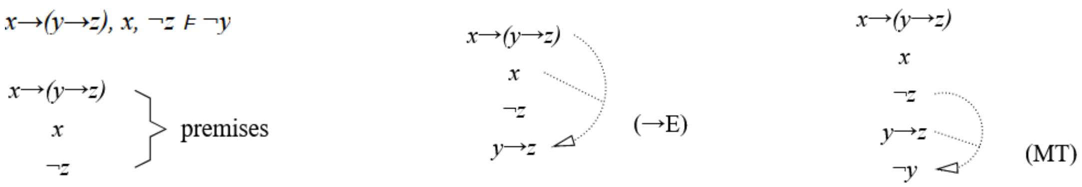
Example



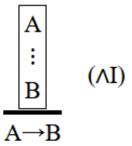
One of the big problem of the deduction method is that it is not obvious how to apply the rule. Many many rules but it is not obvious how to apply them, while in the tableau method we have a formula and we split the formula, it is always in this way so we have a choise to which formula to split but otherwise if we have a conjunction and disjunction the only doubt is that which one we need to break first. In tableau all rules are breaking formula.
In natural deduction, instead, we might sometime introduce formula, so it is even possible, for example, start with x apply the rule for double negation and introduce another formula $\neg\neg x$ and repeat and for instance introduce $\neg\neg\neg\neg x$. It is possible, there is no rule forbidding the repeated application of a rule; it is not like we have a limit. Even the propositional case doesn't consider quantifier (forbid quantifier), still the number of rules that can be applied is infinite.
In the third figure above (from the left) we remove the double negation because the final formula does not contain the negation of y and z so we might remove it. Then we split, see fourth figure above, so at least this rule can be applied only in a finite number of ways. Finally, we joint $\neg\neg x \wedge z$, figure 5 and the proof is proved.
When I say that rules are valid what I mean is that if the premises are true then consequences are real consequence according to propositional logic, so all rules obe y to propositional logic meaning that we always obtain correct conclusion. If we add a formula at the end of Natural deduction proof, this formula is a consequence, this is guaranteed to a logical conclusions to the premises. It is very easy to prove because if you look at the rules you can see that every consequence of a rule is really a consequence of the premises. All rules are correct according to propositional logic. We have the correctness.
The problem, indeed, is not the correctness but the completeness.

Can we prove?
We always prove that if formula follows from another formula then we can add formula at the end of another deduction? Is a formula a consequence always at the ending point of another deduction? This is not obvious, we will not see the formal proof but it can be prove. But for this we need the rule for introducing the implication.

Example 2 - implication rule

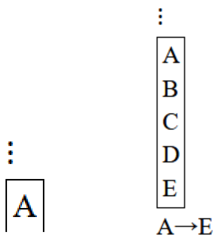


We have 3 premises and 1 consequence. We start with premises and we apply modus ponens (see figure 2 from the left). Then we apply modus tollens, figure 3 from the left, so given an implication and the negation of the conclusions we derive the negation of the premises. The statement is proven because $\neg y$ is consequence of the three starting formulas.
We need to introduce implication because otherwise the method is incomplete; even if we do not have implication. Implication is also involved in a way with disjunctions. The method is only complete regarding conjunctions but still incomplete regarding implication and disjunction, so the method is not complete so with just conjunction we do not have complete propositional logic.
We might make an assumption, so we assume a formula and we draw conclusion for this formula, and the conclusion is not something which is true in general but it is only true given the assumption.



We introduce an assumption A. At any point in the proof we can introduce which ever formula we want, it is a random assumption and them we proceed with the method using the assumption A and at some point we obtain B. The conclusion is not a B true in general but it is only true given A, so we conclude that A implies B.

What we conclude is that $A \to B$ so we assume A, we derive conclusion, we use the rule to derive effects of the formula that we have so far and at the end we obtain B and what we prove is that A implies B is true.



We start with premises, we derive conclusions and at any point we can say that we can assume x. Even if x is not true, even if x is actually contraddictory with what we know so far. So, even if we conclude $\neg x$ we might still introduce x if we want. It is a complete arbitrary formula, so we open a box with arbitrary command and the only need for a box is a reminder that we are working not in general but under this assumption, we have formulate this assumption. This is a reason with we have a box, it is a reminder, in the figure on the left.
Then we proceed draing a conclusion, we still apply the rules and we go ahead. At any point we can close the box and then after closing the box, we can draw the formula outside the box, like in the figure on the right.

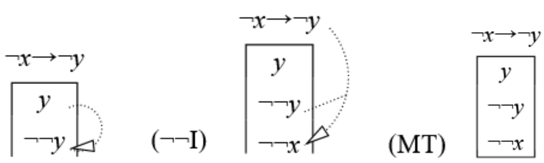
All sequence prove that $A \to E$.
The idea is that if we can use any assumption that we want and we can close the box at any point, **when we should do that?** We should do that if we believed that formula A implies B could be useful for the overall proof. We try not to derive the overall conclusion that we are trying to obtain but only B. We want only that $A \to B$.

Example Assumptions

$\neg x \to \neg y \vDash y \to \neg\neg x$

Basically we start with premises and there is not rule that we can apply at this point because, ok there is a rule for removing implication using modus ponens but we do not apply, so there is no rule that we can apply. But we can still open a box with assumption, figure on the right.

We can ass assumption if we do not have any rule to apply, and this is something that we cannot do in the tableau method because if we cannot apply any rule then we say that branch reamins open. Here, instead, we can always opening an assumption. We assume y because in the conclusion we have $y \to \neg\neg x$. There are two ways to obtain $y \to \neg\neg x$: by splitting the biggest formula or by building it. Since this $y \to \neg\neg x$ is not part of something that we already know we need to build it. The only way to build an implication is to introduce an assumption.



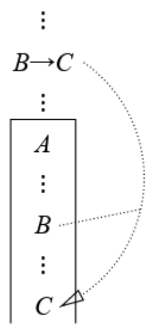
Then we apply the double negation introduction (figure on the left). Then we apply modus tollens (figure in the center) and finally we close the box and we obtain $y \to \neg\neg x$. y comes from the assumption, so given the assumption y we obtain $\neg\neg x$. it is the case that y implies $\neg\neg x$. This is what we proved.

Semantics of the assumption
The idea is that everything in the "line" is a consequences of the premises. The formulas that we obtain are not true in general but are only true given the premises. Premise is something that we started from. The assumption is the first formula in a box. The point is that, everything that we write is not true in general but

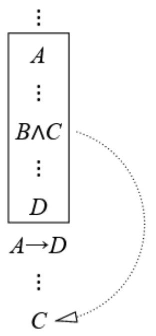
it is a consequence of the premise.

Why?
Because we just drawing a conclusions of the premises according to propositional logic, so everything is a consequence of the premise. Additionally, everything that it is inside of the box is a consequences not only of the assumption but also of the premises. For instance, in the previous example $\neg\neg y$ is a consequence of both premises and assumption. But the formula outside of the box $y \to \neg\neg x$ is only a consequence of the premises. Basically natural deduction works by drawing consequences of the premises if we are outside the box and of the premises and assumption if we are inside of the box. We keep this condition true is an invariant of the mechanism, it is always true. Even this condition is true, it also remains true because we apply rules that are correct according to this rule.

One possible mistake



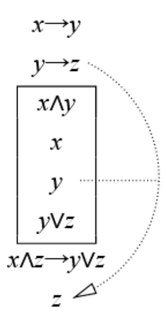
This is perfectly valid. So we can use a formula inside the box and a formula outside the box to obtain obtain something which is inside the box.
Why this?
Because C, actually, obeys the rule because we can write a formula inside the box if it is a consequence of the premises and the assumption.
It is the case?
Yes, because B is a consequence of A and the premise $B \to C$. C is a consequence of both assumption and premise, so, t follows from both A and original $B \to C$.



We cannot, instead, write it outside the box. **Why this?**
Because $B \wedge C$ is only true given A and premises and write C outside means that C is true given premises without assumption A and this is not valid. What we write outside the box should be only a consequence of the premises, instead, $B \wedge C$ is only true given A. When we retract the assumption, so when we say that assumption is not something that we want to assume any longer we can write $A \to C$. The only way to use a formula inside the box outside is by this simple step, so use the whole box to derive the implication of the last, in this case $A \to D$.

Example - Violation condition

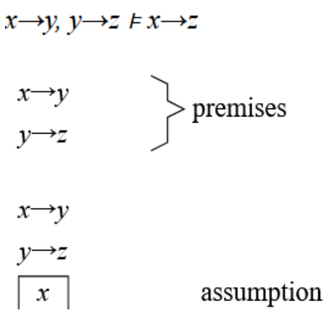
$x \to y,\ y \to z \vDash z$

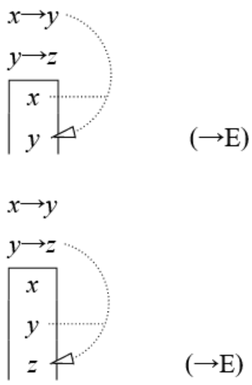


(→E)

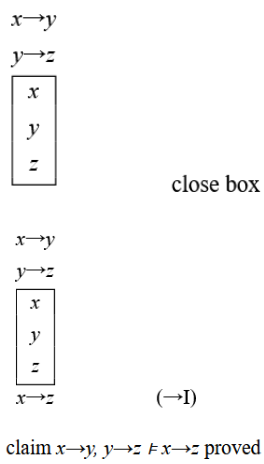
We can conlude that $x \to z$ but not z alone. If we do not obey the rule about the box, z alone is something that we can prove. This meaning that Natural deduction is incorrect if we do not apply box's rule.
We can assume $x \wedge y$ and until $x \wedge y \to y \vee z$ Natural deduction is correct. Now we make a mistake considering that $\to z$ and y and we apply modus ponens.
What's wrong with this implication?
The proble is that we know that y implies z but y is only true by $x \wedge y$ and not in general. This $x \wedge y$ is not a premise, it is a random assumption. So, z is true give $x \wedge y$ and not given the premises.

Example - Correct



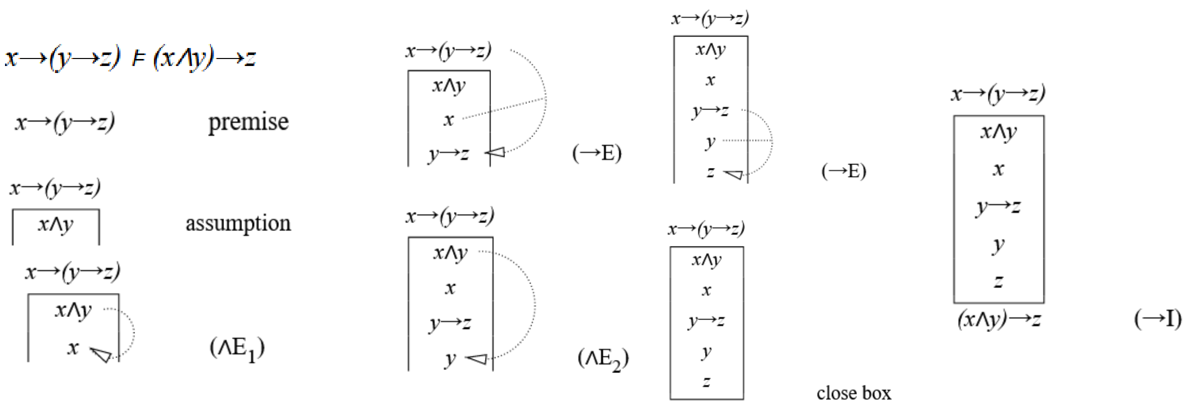


We assume x and we apply modus ponens so we obtain y in the box. This is correct at this time because if we have $x \to y$ and we assume x we conclude y as a consequence of the assumption, it is still inside the box (figure on the top right). Then again modus ponens with y and $y \to z$ and conclusion z is inside the box for the same reason as before (figure on the bottom right).

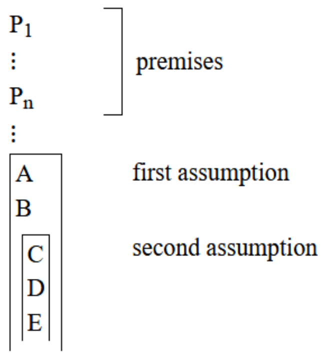


We close the box ans we have the conclusion and derive that given premises we derive $x \to z$. This is the claim and we stop.

Example - Complicated



Nested Boxes

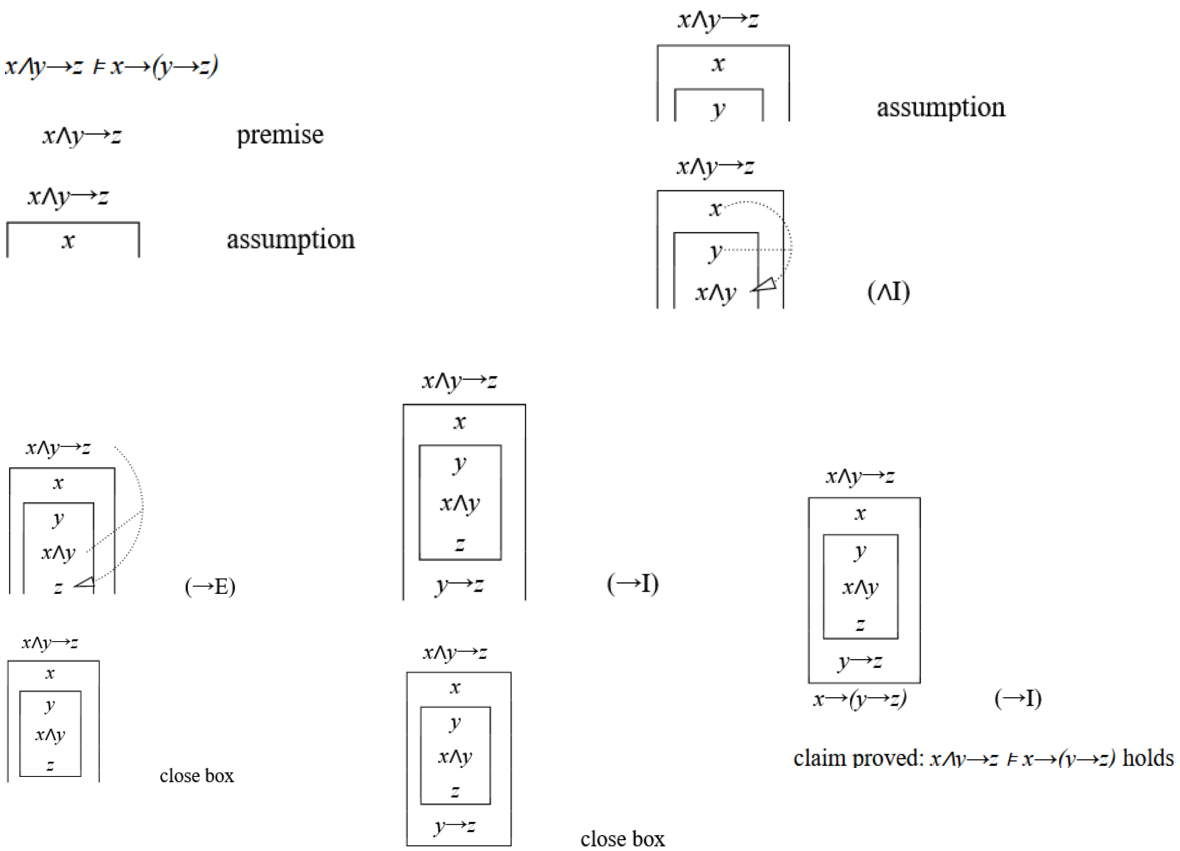


We cay say that at some point I have my set of premises and I work with premises, I realize that it is not enough and I open an assumption but I realize that I need another assumption inside the box, so I do that another assumption, so open a box inside the box. All formula inside this external box are consequences of the premises (A, B). What inside both boxes are consequences of the premises, A and C.

Semantic of the boxes

Every formula that it is inside the box is a consequence of the premises and all assumptions of all the boxes in which the formula is in. Not of any box is not inside, noly the boxes in which the formula is inside.

Example - Nested Boxes



The aim is to obtain a formula from a set of given formulas

How we do that?

We need to by this formula unless the formula is already a subformula of something that we already have. Now when we build, we build a conjunction by building its part, we build a disjunction by obtaining either the two parts and then we build an implication opening a box. If we have for example:

$$a \wedge (b \to c) \quad |= \quad b \to c$$

We do not need an assumption because the formula that we want to prove it is just a consequence of conjunction. Only if we do not have already the formula we open an assumption. Other than this, there is no specific policy

of the application of the rule, so, it is not easy to automate.
A part from that I can introduce double negation as long as I want and I can introduce disjunction as long as I want, I can also introduce arbitrary formula as assumption as long as I want. This is different because, for example, in DPLL if we choose the wrong branching literal the only problem is that we are running longer than needed. In propositional tableau method if we choose the wrong policy application of the rule the tableau may become longer than necessary but it still ends at some point. Instead, in Natural deduction, even the propositional case, the wrong application of the rule may leads to an infinite sequence.

2 First-order Natural Deduction

We have free variables and we consider them as universal quantifier.

Rules

$$\frac{\forall x\, A}{A[t/x]}\,(\forall E) \qquad \text{where } t \text{ is a term} \qquad \frac{A[y/x]}{\forall x\, A}\,(\forall I)$$

Rules for removing (left) and for introducing (right) the universal quantifier

$$\frac{A[t/x]}{\exists x\, A}\,(\exists I) \qquad \frac{\exists x\, A \quad \boxed{\begin{array}{c} A[y/x] \\ \vdots \\ C \end{array}}}{C}\,(\exists E)$$

Rules for removing (left) and for introducing (right) the existential quantifier. For introducing we need formula and box as a premise and formula, in this case C, as a consequence.

Example

$\forall x(P(x) \rightarrow Q(x)) \wedge P(c) \vDash Q(c)$

$\forall x(P(x) \rightarrow Q(x)) \wedge P(c)$ premise

$$\begin{array}{ll} \forall x(P(x) \rightarrow Q(x)) \wedge P(c) & \\ \quad \forall x(P(x) \rightarrow Q(x)) & (\wedge E_1) \end{array}$$

Remove the conjunction by only taking the first part of it.

$$\begin{array}{ll} \forall x(P(x) \rightarrow Q(x)) \wedge P(c) & \\ \quad \forall x(P(x) \rightarrow Q(x)) & \\ \qquad P(c) & (\wedge E_2) \end{array}$$

Then I also take the second part of the conjunction.

$$\begin{array}{ll} \forall x(P(x) \rightarrow Q(x)) \wedge P(c) & \\ \quad \forall x(P(x) \rightarrow Q(x)) & \\ \qquad P(c) & \\ \quad P(c) \rightarrow Q(c) & (\forall E) \end{array}$$

Given formula for removing the universal quantifies the idea is that if I have a formula and this is quantified I can just replace x with whathever I want and remove the quantifier

$$\begin{array}{ll} \forall x(P(x) \rightarrow Q(x)) \wedge P(c) & \\ \quad \forall x(P(x) \rightarrow Q(x)) & \\ \qquad P(c) & \\ \quad P(c) \rightarrow Q(c) & \\ \qquad Q(c) & (\rightarrow E) \end{array}$$

We can now apply modus ponens and claim is proved.
$\forall x(P(x) \rightarrow Q(x)) \wedge P(c) \quad |= \quad Q(c)$

3 Hilbert Systems

Hilbert's systems are basically like natural deduction, in fact, they came even earlier. The idea is that I simply make implications. I start from a set of premises and derive conclusions from them, only in this case I do not have the assumptions but I have axiom schemes (or axioms or rules of substitution). Therefore, I have mechanisms that allow me to obtain all possible deductions even in the case that I would not normally have to use the assumptions of the natural deduction. For the rest, they are quite similar.
In the case of natural deduction I start from the premises and simply have rules. Instead, in the case of Hilber's systems I have these axioms which are valid formulas; they are not all valid formulas because obviously they are infinite but they are some valid formulas which however are sufficient to obtain all the others (always by deduction).

axioms

1. $x \to (y \to x)$
2. $(x \to (y \to z)) \to ((x \to y) \to (x \to z))$

rules

$$\frac{A \quad A \to B}{B} \text{ modus ponens}$$

$$\frac{A}{A[B/x]} \text{ substitution } (x \text{ variable, } B \text{ formula})$$

These are two possible axioms that use implication only. Therefore, using this system it is possible to obtain all the derivations but only with formulas constructed with the implication. If a formula is made up of implications only, starting from these two axioms, I get it.
I start from the axioms and can only use these two rules: modus ponens and substitution. Substitution tells me that if I have a formula and this formula contains a variable x, I can put any formula B in its place. However, the substitution must be uniform, so every occurrence of x must be replaced by B.

A proof is always a sequence of formulas like natural deduction but there are no assumptions, so there are no boxes. Here, I can always start from an axiom and then apply the rules. Hence, there is an implicit rule which says that without premises I can directly add axioms.

Example

prove that $x \to x$ holds

I prove without premises that $x \to x$.

1. $(x \to (y \to z)) \to ((x \to y) \to (x \to z))$	axiom 2
2. $(w \to ((w \to w) \to w)) \to ((w \to (w \to w)) \to (w \to w))$	from 1. by substitution $[w/x,\ w \to w/y,\ w/z]$
3. $x \to (y \to x)$	axiom 1
4. $w \to ((w \to w) \to w)$	from 3. by substitution $[w/x,\ w \to w/y]$
5. $(w \to (w \to w)) \to (w \to w)$	from 2. and 4. by modus ponens
6. $w \to (w \to w)$	from 3. by substitution $[w/x,\ w/y]$
7. $w \to w$	from 5. and 6. by modus ponens
8. $x \to x$	from 7. by substitution $[x/w]$

I start from axiom 2 and at this point the substitution takes effect and in place of x I put w. In place of y I put $w \to w$ and in place of z I put w. At this point I apply axiom 1 and again I make the substitution where in place of x I put w and in place of y I put $w \to w$.
Why all this?
Because in this way I put myself in a position to have the same formula in 2 and in 4. So I can apply modus ponens and I get the consequence of the implication given by 2. At this point I make a further substitution, in which I take formula 3 and in place of x I put w and in place of y I put w. At this point I reapply modus ponens between 5 and 6. Now instead of w I put back the x and I get what I wanted to achieve. I get that $x \to x$ is a propositional calculus theorem. If I wanted, I could use Hilbert for the deduction but Hilbert's system is used to obtain theorems. Obviously the problems I had in natural deduction here get even worse because if I didn't know what rules to apply before (I didn't have guidelines) here it becomes even more obscure and this is why modern systems use more specific methods, for example, in the propositional case they use local search or enhanced backtracking (DPLL etc). Instead, in the case in which this thing is not possible but symbolic reasoning is needed, tableau are generally used, which at least you know what the rules to use are.
Hilber's systems have some advantages including that I could at least read the proof from a sequence of single operations; it is not that it becomes a tree or even a line. For some esoteric logics, Hilbert's systems are the

only known methods of automatic deduction; and in theory, they are the best systems as far as the length of the proof is concerned. That is, Hilber's systems admit the shortest proofs.

What do I mean by proof?
In DPLL the proof is the recursive call tree, in the case of the tableau it is the tableau tree, in the case of the natural deduction it is the line of the natural deduction in the case of all the boxes, finally in the case of the Hilbert systems a proof is a sequence of formulas. Now, it can be shown that the best lengths of proofs are Hilbert systems. The concept is that if I have a proof with tableaus then there is a proof with Hilbert systems of the same length or less. There is not a single Hilbert system but there are several. For example, instead

3. $\bot \to x$
4. $x \to T$
5. $\neg\neg x \to x$
6. $x \to (\neg x \to y)$
7. $(x \wedge y) \to x$
8. $(x \wedge y) \to y$
9. $(x \to z) \to ((y \to z) \to ((x \vee y) \to z))$

These are the axioms for top, bottom, negation, and and or.

of using a single axiom in substitution, another of the methods uses patterns of axioms which basically means that I can apply substitution only or directly to axioms. When I put the axiom I can do substitution together and the substitution however I can no longer use it.

see (Fitting):

- only $\uparrow$
  ("nand": $x \uparrow y = \neg(x \wedge y)$)
- rule:
  $$\frac{x \quad x \uparrow (y \uparrow z)}{z}$$
- axiom scheme: $(x \uparrow (y \uparrow z)) \uparrow (((w \uparrow z) \uparrow ((x \uparrow w) \uparrow (x \uparrow w))) \uparrow (x \uparrow (x \uparrow y)))$

There are some methods like this that use only one connective, nand (not with conjunction), and one rule and one axiom scheme.

4 Sequent Calculus

Sequent calculus are manipulations of objects, which are pairs of sets of formulas. A sequent is a set that implies another set $\Gamma \to \Delta$, where I am saying that the conjunction of the formulas in Γ implies the disjunction of the formulas in Δ. So I will write something like $A, B, C| - D, E$ and the meaning of this thing here, contrary to what one would expect, is not that A, B, C imply D, E but that $A \wedge B \wedge C| - D \vee E$.
Practicing in this system I have a manipulation of these sequents. I start from the sequents and put them together.

Rules

axioms: $\dfrac{}{x \vdash x}$ $\quad \dfrac{}{\bot \vdash}$ $\quad \dfrac{}{\vdash \top}$

structural rule: $\dfrac{\Gamma_1 \vdash \Delta_1}{\Gamma_2 \vdash \Delta_2}$ if $\Gamma_1 \subseteq \Gamma_2$ and $\Delta_1 \subseteq \Delta_2$

negation rules: $\dfrac{\Gamma \vdash \Delta, X}{\Gamma, \neg X \vdash \Delta}$ $\quad \dfrac{\Gamma, X \vdash \Delta}{\Gamma \vdash \Delta, \neg X}$

conjunction rules: $\dfrac{\Gamma, X, Y \vdash \Delta}{\Gamma, X \wedge Y \vdash \Delta}$ $\quad \dfrac{\Gamma \vdash \Delta, X \quad \Gamma \vdash \Delta, Y}{\Gamma \vdash \Delta, X \wedge Y}$

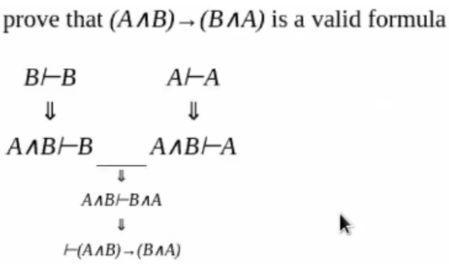
disjunction rules: $\dfrac{\Gamma, X \vdash \Delta \quad \Gamma, Y \vdash \Delta}{\Gamma, X \vee Y \vdash \Delta}$ $\quad \dfrac{\Gamma \vdash \Delta, X, Y}{\Gamma \vdash \Delta, X \vee Y}$

implication rules: $\dfrac{\Gamma \vdash \Delta, X \quad \Delta, Y \vdash \Delta}{\Gamma, X \to Y \vdash \Delta}$ $\quad \dfrac{\Gamma, X \vdash \Delta, Y}{\Gamma \vdash \Delta, X \to Y}$

So I have rules that given sequents allow me to get other sequents. Here the formulas are made like the natural deduction, that is, I have several premises and a single conclusion, but they are always sequents.

The prof is an "inverted" tree, it is always a tree but it is built from the leaves.

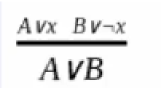
Example



I'm trying to prove that $(A \wedge B) \rightarrow (B \wedge A)$. The proof is not a very simple tree. There is an axiom that says that I can always introduce a sequent of the type A |− A Then there is a second rule that tells me that if I have a sequent I can add "pieces", so I can add a conjunction. Then there is another rule that tells me that if I have two sequents that have the same thing, ie two sequents with the same premise, then I can put them together. Latest rule says that if I have a sequent then I can bring to the right what is on the left (deduction theorem). All this leads to the validity of this formula.

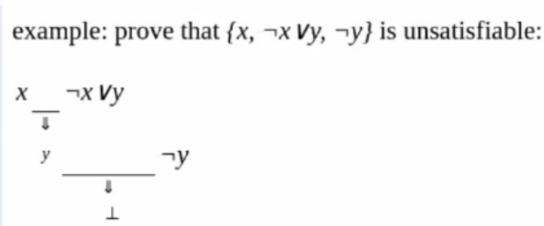
5 Resolution

Resolution is a refutation method, a set of clauses. As when there was DPLL, a clause is or of literals.
I have a set of these clauses, which are in conjunction with each other. What I do in the resolution is to take these clauses and prove a contradiction. If I want to prove satisfiability I apply the resolution and apply it as much as possible and if I can't, it means that the formula is satisfiable. If I want to prove an implication, I show that the pluses and negatives of the consequence are unsatisfiable.



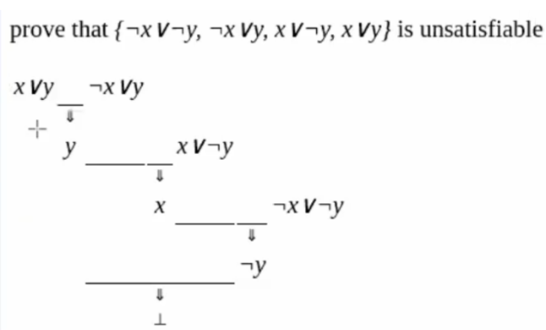
The method is based on a single rule, if I have two clauses with a variable with different sign then I can combine them by taking the clause which is the union without the variable which is negated on one side and directed on the other.

Example



The proof that this formula is unsatisfiable.

Another example



In the first step we have x and $\neg x$, so we take only y. Then we have y and $\neg y$ so we combine formula and obtain only x. We do this until we reach $\perp$.

5.1 Resolution in FOL

Clause in first order logic

$$\forall x_1 \dots \forall x_m \, L_1 \lor \dots \lor L_n$$

where L_i's are literals

no free variables in the clause (all quantified)

The clauses are a bit different. L_i are literals, where literals are single predicates applied to a set of terms $P(c, f(x))$ or the negation $\neg Q(d)$. All these if I do it now and I put universal quantifiers I get the clause in FOL.

Since the clauses always have all the quantifiers of all their variables, these quantifiers can be omitted, it is a convention.

Resolution rule

A, B clauses

L_1, L_2 literals

$$\frac{A \lor L_1 \qquad B \lor \neg L_2}{(A \lor B\delta)\sigma}$$

In the resolution rule I have a clause that contains a literal and a clause that contains another literal, where there are two literals such that if I rename the variables of the second clause and there is a unification of L_1 with $\neg L_2$ then I can apply the rule.

The rule is a bit complicated, why?

Because in the case of the propositional I have variable and the negated of the variable; instead in FOL I have universal quantifiers and variables. This means that instead of the variable I can put anything, so maybe two literals are not exactly the negated one of the other but in some cases they can be because if any x is equal to that other literals that maybe is $\neg x$ then they become equal. However, then I have to bring the specific case to a conclusion. In addition, there is the problem that variables can be given the names they want so maybe two literals are not the opposite of each other just because they have different names.

Factoring rule

A clause

L_1, L_2 literals

$$\frac{A \lor L_1 \lor L_2}{(A \lor L_1)\sigma}$$

where:

- σ is a MGU of L_1 and L_2

next: why δ?

why σ?

The factoring rule, also in this case, may be that a clause contains two literals that are equal, and in the propositional case two equal are 1; instead in the FOL it can happen that two literals are not identical but have a case in common, that is there is a certain subcase of these universals in which they coincide, in that subcase I can simplify the clause by merging the two literals into one.

$$P(x) \lor \neg Q(f(z), y)$$
$$R(z, f(b)) \lor Q(x, y)$$

I have two clauses and i am wondering if i can enforce resolution. If I used the purely propositional rule I would say yes because in one I have Q and in the other $\neg Q$. However, there is a difference, they are not exactly identical because one has f(z) and the other has x. So these two literals almost coincide but are not quite the same. If these were two different constants then they are just different, but in this case this z and this x I have not put the universal quantifiers but in reality there are. So the second clause will be true even if I substitute f(z) for x, so in particular cases these two clauses are really in opposition.

When I apply the substitution I know basically saying that I am doing a subcase of the universal quantifier. What used to be for any value, now I'm saying for any f(z).

6 Proof complexity

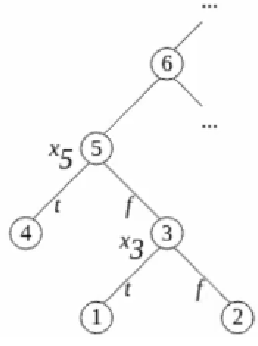
Proof complexity is not a method. In all this discretion of automatic reasoning we have so far seen the single methods (unit propagation, GSAT, DPLL, etc), in particular, the complete ones are: backtracking, DPLL, tableau, natural deduction, Hilbert systems, sequents calculus and resolution. These are only the complete methods, the ones that allow us in one way or another to prove the unsatisfactory of the formulas.

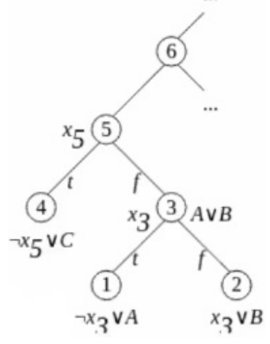
Which of these methods is the best?
Clearly there is an experimental evaluation. Experimentally from all these systems what emerges in the propositional calculus is DPLL, it is the most used system among the complete ones. Instead in the FOL the tableau is used because it is not possible to use the descriptive logics because the problem consists in the fact that backtracking, DPLL can only go to use them when these variables have specific values, such as 0, 1. In the other logics, different from the propositional ones, the domain is not determined, it is not fixed therefore the values to be considered are potentially infinite. So I can't backtrack, I have to use tableau.
Proof complexity makes another type of evaluation, a more theoretical evaluation, it says which one is potentially the best. The one that admits the shortest proofs. In the case of the DPLL we have the tree with the recursive calls, in the case of the tableau we have the tableau, in the case of resolution the number of clauses I get for the resolution, Hilbert the length of the proof and so on. There is no question of time, but deciding which is the system that admits the shortest proofs and which therefore, theoretically, would be shorter if someone told me which rules to apply. Among other things, in this case the winners are the Hilbert systems, the ones that are in practice the least efficient, those that admit potentially infinite proofs. They admit many proof, including some, which are also very short. A sort of inverse hierarchy therefore occurs in which efficient systems according to proof complexity are the least efficient in practice.

DPLL
    the tree of recursive calls
tableau
    the tableau (the tree of formulae)
natural deduction
    the sequence of formulae, and the boxes
Hilbert systems
    the sequence of formulae
resolution
    the DAG of clauses
    (root is $\perp$, leaves the clauses of the set to prove unsatisfiable)

The basic concept is that of p-simulation, so method A p-simulates B if each proof of B has a corresponding in A of comparable size, apart from a polynomial increment. For example, I take resolution as A, Hilbert and B. So Hilbert p-simulates resolution means that if I take a formula and a proof in resolution then a proof in Hilbert of the same formula with comparable length (polynomial size).

Example - resolution p-simulates backtracking
For example, one of the simplest proofs is to show that resolution p-simulate backtracking, so if I take a formula and do the backtracking tree, which proves that it is unsatisfiable, then it can be shown that there is a resolution proof that has the same size.





Suppose we have a resolution tree made like this (figure left), practically a resolution tree is a tree that starts from nothing and chooses a variable and does "true" or "false" and eventually arrives at a set of formulas that are all unsatisfiable . The clause that is false in 1) and the clause that is false in 2) have become false precisely because the last variable is true in one branch and false in the other. This means that I have two clauses that have exactly one opposing literals and therefore these two resolve. So it can be shown in some way that I build and eventually go up this tree by getting the empty clause in the root, hence the contradiction (a part of reconstruction in the figure on the right). So given the backtracking, the recursive call tree I can build a proof for the resolution.

What is this for?
Obviously not in practice because once I have proved that the formula is unsatisfactory, I don't start building the resolution tree. But this is a completely theoretical discourse. The proof is that for every backtracking tree there is a proof of the resolution that has the same size, more or less. This shows that as length of proofs resolution is better than backtracking because any backtracking proof can be reproduced in resolution. Instead, it can be shown that the opposite does not happen.