

Contents

1	The Beginning	3
1.1	Russell's Paradox	3
1.2	Enters Formal Logic!	9
1.3	A look <u>back</u> at strings	15
1.3.1	A Bad Alphabet	17
1.4	The Formulas or well-formed-formulas (wff)	25
2	Properties of the wff	35
2.1	Boolean Wff	35
2.2	Boolean Semantics	60
3	What makes our Logic “Classical”	65
3.1	States and Truth tables	65
3.2	Finite States	71
3.3	Tautologies and Tautological Implication	74
4	Substitution and Schemata	81
4.1	Rules and Axioms of Boolean Logic	91
4.2	Equational Proofs	121
5	A Weak Post's Theorem and the Deduction Theorem Retold	161
5.1	Soundness of Boolean Logic	162
5.2	Completeness of Boolean logic (“Post's Theorem”)	170
5.3	Deduction Theorem and Proof by Contradiction	185
6	Resolution	191
7	Predicate Logic	197
7.1	The language of First-Order Logic	200
7.2	Axioms and Rules for Predicate Logic	231
7.3	First-order Proofs and Theorems	233
7.4	Deduction Theorem	236
7.5	Adding (Removing) “ $(\forall \mathbf{x})$ ” to (from) the beginning of a wff.	239
7.5.1	Examples	247
7.5.2	A Few Memorable Examples	248

7.6	Weak Leibniz for 1st-Order Logic	253
7.7	Ad hoc Memorable Examples	270
7.8	Adding and Removing the Quantifier “ $(\exists x)$ ”	274
7.9	Interpretations	289
7.10	Soundness in Predicate Logic	306

Chapter 1

The Beginning

Sep. 4, 2024

1.1 Russell's Paradox

...or, when things (in MATH) go “sideways” ...

1.1.1 Example. (Briefly about set notation) We represent sets either by explicit *listing*,

- $\{0\}$
- $\{\$, \#, 3, 42\}$
- $\{0, 1, 2, 3, 4, \dots\}$

or by some “*defining property*”: *The set of **all** x^* that make $P(x)$ true*, in symbols

$$S = \{x : P(x)\} \tag{1}$$

As we know from discrete MATHs, (1) says the *same thing* as the statement

$$x \in S \equiv P(x) \tag{2}$$

read “for any value of x , $x \in S$ is equivalent to $P(x)$ ”

*Strictly speaking you **DON’T** collect the various shapes and colours of the letter x . There is only **ONE** x . The expression “set of all x such that $P(x)$ is true” is sloppy for “the set of all **VALUES** of x such that $P(x)$ is true”.

Why so? Because $P(x)$ is an *entrance condition*!

A value of x is included in the set S IF and *ONLY* IF (iff) said value passes the test $P(x)$.

Wait! Shouldn't I have written (2) as

$$x \in S \equiv P(x) \text{ is true} \quad (2')$$

Nope. When mathematicians state $P(x)$ for some unspecified *fixed* x they mean “ $P(x)$ is true” for that value.

Cantor believed (good grief! “believed”?! as did the mathematician and philosopher Frege) that, for any property $P(x)$, (1) defines a set—they never proved this; just “*believed*” and used (!!) in Cantorian Set Theory.

Which is neither here nor there because **they never said what a set is**. They allowed **ANY** collection of (mathematical) objects to be set!

Russell begged to differ, so he said: “Oh, yeah? How about this? Is it a ‘*set*’?”

$$R = \{x : x \notin x\}$$

where the property “ $P(x)$ ” here is “ $x \notin x$ ”

Now, by (2) we have

$$x \in R \equiv x \notin x$$

If *R* IS a set, then we can plug in it the set variable *x* above to obtain

$$R \in R \equiv R \notin R$$

Sep. 9, 2024

How do we avoid Russell's *contradiction*?

By *admitting* that *R* is *NOT* a set so we do not allow the substitution! □

1.2 Enters Formal Logic!

How do we get out out of this contradiction?

- Cantor never said what sets really are and how they are built. He just used a *dictionary of synonyms* instead of a definition! (collection, class, aggregate, etc., he suggested as synonyms.)
- MUST use logic to argue how sets behave and what they are.



WE, however in 1090A, will never deal with sets again. So *we leave the task to set theorists* ([Jec78, Lev79, Tou03b]) or to EECS 1028E and Z (my other course).

We only felt we should motivate the case for Logic at the expense of Cantor's approach to Set Theory.



So Cantor was sloppy about what a set is

AND how sets get formed.

Formal—*meaning SYNTACTICALLY PERFORMED; Based on FORM*—**logic** was invented by Russell and Whitehead, and **Hilbert** to *salvage* Mathematics from “antinomies” and “paradoxes”, both words derived from Greek, and both meaning **contradictions**.



How does *formal logic salvage* Mathematics?

By helping you *stay on track* in your argumentation.

You cannot pull facts and *fake facts off the air*, but your facts *MUST* be *axioms* or *PREVIOUSLY proved* theorems, and the *rules* of logic that you use

MUST *NOT* DEPEND *on an Interpretation* (yours, Cantor’s, or mine).



Connection of Formal Logic with Programming

- (1) In programming we use *syntactic rules* to write a *program* in order to solve some problem computationally.
- (2) In logic you use the syntactic rules to write a *proof* that establishes a theorem.

Kinds of logic reasoning that we will thoroughly examine and use in this course.

1. Equational logic —also known as *calculational logic*.

Introduced by [DS90] and simplified by [GS94] and later by [Tou08] to make it accessible to undergraduates. Software Engineers like it.

2. Hilbert-style logic. This is the logic which most people use to write their mathematical arguments in **publications**, **lectures**, etc.

Logic is meant to certify mathematical truths syntactically.

Logic is normally learnt by

- A *LOT* of practice.
- By presenting and teaching it gradually, namely
 1. First, learning the *Propositional Logic* (also known as *Boolean Logic*).

Here one learns how logical truths *combine* using *connectives* familiar from programming like OR, AND, and NOT.

Boolean logic is *not expressive enough* to formulate statements about *mathematical objects*. Naturally, if you cannot ask it—a question about such objects— then you cannot answer it either.

2. Next, learning *Predicate Logic* (also known as *First-Order Logic*).

This is the full logic of the mathematician and computer scientist as it lets you formulate and explore statements that are about *mathematical objects* like numbers, strings and trees, and many others.

1.3 A look back at strings

1.3.1 Definition. (Strings; also called Expressions)

1. What is a *string* over some *alphabet* of symbols?

It is an **ordered** finite sequence of symbols from the alphabet —with no gaps or other “separators” between symbols.

1.3.2 Example. If the alphabet is $\{a, b\}$ then here are a few strings:

- (a) a
- (b) $aaabb$
- (c) $baaaa$
- (d) $bbbbbbb$

□

What do we mean by “ordered”? We mean that *order matters*! For example, *aaabb* and *baaaa* are different strings. We indicate this by writing *aaabb* \neq *baaaa*.



Two strings are equal iff[†] they have the same length n and at *each* position—from 1 to n —both strings have the same symbol. So, $aba = aba$, but $aa \neq a$ and $aba \neq baa$.



[†]If and only if.

1.3.1 A Bad Alphabet

Consider the alphabet $B = \{a, aa\}$.

This is bad. WHY?

Because if we write the string **aaa** over this alphabet we do not know what we mean by just **looking** at the string!

Do we mean 3 a like

$a \quad a \quad a$

Or do we mean

$a \quad aa$

Or perhaps

$aa \quad a$

We say that alphabet B leads to *ambiguity*.

*Since we use NO separators —like a space or a comma— between symbols in denoting strings we **MUST ALWAYS** choose alphabets with **single-symbol** items.*

2. Names of strings: A, A'', A_5, B, C, S, T .

What for? CONVENIENCE AND EASE OF EXPRESSION.

Thus $A = bba$ gives the string bba the name A .

Names vs IS: “Practicing” mathematicians and computer scientists take a sloppy attitude towards using the verb “IS”.

When they say “let A **be** a string” they mean “let A **name** a string”.



Same as in “let x **be** a rational number”. Well x is **not** a number at all! It is a letter! We mean “let x **STAND** for, or **NAME**, a rational number”



3. Operations on strings: Concatenation. From strings *aab* and *baa*, concatenation in the order given yields the string *aabbaa*.

If A is a string (meaning **names** a string) and B is another, then their concatenation AB is not a concatenation of the names but *is a concatenation of the contents*. If $A = aaaa$ and $B = 101$ then $AB = aaaa101$.

Incidentally,

$$BA = 101aaaa \neq aaaa101 = AB$$

Thus *in general concatenation is not commutative as we say*.

Why “in general”?

Well, if $X = aa$ and $Y = a$ then $XY = aaa = YX$.

Special cases where concatenation commutes exist!

4. Associativity of concatenation.

It is expressed as $(AB)C = A(BC)$ where bracketing here denotes invisible **META** symbols (*they are NOT part of any string!*) that simply INDICATE **the order in which we GROUP**, from left to right.

At the left of the “=” we first concatenate A and B and then glue C at the *right* end.

$$A \quad B \quad C$$

if $A = 1, B = 2, C = 3$ then $A(BC) = 123$ NOT $1(23)$

To the right of “=” we first glue B and C and then glue A to the *left* of the result.

In either case we did not change the relative positions of A , B and C .

The property is self-evident.

I can now skip brackets and write $(ABC)D$ and you know what I mean!

$$((((ABC)D)E)F)G$$

5. Empty string. A string with *no symbols*, hence with *length 0*. Denoted by λ .

$$\lambda Q = Q, \text{ and } Q\lambda = Q$$



How is λ different than \emptyset the empty *set*?

Well one is of *string type* and the other is of *set type*. So? The former is an **ORDERED empty set**, the latter is an **UNORDERED empty set** that moreover is *oblivious to repetitions*.

I mean, $aaa \neq a$ but $\{a, a, a\} = \{a\}$.



6. Clearly, for any string A we have $A\lambda = \lambda A = A$ as concatenation of λ adds nothing to either end.

7. Substrings. A string A is a *substring* of B iff A appears as is as a *part* of B .

So if $A = aa$ and $B = aba$ then A is NOT a substring of B . So if $A = aa$ and $B = baab$ then A IS a substring of B .

Its members both appear in B (the two a) but are *not* together as they are in A . *A does not appear “as is”.*

Can we get rid of all this bla-bla with a proper definition? **Sure:**

1.3.3 Definition. A is a substring of B iff for some strings (named) Q and R we have $B = QAR$. \square

We also say A is *part* of B .

Sep. 11, 2024

8. *Prefix and suffix*. A is a *prefix* of B if for some string V , $B = AV$.

So A is part of B up in front!

A is a *suffix* of B if for some string U , $B = UA$. \square

Example: λ is a prefix and a suffix, indeed a part, of *any* string B . Here are the “proofs” of the two cases I enumerated:

- $B = \lambda B$
- $B = B\lambda$

WHAT ABOUT THE THIRD CASE?

Well, $B = B\lambda\lambda$.

1.4 The Formulas or well-formed-formulas (wff)

The Syntax of logic. Boolean Logic at first!

Boolean logic is the “Algebra of statements”. We start with *atomic* statements and build complex statements using “glue” as I call the Boolean *connectives*

$$\neg, \wedge, \vee, \rightarrow, \equiv$$



Atomic statements have NO glue! We usually denote them by p, q, r with or without primes or subscripts.

E.g., p, q, r, r'''_{105} all are (meaning, all stand for) *un-specified* atomic statements.

Boolean logic has precisely two specific statements (“constants”). Read on!



Examples of statements that Boolean logic can express:

$p, (\neg p)$ and also $((p \vee q) \wedge r)$. And more!

Can I *see inside* atomic statements like p to see what they *mean*?

NO!! We cannot! But we can assign *arbitrarily* “true” or “false” *values* to atomic statements and then proceed to see how these *truth values* propagate when I *apply glue*.

That is *all* that Boolean logic can do.

And this ends up being quite useful! Read on!

1.4.1 Definition. (Alphabet of Boolean Symbols)

A1. *Names for variables*, which we call “propositional” or “Boolean” variables.

These are p, q, r , with or without primes or subscripts (indices) (e.g., $p, q, r, p', q_{13}, r'''_{51}$ are all names for Boolean variables).

A2. Two *symbols* denote the Boolean *constants*, \top and \perp . We pronounce them “**top**” and “**bot**” respectively.

What are \top and \perp good for? We will soon see!

A3. (Round) brackets, i.e., “(” and “)” (employed without the quotes, of course).

A4. Boolean “*connectives*” that I will usually call “*glue*”.

We use glue to put a formula together much like we do so when we build model cars or airplanes or houses.

The symbols for Boolean connectives are

$$\neg \wedge \vee \rightarrow \equiv \quad (1)$$

and are read from left to right as “negation, conjunction, disjunction, implication, equivalence”.

□



We stick to the above symbols for glue (no pun!) in this course! *Just as in programming,*



NO DEVIATION IS PERMITTED!!!



You *cannot use* any symbols you **please** or **like**.



SPEAKING BY ANALOGY, You use *THE* symbols of A Programming Language as *THEY ARE GIVEN*.

If not, your program does NOT work and your GRADE bottoms!

Same holds for logic!



1.4.2 Definition. (Formula Construction (process))

A *formula construction* (in the text called “*formula calculation*”) is any *finite (ordered) sequence of strings* *over the alphabet*[†] of Boolean logic \mathcal{V} that obeys the following three specifications:

- C1. At any step we may write precisely one symbol from categories **A1.** or **A2.** above (1.4.1), that is, *variables* (p, q'', r''' , etc.) or *constants* (\perp or \top).
- C2. At any step we may write precisely one string of the form $(\neg A)$, as long as we have written the string (*named*) A already at a previous step.

So, “ $(\neg A)$ ” is a string that has “ $(\neg$ ” (no quotes) as a prefix, then it has a part we named A , and then it has “ $)$ ” (no quotes) as a suffix.



I must stress that the letter A names the string that was written down.

Just as in a *program*: When you issue the command “**print** X ” you mean to **print what the X contains as value** —**what it names**. You do *not* mean to print the **letter** “ X ”!



- C3. At any step we may write precisely one of the strings $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \equiv B)$, as long as

[†]“Over the Alphabet”: Using exclusively symbols from the Alphabet \mathcal{V} that we adopted.

we have already written *each* of the strings A and B **earlier**. □



We do not care which we wrote first, A or B .



1.4.3 Definition. (Boolean formulas (wff)) Any string A over the alphabet \mathcal{V} (A1.–A4.) is called a *Boolean formula* or *propositional formula*—in short **wff**—*iff* A is a string that **appears** in **some** formula construction. \square

1.4.4 Example. First off, the above says more than it pretends to:

For example, it says that *every* string that appears in a formula construction is a wff. **The definition also says,**

*“do you want to know if A is a wff? **Just make sure you can build a formula construction where A appears.**”*

We normally write formula constructions **vertically**. Below I use numbering and annotation (in “ $\langle \dots \rangle$ ” brackets) to explain each step.

•

- (1) \perp $\langle \text{const} \rangle$
- (2) p $\langle \text{var} \rangle$
- (3) $(\neg \perp)$ $\langle (1) + \neg \rangle$
- (4) \perp $\langle \text{const} \rangle$
- (5) \top $\langle \text{const} \rangle$

Note that we *can* have *redundancy* and *repetitions*.

Ostensibly the only nontrivial info in the above is that $(\neg \perp)$ is a formula. But it also establishes that \perp and \top and p are formulas.

•

- (1) \perp $\langle \text{const} \rangle$
- (2) p $\langle \text{var} \rangle$
- (3) $(\neg \top)$ $\langle \text{oops!} \rangle$
- (4) \perp $\langle \text{const} \rangle$
- (5) \top $\langle \text{const} \rangle$

Why the “oops”? The above is wrong at step (3). I have not written \top in the construction *before* I attempted to use it!

□

Chapter 2

Properties of the wff

Sep. 16, 2024

Here we **speak** **about** wff —and discover some useful properties they have— before we get to our main task, eventually, of **USING** wff *in proofs*.

2.1 Boolean Wff

Let us repeat

2.1.1 Definition. (Boolean formulas or wff) *A string (or expression) A over the alphabet of Boolean symbols \mathcal{V} is called a Boolean formula or a Boolean well-formed formula (in short *wff*) iff it occurs in some formula construction.*

The set of all wff we denote by the all-capitals **WFF**.

The wff that are either propositional variables

$$p, q, p'', r_{123}, \dots$$

or \perp or \top , in short, *glue-less*, we call Atomic wff.





Notation. META names. We often want to say things such as “...bla-bla ... *all* variables p ...”.

► Well this is not exactly right! **There is only ONE variable p !**

We get around this difficulty by having *informal names* (in the *metatheory* as we say) for Boolean variables: $\mathbf{p}, \mathbf{q}, \mathbf{r}'$, etc.

Any such bold face informal variable can stand for *any* actual variable of our alphabet \mathcal{V} whatsoever.

So “all variables \mathbf{p} ” means “any of the actual variables $p, q, r_{1110001}, \dots$ that \mathbf{p} may stand for” while “all p ” is *meaningless!*



We can give a definition of formulas that is *independent* from formula constructions: OK, the above Definition 1.4.3 says that *A* is a wff iff it appears in a construction as

1. Atomic: \perp, \top, \mathbf{p}
2. A negation $(\neg B)$, where B appeared earlier in the construction *so is a wff*
3. An expression $(B \wedge C)$ or $(B \vee C)$ or $(B \rightarrow C)$ or $(B \equiv C)$, where B *and* C appeared earlier in the construction *so are wffs*
and



BUT we can say “*B* (or *C*) Appeared EARLIER” differently:

“*B* (or *C*) is a wff”



So,

2.1.2 Definition. (The Inductive Definition of wff)

An expression A over \mathcal{V} is a wff just in case A is:

(1) Atomic (\mathbf{p}, \perp, \top)

or one of

(2) $(\neg B)$, $(B \wedge C)$, $(B \vee C)$, $(B \rightarrow C)$, $(B \equiv C)$, *where B and C are wff.* \square

2.1.3 Remark. The formulas $(\neg A)$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \equiv B)$ are pronounced **in English**, from left to right, “not A ”, “ A and B ”, “ A or B ”, “if A then B ” (but also “ A implies B ”), “ A is equivalent to B ”.

$$\begin{array}{ccc} & \text{last glue} & \\ & \downarrow & \\ ((A \rightarrow B) & \vee & C) \end{array}$$

The wff in the remark have the same names as their “last glue”, namely, *negation, conjunction, disjunction, implication and equivalence*.

Pause. Why did I say “**LAST**” glue?





2.1.4 Example. Using 1.4.3 let us verify that $((p \vee q) \vee r)$ is a wff.

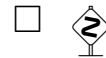
Well, here is a formula construction written with annotations:

- | | | |
|-----|-----------------------|--|
| (1) | p | $\langle \text{atomic} \rangle$ |
| (2) | q | $\langle \text{atomic} \rangle$ |
| (3) | r | $\langle \text{atomic} \rangle$ |
| (4) | $(p \vee q)$ | $\langle 1 + 2 + \vee\text{-glue} \rangle$ |
| (5) | $((p \vee q) \vee r)$ | $\langle 4 + 3 + \vee\text{-glue} \rangle$ |

Do we have to write down *all* the atomic wff at the very *beginning*? *Not really*, but it is important to write them BEFORE they are *used* in the construction!

So, this works too:

- | | | |
|-----|-----------------------|--|
| (1) | p | $\langle \text{atomic} \rangle$ |
| (2) | q | $\langle \text{atomic} \rangle$ |
| (3) | $(p \vee q)$ | $\langle 1 + 2 + \vee\text{-glue} \rangle$ |
| (4) | r | $\langle \text{atomic} \rangle$ |
| (5) | $((p \vee q) \vee r)$ | $\langle 4 + 3 + \vee\text{-glue} \rangle$ |



Intuitively, immediate predecessors of a wff are the formulas on which we applied the **last glue**.

2.1.5 Definition. (Immediate predecessors (i.p.))

1. No atomic formula has immediate predecessors.

2. Any of the following wff

$(A \wedge B), (A \vee B), (A \rightarrow B), (A \equiv B)$

has as i.p. A and B .

3. A is an i.p. of $(\neg A)$.

□

2.1.6 Example.

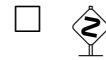
- The i.p. of $((p \vee q) \vee r)$ are $(p \vee q)$ and r
- The i.p. of $(p \vee q)$ are p and q
- The only i.p. of $(\neg \top)$ is \top

□



2.1.7 Remark. (Priorities of glue (connectives)) The *priorities* of glue, from left to right in (1) below, go from *strongest* to *weakest*.

$$\neg, \wedge, \vee, \rightarrow, \equiv \quad (1)$$



Why do we care? What does “priority” do?

Well, *suppose* we do not want to always write wff down with all the brackets that Definitions 1.4.3 and 2.1.2 require.

Why wouldn't we? For better readability!



Thus we **agree** to judiciously omit brackets in a manner that *we can reinsert them correctly* if we are required to!



That is, we **agree** on how to write formulas *sloppily* and *get away with it!*

Is there any other way to agree on priorities?

Yes, **BUT**: As it is with **any agreement between any two parties**, there can be ONLY ONE agreement.

Remember. We are learning a “programming” language!!

So please *do* follow (1) above and the clarifications that follow below. **Anything else will be wrong.**



The “algorithm” is that whenever two pieces of glue compete for a variable as in, *for example*,

$$\dots \vee p \wedge \dots$$

then the *stronger glue wins* (higher priority). In this case it is \wedge that wins and “gets” the p .

This means brackets were intended —and hence are reinserted— this way:

$$\dots \vee (p \wedge \dots$$

What if we have the situation

$$\dots \vee p \vee \dots \tag{2}$$

i.e., same glue left and right of p ?

We have the agreement that all glue is right-associative, that is, in a chain like (2) *the glue on the right wins!* We insert brackets this way:

$$\dots \vee (p \vee \dots$$

In particular

$$\neg\neg\neg p$$

means

$$\left(\neg(\neg(\neg p)) \right)$$

$$p \rightarrow q \rightarrow r \rightarrow \perp$$

means

$$(p \rightarrow (q \rightarrow (r \rightarrow \perp)))$$

In $(p \rightarrow q) \rightarrow r$ cannot remove the brackets; all are needed.

2.1.8 Definition. (Complexity of a wff) The *complexity* of a wff is *the number of occurrences* of connectives (glue) in it. Counting occurrences means that *multiplicity matters* and counts! \square

2.1.9 Example. Clearly we can compute complexity correctly whether we wrote a formula with all its brackets or not.

For example, the complexity of $p \rightarrow \perp \rightarrow r$ is 2 whether we wrote it with no brackets or wrote it as Definitions 1.4.3 and 2.1.2 want: $(p \rightarrow (\perp \rightarrow r))$.

Directly from the definition above, every atomic formula has complexity zero. \square



All the theorems (and their corollaries) in this section are **ABOUT** formulas of Boolean logic, and their *FORM*.

They are not *theorems OF* Boolean logic. This concept we have not defined yet!!

Theorems that are ABOUT Boolean logic we call **META**theorems.



2.1.10 Metatheorem. *Every formula A has equal numbers of left and right brackets.*

Proof. Induction on the *complexity*, let's call it n , of A .

1. *Basis.* $n = 0$. Then A has no glue, so it is atomic.
But an atomic formula has no left or right brackets!

Since $0=0$ we are good!

2. *Induction Hypothesis*, in short “I.H.” Fix an n and assume the statement for all A of complexity $\leq n$.

3. *Induction Step*, in short “I.S.”, is for any A of complexity $n+1$. As $n+1 > 0$, A is NOT atomic **THEREFORE** it has one of *TWO* forms:

- (a) A is $(\neg B)$ —where B is a wff.

By I.H. —applicable since A has complexity $n+1$ hence the complexity of B is $\leq n$ — B has equal number of left and right brackets. Forming A we added one left and one right. So, total left=total right for A .

- (b) A is $(B \circ C)$, where we wrote “ \circ ” as a *metasymbol* that stands for any *binary glue* among

$$\wedge, \vee, \rightarrow, \equiv$$

By I.H.

$$\text{Blefts} = k, \text{Brights} = k, \text{Clefts} = k', \text{Crights} = k'$$

So, after gluing,

$$\text{BandClefts} = k + k', \text{BandCrights} = k + k'$$

Overall (after adding external brackets for A),

we have $k + k' + 1$ lefts and $k + k' + 1$ rights.

Bingo!

□



IMPORTANT! You will note that the induction for the formula A above essentially went like this:

- Prove the property for the atomic formulas \mathbf{p}, \perp, \top

Then we assumed the I.H. that all the i.p. of A have the property.

and we proved (I.S.)

- If A is $(\neg B)$, then A has the property *since the i.p. B does* (**WHY B does?**
BY I.H. on i.p.)

The technique above is called **Induction on** (the *shape of*) **formulas** and **does not need the concept of complexity**.

This is how we will do it *in our inductions going forward*.



2.1.11 Corollary. *Every nonempty proper prefix of a wff A has an excess of left (compared to right) brackets.*

Proof. I will do induction on formulas A .

- *Basis.* A is atomic. Then we are done since A has NO nonempty proper prefix!

People also say “**then there is nothing to prove**”
or “**the statement is vacuously satisfied**”.

⚡ What just happened here?! Well, I am claiming “**the statement is true**” and suppose that you are claiming “**the statement is false**”.

It is for you to give me a *counterexample* to what I said in order to show that you are right: Namely,

You must produce a *nonempty proper prefix* of A that fails the property.

BUT there is no way! There is NO *nonempty proper prefix* of A !

So I win!



- *Assume* the I.H. that *all the i.p. of A have the property.*

- For the I.S. we examine *ALL possible forms* of nonempty proper prefixes. These are:
 1. Case where A is $(\neg B)$. A nonempty proper prefix of A has one of the four **forms** below:
 - (a) $($ Then clearly we have an excess of “(
The I.H. was NOT needed.
 - (b) $(\neg$ Then clearly we have an excess of “(
The I.H. again was NOT needed.
 - (c) $(\neg D$, where D is an nonempty proper prefix of B . D already has an excess of “(
by the I.H. that applies since B is an i.p. of A .
So, adding to them the leading red “(
no harm!
 - (d) $(\neg B$ Now (2.1.10) B has equal number of lefts and rights. The leading (red)“(“ contributes an excess. The I.H. again was NOT needed.

So C' has an excess of “(” and the leading “(” of A helps too.

red “(” breaks the balance in favour of “(”.



This is easy:

2.1.12 Theorem. *Every formula A begins with an atomic wff, or with a “(”.*

Proof. By 2.1.2, A is one of

- Atomic \mathbf{p}, \perp, \top
- $(\neg B)$
- $(B \circ C)$ where $\circ \in \{\wedge, \vee, \rightarrow, \equiv\}$

So, in the first case A begins with an atomic wff, and in the other two begins with an “(”.

No Induction was used or needed!

□

Sep. 18, 2024

2.1.13 Theorem. (Unique Readability) *The i.p. of any formula A are unique.*



So we can “**deconstruct**” or “**parse**” a formula in a unique way:

A formula is **exactly one of** atomic, a negation ($\neg B$), a disjunction ($B \vee C$), a conjunction ($B \wedge C$), an implication ($B \rightarrow C$), an equivalence ($B \equiv C$).



Proof.

- Clearly no atomic formula can also be “read” as one of
a negation, a disjunction, a conjunction, an implication, an equivalence
since the atomic *contains no glue*, but all the others do.
- Can we read a formula A as two *distinct* negations?
That is, *using in this proof ONLY “=” as equality of strings*, can we have

$$A = (\neg B) = (\neg C)?$$

No, since $(\neg B) = (\neg C)$ implies that after we match the *first two symbols* (left to right) then we will continue matching all symbols —by position— until we match all of B with C and finally match the rightmost “)”.

- Can we read a formula A as a *negation* **and** as a *disjunction*, or a *conjunction*, or an *implication*, or an *equivalence*? That is, can I have

$$A = (\neg B) = (C \circ D)?$$

No, since if we have $(\neg B) = (C \circ D)$, then from left to right the first position is OK (match) but the 2nd is NOT: *C cannot begin with “ \neg ”* (see 2.1.12).

- Can we read a formula A as a $(B \circ C)$ and also *differently* as a $(D \diamond Q)$, *where \diamond stands for any binary glue (including “ \circ ”)*?

Let's assume that we can and get a contradiction.

Well, note first that if $(B \circ C) = (D \diamond Q)$ then if we have $B = D$ then this forces $\circ = \diamond$ and hence also that $C = Q$) which trivially (remove the ending “)”) leads to $C = Q$.

BUT this is not the case that we are looking at.

So, assume that $B \neq D$. There are **two cases**.

Case 1. B is shorter than D , so is a nonempty proper prefix of D . Then, by 2.1.11, B has an excess of left brackets. But being a wff it also has balanced numbers of left/right brackets. **Contradiction!**

Case 2. D is shorter than B so is a nonempty proper prefix of B . Then, by 2.1.11, D has an excess of left brackets. But being a wff it also has balanced numbers of left/right brackets. **Contradiction!**

□

Why do we care about unique readability?

OK. Let us define *arithmetic expressions* (using only $+$, \times) on three numbers, 1, 2, 3. Let us name all such expressions by the *generic name* “ E ”.

Consider the recursive definition below for E , where “ $::=$ ” is read “is defined as (the rhs)”

1. $E ::= 1$ or $E ::= 2$ or $E ::= 3$

OR

2. $E ::= E + E$.

The lhs says that to figure it out, figure out the *two* rhs E (**recursive calls!**) and put a glue of “ $+$ ” between the two.



I can also say that the definition here says “An (arithmetic) expression E is the sum of TWO arithmetic expressions E ”.

Is $E ::= E' + E''$ “more correct?”. **NO!**

In recursive programming you call the SAME procedure E recursively. Not some other E' and E'' !



OR

3. $E ::= E \times E$.

The lhs says that to figure it out, figure out the *two* rhs E (**recursive calls!**) and put a glue of “ \times ” between the two.

So an example of E is

$$1 + 2 \times 3 \tag{1}$$

What is its value? What are the i.p. of (1)?

Well, *if I say 1 + 2 (+ done first) and 3*, then I get value 9.

If I say 1 and 2 × 3 (× done first), then I get value 7.

Value is slippery! We have ambiguity!



So unique i.p. IS important!



2.2 Boolean Semantics

*Boolean Logic is about the **behaviour of glue**. That is, we use Boolean logic to find out how glue influences the truth-value of an entire wff, assuming values are arbitrarily assigned to its atomic subformulas.*

What values do we have in mind?

The so-called truth-values, *true* and *false*.

These values are OUTSIDE Boolean Logic.

Did *you* see them in the alphabet \mathcal{V} ? **Nor did I!!**

They are in the metatheory of Boolean Logic, that is, in the domain where we are speaking about the logic, rather than using the logic.

2.2.1 Definition. A state v (or s) is a *function* that assigns the value **f** (*false*) or **t** (*true*) to *every* Boolean variable —any way we please!— while the constants \perp and \top , necessarily, *always* get the values **f** and **t** respectively.

None of these symbols — v , s , **t**, **f**— are in the Boolean logic alphabet \mathcal{V} . They are *all metasymbols* in the *meta*theory.

The **f** and **t** we call *truth values*.

On paper or on the chalk board one usually *underlines* rather than *bolds* —as bolding is cumbersome— so one denotes **f** as f and **t** as t respectively.

The fact that v gives (assigns) the value **f** to the variable q'' is denoted by $v(q'') = \mathbf{f}$. □



Therefore a state v is an *infinite* input/output *table* like the one below

input	output
\perp	f
\top	t
p	t
q	f
\vdots	\vdots

where no two rows can have the same input but different outputs.

In the jargon of MATH1019/1028 the table is what we call a *function*! This observation justifies the notation

$$\begin{array}{ccc}
 \text{function} & & \text{output} \\
 \downarrow & & \downarrow \\
 v & (\quad q'' \quad) = & \mathbf{f} \\
 \uparrow & & \\
 \text{input} & &
 \end{array}$$

in the last sentence of Definition 2.2.1.

► Why an *infinite* table?

Because our *Boolean logic language* has *infinitely many variables* and a state, by definition, assigns a value to *each of them*.



2.2.2 Definition. (Truth tables) In the *metatheory* of Boolean logic —so, outside logic itself— there are five *operations* we are *interested* in that can be applied on the members of the set of *truth values* $\{\mathbf{t}, \mathbf{f}\}$.

Each operation takes its input(s) from the above set, and its outputs are also in this set.

*We have one operation for each connective (glue) and in order to keep track of which is **formal** and which is **not** we use in the METATHEORY (outside Logic) the generic letter F (for “function”) subscripted by the name of the corresponding glue.*

These functions of the metatheory are called Boolean functions and are the following.

$$F_{\neg}(x), F_{\vee}(x, y), F_{\wedge}(x, y), F_{\rightarrow}(x, y), F_{\equiv}(x, y)$$



So, “ \vee ” does *NOT* operate on *inputs* \mathbf{f}, \mathbf{t} .

\vee expects **TWO** inputs that are wff! \mathbf{t}, \mathbf{f} are NOT wff and we operate with F_{\vee} on these two instead. **In the META THEORY.**

The behaviour of these functions —input/output behaviour, that is— is fully described by the following table that goes by the nickname “*truth table*”.

x	y	$F_{\neg}(x)$	$F_{\vee}(x, y)$	$F_{\wedge}(x, y)$	$F_{\rightarrow}(x, y)$	$F_{\equiv}(x, y)$
f	f	t	f	f	t	t
f	t	t	t	f	t	f
t	f	f	t	f	f	f
t	t	f	t	t	t	t

□

One often sees truth values 0 for **f** and 1 for **t** or even the other way around! The metatheory allows many tastes! Not, though, the formal theory!

x	y	$F_{\neg}(x)$	$F_{\vee}(x, y)$	$F_{\wedge}(x, y)$	$F_{\rightarrow}(x, y)$	$F_{\equiv}(x, y)$
0	0	1	0	0	1	1
0	1	1	1	0	1	0
1	0	0	1	0	0	0
1	1	0	1	1	1	1

The above depiction of truth values as “0” and “1” makes clear that truth values are NOT part of the theory.

Chapter 3

What makes our Logic “Classical”

3.1 States and Truth tables

Refer to the truth table on p.64 and let us discuss the column of $F_{\rightarrow}(x, y)$.

The most “straightforward” entry in this column is arguably, the one for input (\mathbf{t}, \mathbf{f}) .

This function is describing the truth-value of *implications*, and the x input is the *hypothesis* while the y input is the *conclusion*.

Thus having $F_{\rightarrow}(\mathbf{t}, \mathbf{f}) = \mathbf{f}$ can be **interpreted** as saying that *the implication cannot be RIGHT, i.e., t, IF we start with a **true** hypothesis and end up with a **false** conclusion.* IMPLICATION MUST PRESERVE TRUTH is our PRINCIPLE.

This *principle indeed supports* the behaviour of F_{\rightarrow} in the other three rows.

For example you would be wrong to tell me: “Hey, $F_{\rightarrow}(\mathbf{f}, \mathbf{t})$ is not right”. I will respond: “Oh yeah? Show me that it does *not preserve* truth from hypothesis to conclusion! **You cannot show me a truth here that fails to be preserved!**”



So far, *states* give meaning (values) *to atomic formulas only*. Let us *extend* this meaning-giving to *any wff*.



3.1.1 Definition. (The value of a wff in some state, v)

We *extend any* state v to be meaningful *not only for atomic arguments* but also for any *wff* arguments.

We will call such an *extension of v* by the same letter, but will “cap” it with a “hat”, \bar{v} , since it is a *different function!*

This one, \bar{v} , acts on ANY wff, not only on atomic ones.

What *IS* an “EXTENSION” of v ?

It is a *function* \bar{v} that *on the arguments where v is defined* so is \bar{v} and *gives the same output!*

But \bar{v} is ALSO defined *on more inputs: On ALL wff found in WFF.*

The definition of \bar{v} is **INDUCTIVE** (also called **RECURSIVE**):

The first three lines below simply say that \bar{v} agrees with v on the inputs —atomic they are— that the latter is defined on.

The remaining lines trace along **the inductive definition of wff**, and give the value of a wff **using the values** —via “recursive calls”— **of its UNIQUE i.p.**



You see now the **significance** of the uniqueness of i.p.!!!



$$\begin{aligned}
 \bar{v}(\mathbf{p}) &= v(\mathbf{p}) \\
 \bar{v}(\top) &= \mathbf{t} \\
 \bar{v}(\perp) &= \mathbf{f} \\
 \bar{v}((\neg A)) &= F_{\neg}(\bar{v}(A)) \\
 \bar{v}((A \wedge B)) &= F_{\wedge}(\bar{v}(A), \bar{v}(B)) \\
 \bar{v}((A \vee B)) &= F_{\vee}(\bar{v}(A), \bar{v}(B)) \\
 \bar{v}((A \rightarrow B)) &= F_{\rightarrow}(\bar{v}(A), \bar{v}(B)) \\
 \bar{v}((A \equiv B)) &= F_{\equiv}(\bar{v}(A), \bar{v}(B)) \quad \square
 \end{aligned}$$



Truth tables are more convenient to understand than a bunch of recursive equations; **but even easier to misunderstand!**

For example the \vee -column of the Table in 2.2.2 can also be depicted as:

A	B	$A \vee B$
f	f	f
f	t	t
t	f	t
t	t	t

In recursive definition jargon the above says

$$\bar{v}((A \vee B)) = F_{\vee}(\bar{v}(A), \bar{v}(B))$$

At a glance the table says that to compute the value of $A \vee B$ you just utilise the values of the i.p. A and B as indicated.

The misunderstanding you MUST avoid is this: The two left columns are NOT values you assign to A and B .

You can assign values ONLY to ATOMIC formulas!

What these two columns DO say is that *the formulas A and B have each two possible values.*

That is, 4 pairs of values, as displayed!



3.2 Finite States

Sep. 23, 2024



We say a variable **p** occurs in a formula meaning the obvious: It is, as a string, a substring —a part— of the formula.



3.2.1 Theorem. *Given a formula A . Suppose that two states, v and s , agree on **all** the variables occurring in A . Then $\bar{v}(A) = \bar{s}(A)$.*

Proof. We do induction on the (**formation of the**) formula A :

1. Case where A is atomic. Well if it is \top or \perp then $\bar{v}(A) = \bar{s}(A)$ is true.

If A is **p**, then

$$\bar{v}(A) = v(A) \stackrel{\text{Hypothesis}}{=} s(A) = \bar{s}(A)$$

I.H.: Claim is true for all i.p. of A .

2. Case where A is $(\neg B)$. The value of A —whether under v or under s — is *determined* by the recursive calls $\bar{v}(B)$ and $\bar{s}(B)$.

Seeing that all the variables of B are in A , the I.H. yields $\bar{v}(B) = \bar{s}(B)$ and —as F_{\neg} “FLIPS” both these values— we get $\bar{v}(A) = \bar{s}(A)$.

3. Case where A is $(B \circ C)$. The value of A —whether under v or under s — is *determined* by the recursive calls $\bar{v}(B)$ and $\bar{v}(C)$ on one hand and $\bar{s}(B)$ and $\bar{s}(C)$ on the other.

Seeing that all the variables of B and C are in A , the I.H. yields

$$\bar{v}(B) = \bar{s}(B) \text{ and } \bar{v}(C) = \bar{s}(C) \quad (*)$$

Hence

$$\bar{v}(A) = F_{\circ}(\bar{v}(B), \bar{v}(C)) \stackrel{\text{by } (*)}{=} F_{\circ}(\bar{s}(B), \bar{s}(C)) = \bar{s}(A)$$

□



3.2.2 Remark. (Finite “appropriate” States) A state v is *by definition an infinite table*.

By the above theorem, the value of any wff A in a state v *is determined only by the values of v ON THE VARIABLES OF A* , since *any other state that agrees with v on said variables gives the same answer*.

Thus, going forward we will be utilising *finite appropriate states* to compute the truth values of any wff.

*That is, we discard from the infinite state all the rows that contain variables **NOT** occurring in the formulas of interest.*

□



3.2.3 Example. Under **no state** v can we have

$$((A \rightarrow B) \rightarrow A) \rightarrow A$$

evaluate as **f**. Exercise!

□

3.3 Tautologies and Tautological Implication

3.3.1 Definition. (Tautologies and other things...)

1. A *Tautology* is a formula A which is true in *all* states.
That is, for *all* v , we have $\bar{v}(A) = \mathbf{t}$.

We write “ $\models_{\text{taut}} A$ ” for “ A is a tautology”.

2. A *contradiction* is a formula A such that, for *all* v , we have $\bar{v}(A) = \mathbf{f}$.

Clearly, for *all* v , we have $\bar{v}(\neg A) = \mathbf{t}$.

3. A is *satisfiable* iff for *some* v , we have $\bar{v}(A) = \mathbf{t}$.

We say that v satisfies A .

► *Boolean logic for the user helps to discover tautologies.* □

We saw that WFF denotes the set of all (well-formed) formulas.

Capital Greek letters that are different from any Latin capital letter are used to denote arbitrary sets of formulas. Such letters are $\Gamma, \Delta, \Phi, \Psi, \Omega, \Pi, \Sigma$. As always, in the rare circumstance you run out of such letters you may use primes and/or (natural number) subscripts.

3.3.2 Definition. (Tautological implication: binary \models_{taut})

1. Let Γ be a set of wff. *We say that v satisfies Γ iff v satisfies —that is, makes **t**— every wff in Γ .*
2. We say that Γ *tautologically implies* A —and we write this as $\Gamma \models_{\text{taut}} A$ — iff every state v that satisfies Γ also satisfies A .

The configuration

$$\Gamma \models_{\text{taut}} A \quad (1)$$

is called a tautological implication claim.

We call Γ *the set of hypotheses* or *premises* of the

tautological implication, while A is the *conclusion*.

Instead of $\{A, B, C\} \models_{\text{taut}} D$ we write $A, B, C \models_{\text{taut}} D$. □



IMPORTANT! The task to verify (1) needs work on our part **ONLY** with v that satisfy Γ .

If there is NO such v then the claim (1) is VACUOUSLY valid! YOU *cannot* contradict its validity, because to do so you **will need** a v that *satisfies* Γ but NOT A .

You have **NO COUNTEREXAMPLE.**



3.3.3 Example.

(1) If $\models_{\text{taut}} A$, then for any Σ , we have $\Sigma \models_{\text{taut}} A$.

The converse is not valid:

(2) We have $\mathbf{p} \models_{\text{taut}} \mathbf{p} \vee \mathbf{q}$. Indeed, for any v such that $v(\mathbf{p}) = \mathbf{t}$ we compute $\bar{v}(\mathbf{p} \vee \mathbf{q}) = \mathbf{t}$ from the truth table for \vee .

Yet, $\mathbf{p} \vee \mathbf{q}$ is NOT a tautology. Just take $v(\mathbf{p}) = v(\mathbf{q}) = \mathbf{f}$

Note also the obvious: $A \models_{\text{taut}} A \vee B$, for any wff A and B . Again use the truth table of p.70. \square

In view of 3.2.1 we can check all of *satisfiability*, *tautology* status, and *tautological implication* with *finite Γ using a finite truth table*.

Examples.

Example 1. $\perp \models_{\text{taut}} A$.

Because no v satisfies the lhs of “ \models_{taut} ” so according to Definition, I rest my case.

Example 2. Let us build a truth table for $A \rightarrow B \vee A$ and see what we get.

I wrote sloppily, according to our priorities agreement.

I mean $(A \rightarrow (B \vee A))$.

We align our part-work under the glue since it is the glue that causes the output.

Here \rightarrow is the last (applied) glue. *Under it we write the final results for this formula.*

Since A and B *are not necessarily atomic*, the values under A and B in the table below are *possible* values *NOT assigned values!* *So $(A \rightarrow (B \vee A))$ is a tautology.*

A	B	A	\rightarrow	B	\vee	A
f	f		t		f	
f	t		t		t	
t	f		t		t	
t	t		t		t	

Sep. 25, 2024

Example 3. Here is another tautology. I will verify this by a shortcut method, WITHOUT building a truth table.

I will show

$$\models_{\text{taut}} ((A \rightarrow B) \rightarrow A) \rightarrow A \quad (1)$$

I will do so by arguing that *it is IMPOSSIBLE TO MAKE (1) FALSE*.

- *If (1) is false* then *A is false* and $(A \rightarrow B) \rightarrow A$ *is true*.
- Given the two blue statements above, it *must* be that *A \rightarrow B is false*. *IMPOSSIBLE, since A is false!*

And another thing: If A is a tautology we say this in symbols as

$$\models_{\text{taut}} A \quad (1)$$

Contrast with tautological implication from Γ :

$$\Gamma \models_{\text{taut}} A \quad (2)$$

Incidentally, (2) does NOT imply (1): E.g., we have $p \models_{\text{taut}} p \vee q$, but $\not\models_{\text{taut}} p \vee q$.

On the other hand (1) implies (2) for all Γ choices!

Chapter 4

Substitution and Schemata

4.0.1 Definition. (Substitution in Formulas)

The *META*notation

$$A[\mathbf{p} := B] \tag{1}$$

where A and B are formulas and \mathbf{p} is any variable means

- **As an Action:** “*Find and replace by B ALL occurrences of \mathbf{p} in A* ”.
- **As a Result:** *The STRING resulting from the action* described in the previous bullet. \square



1. In the *META*theory of Logic where we use the expression “[**p** := *B*]” **we Agree to Give it The Highest priority**: Thus, $A \wedge B[\mathbf{q} := C]$ means $A \wedge (B[\mathbf{q} := C])$ and $\neg A[\mathbf{p} := B]$ means $\neg(A[\mathbf{p} := B])$
2. Clearly if **p** does NOT occur in *A*, then the “action” found nothing to replace, so the resulting string — according to (1)— in this case is just *A*; NO CHANGE.




We observe the following, according to the inductive definition of formulas.

With reference to (1) of page 81, we prove that the result of (1) is a wff.

So how do we do $A[\mathbf{p} := B]$?

Case 1. Say A is atomic. We have three subcases.

- A is \mathbf{p} . Then $A[\mathbf{p} := B] = B$
- A is \mathbf{q} —where by \mathbf{q} I denote here a *variable other than the one \mathbf{p} stands for*. Then $A[\mathbf{p} := B] = A$ —*no change*.
- A is \perp or \top . Then $A[\mathbf{p} := B] = A$ —*no change*.

 So in the atomic case, $A[\mathbf{p} := B]$ is ONE OF A or B . A wff!



According to the recursive definition of wff, we have two more cases for what A is.

Case 2. A is $(\neg C)$. Thus all \mathbf{p} 's of A are in C .



Any substitution done happens in C .



The substitution steps are depicted below:

$$A = \overbrace{\boxed{ (}}^{\text{3 add "("; 2 add } \neg; \text{ 1 plug } B \text{ in each } \mathbf{p}; \text{ 4 add ")"}} \overbrace{\boxed{ \neg }}^{\text{3 add "("; 2 add } \neg; \text{ 1 plug } B \text{ in each } \mathbf{p}; \text{ 4 add ")"}} \underbrace{\boxed{ \dots \mathbf{p} \dots \mathbf{p} \dots }}_C \overbrace{\boxed{) }}^{\text{3 add "("; 2 add } \neg; \text{ 1 plug } B \text{ in each } \mathbf{p}; \text{ 4 add ")"}} \quad (\dagger)$$

The above actions in (\dagger) have as result

$$A[\mathbf{p} := B] \text{ is } \left(\neg \underbrace{C[\mathbf{p} := B]}_{\text{by I.H. a wff}} \right)$$

Hence $A[\mathbf{p} := B]$ is a wff in our Case 2.

(\dagger')

Case 3. A is $(C \circ D)$. Thus all \mathbf{p} 's of A are in C and D .

The substitution steps are depicted below:

$$A = \overbrace{\boxed{(\quad)}}^{\text{4 add "("; 1 plug } B \text{ in each } \mathbf{p}; \text{3 add "o"; 2 plug } B \text{ in each } \mathbf{p}; \text{5 add ")"}} \underbrace{\boxed{\dots \mathbf{p} \dots}}_C \boxed{\circ} \underbrace{\boxed{\dots \mathbf{p} \dots}}_D \boxed{)} \quad (\ddagger)$$

The above actions in (\ddagger) have as result

$$A[\mathbf{p} := B] \text{ is } \left(\underbrace{C[\mathbf{p} := B]}_{\text{by I.H. a wff}} \circ \underbrace{D[\mathbf{p} := B]}_{\text{by I.H. a wff}} \right)$$

Hence $A[\mathbf{p} := B]$ is a wff in our Case 3.

(\ddagger')

4.0.2 Proposition. *For every wff A and wff B and any variable \mathbf{p} , $A[\mathbf{p} := B]$ is* a wff.*

Proof. See the preceding argument. \square



We are poised to begin describing the **proof system of Boolean logic**.

To this end we will need the notation that is called *formula schemata* or *formula “schemas”* (use the latter only if you think “schema” is an English word—but it is not!).

$$(A \vee (B \rightarrow \mathbf{p}))$$

$$A[\mathbf{p} := B]$$

4.0.3 Definition. (Schema, Schemata) *Add to the alphabet \mathcal{V} the following symbols:*

1. “[”, “]” and “:=”
2. *All NAMES of formulas: A, B, C, \dots , with or without primes and/or subscripts.*
3. *All **metasymbols** for variables: $\mathbf{p}, \mathbf{q}, \mathbf{r}$, with or without primes and/or subscripts.*

Then a *formula schema* is a **STRING** over the **augmented alphabet**, which becomes a wff whenever all

*We are purposely sloppy with jargon here—like everybody else in the literature: “IS” means “results into”.

metasymbols of types 2 and 3 above, which occur in the string, are replaced by wff and actual variables (meaning non bold p, q, r'', q'''_{13}) respectively, and all actions indicated by $[\mathbf{p} := B]$ are performed.

*A formula that we obtain by the process described in the paragraph above is called an **Instance of the Schema**.* □



Three examples of schemata.

(1) A : This Schema stands (is a placeholder) for a wff!
So trivially, *if I plug into A an actual wff, I get that wff as an **instance**!*

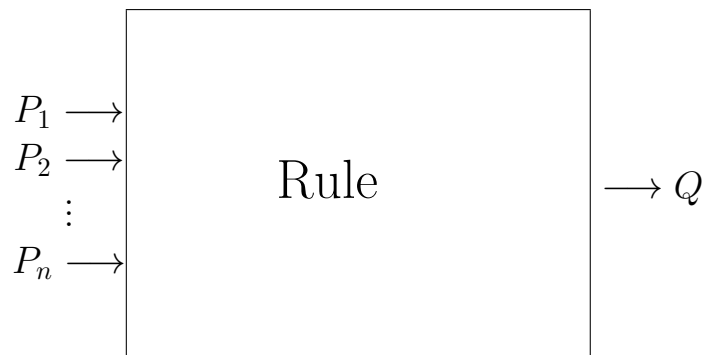
(2) $(A \equiv B)$: Well, whatever formulas I substitute into the (metavariables) A and B I get a wff by the inductive definition of wff.

(3) $A[\mathbf{p} := B]$: We know that if I substitute A and B by actual formulas and \mathbf{p} by an actual Boolean variable I get a wff (4.0.2).



Next stop is Proofs!

In *proofs* we use *Axioms and Rules* (of *Inference*).



A rule as above indicates that in a proof that we have written the wff P_i already, we have the (mathematical) right to write Q *at any proof stage after that!*

It is the habit in the literature to write Rules as *fractions*:

$$\frac{P_1, P_2, \dots, P_n}{Q} \quad (R)$$

where *all* of P_1, \dots, P_n, Q are schemata.

The P_i are *input metavariables* and the Q is the *output metavariable* (only one!)

Thus the “fraction” depicts an action of providing inputs to the P_i and obtaining an output from the metavariable Q .

The P_i are also called *hypotheses* and the Q is called *conclusion*.

More Jargon. We can call the P_i on the numerator of the rule “the *premise(s)*”.

The single schema in the denominator we may also call the “*result*”.

4.1 Rules and Axioms of Boolean Logic

4.1.1 Definition. (Rules of Inference of Boolean Logic)

There are just two:


Rule1

$$\frac{A \equiv B}{C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]} \quad (Leib)$$

*There are NO restrictions in the use of “Leib”
—that means Leibniz.*

In particular,

- (a) *it is NOT required that \mathbf{p} actually occurs in C .*

 *If it does not, then the denominator is just $C \equiv C$.*



- (b) *The single hypothesis can be ANY equivalence.*

- (c) *For any INPUT we have INFINITELY MANY possible results.*

Rule2 “Equanimity” Rule.

$$\frac{A, A \equiv B}{B} \quad (Eqn)$$

There are NO restrictions in the use of “Equanimity” other than

“A” *must* be the left part of the equivalence on the numerator.



Does it matter “left” or “right”? FOR NOW YES!, as we have NO basis to decide otherwise and will NOT be caught “importing” so-called “knowledge” (from other courses) whose validity we did NOT prove in our Logic; YET!!!



□

4.1.2 Definition. (Axioms of Boolean Logic) In the following, (1)–(11), A, B, C *name* or *stand for* arbitrary wff.

Properties of \equiv

$$\text{Associativity of } \equiv \quad ((A \equiv B) \equiv C) \equiv (A \equiv (B \equiv C)) \quad (1)$$

$$\text{Commutativity of } \equiv \quad (A \equiv B) \equiv (B \equiv A) \quad (2)$$

Properties of \perp, \top

$$\top \text{ and } \perp \quad \top \equiv \perp \equiv \perp \quad (3)$$

Properties of \neg

$$\text{Introduction of } \neg \quad \neg A \equiv A \equiv \perp \quad (4)$$

Properties of \vee

$$\text{Associativity of } \vee \quad (A \vee B) \vee C \equiv A \vee (B \vee C) \quad (5)$$

$$\text{Commutativity of } \vee \quad A \vee B \equiv B \vee A \quad (6)$$

$$\text{Idempotency of } \vee \quad A \vee A \equiv A \quad (7)$$

$$\text{Distributivity of } \vee \text{ over } \equiv \quad A \vee (B \equiv C) \equiv A \vee B \equiv A \vee C \quad (8)$$

$$\text{“Excluded Middle”} \quad A \vee \neg A \quad (9)$$

Properties of \wedge

$$\text{“Golden Rule”} \quad A \wedge B \equiv A \equiv B \equiv A \vee B \quad (10)$$

Properties of \rightarrow

$$\text{Implication} \quad A \rightarrow B \equiv A \vee B \equiv B \quad (11)$$

All of the above (1)–(11) except (3) are schemata for axioms.

We call them *Axiom Schemata*, while (3) is an *Axiom*.

Each axiom schema above defines *infinitely many axioms* that are its *Instances*.

So our axioms are (3) and all the instances of the Ax-

iom Schemata (1), (2), (4)–(11).

We reserve the Greek letter Λ for the set of all Axioms of Boolean Logic. □

DISCLOSURE. You can verify (Exercise! TRY IT!!!) that each axiom is a tautology. The aim is that starting with these tautologies in a proof, and having the Rules PRESERVE TRUTH as we proceed with the proof (WAIT and SEE!), it follows that ANY proof is composed ONLY of **true statements**.

4.1.3 Definition. (Proofs) Let Γ (could use Σ, Θ, Ψ etc., instead of Γ —there is nothing special about the letter Γ ! Use Σ or Δ or Φ if you prefer!!!) be some set of wff.

A proof from Γ is any *finite ordered* sequence of formulas that satisfy the following two *specifications*:

At every step of the *Construction (that we call “Proof”)* we *may write*

Proof 1. *Any ONE formula* from Λ or Γ .

Proof 2. *Any wff A which is the RESULT of an Application* of the rule *Leib* or rule *Eqn* to wff(s)

that appeared in *THIS* proof *before A*.

A proof from Γ is also called a “ Γ -proof”. □

Γ -proofs. No mystery in Γ . Follow all definitions (including those for the use of $\Sigma, \Phi, \Delta, \Gamma, \Psi$).



4.1.4 Remark. (1) So, *a proof is a totally syntactic construct, totally devoid of semantic concepts.*

(2) Γ is a *convenient* set of “*additional hypotheses*”.

Syntactically the elements of Γ “behave” like the Axioms from Λ —as it is clear from 4.1.3, item 1— but *semantically they are NOT the same*:

While every member of Λ is a *tautology* by choice,

this *need NOT be the case for the members of Γ* .

(3) Since *every proof* (from some Γ) has *finite length*,

only a finite part of Γ and Λ can ever appear in some proof.



Sep. 30, 2024

4.1.5 Definition. (Theorems) *Any* wff A that *appears in a Γ -proof* is called a Γ -theorem.

In particular, any member from Γ or Λ that appears in a proof is a Γ -theorem.

Remember this!

We also say, “ A is a theorem *from* Γ ”.

In symbols, the sentence “ A is a Γ -theorem”, is denoted by “ $\Gamma \vdash A$ ”.

If $\Gamma = \emptyset$ then we write $\vdash A$.



That is, Λ never appears to the left of the turnstile “ \vdash ”.



We call an A such that $\vdash A$ an absolute or logical theorem. □



4.1.6 Remark. That A is a Γ -theorem is certified by a Γ -proof like this

$$B_1, \dots, B_n, \textcolor{red}{A}, C_1, \dots, C_m \quad (1)$$

the sequence (1) obeying the *specifications* of 4.1.3.

Clearly, the sequence (2) below also satisfies the specifications, since each specification for a B_i or A that utilises *rules* refers to formulas *to the left only*.

Thus the sequence (2) is also a Γ -proof of A !

$$B_1, \dots, B_n, \textcolor{red}{A} \quad (2)$$

The bottom line of this story is expressed as either

1. *If you are proving a theorem A , just stop as soon as you wrote it down with justification* in a proof!

OR

2. *A Γ -theorem is a wff that appears at the END of some proof.*



Concatenating two Γ -proofs

$$A_1, \dots, A_n$$

and

$$B_1, B_2, \dots, B_r$$

results in a Γ -proof.

Indeed, checking

$$B_1, B_2, \dots, B_r, A_1, \dots, A_n$$

from left to right we give EXACTLY the same reasons
that we gave for writing the formulas down in each stan-
dalone proof.

So we learnt that a Γ -theorem, *let's call it A* , satisfies

1. *A is member of Λ or Γ*
2. *A appears in a Γ -proof as the result of an application of Eqn to TWO wff to its left in the proof.*
3. *A appears in a Γ -proof as the result of an application of Leib to ONE wff to its left in the proof.*

Let us rephrase the blue “appears” above, remembering that a Γ -theorem IS a formula that appears in a Γ -proof.

1. *A is member of Λ or Γ*
2. *A is the result of an application of Eqn to two Γ -theorems.*
of the forms X and $X \equiv A$
3. *A is the result of an application of Leib to one Γ -theorem.*
of the form $X \equiv Y$.

4.1.7 Exercise. How do we do this Exercise?

By providing a Γ -proof IN WHICH our target theorem appears, OR by using the Inductive Definition of the previous page.

- (1) $A, B, C \vdash A$, for any wff A
- (2) More generally, if $A \in \Sigma$, then $\Sigma \vdash A$
- (3) $\vdash B$, for all $B \in \Lambda$

□



4.1.8 Remark. (Hilbert-style Proofs)

A Γ -proof is also called a “Hilbert-style proof” —in honour of the great mathematician *David Hilbert*, who was *the first big supporter* of the idea *to use SYNTACTIC (FORMAL) logic as a TOOL in order to do CORRECT mathematics*.

We arrange Hilbert proofs *vertically*, one formula per line, *numbered by its position number*, adding “*annotation*” to the right of the formula we just wrote, articulating briefly *HOW exactly* we followed the spec of Definition 4.1.3.

Practical Note. If one forgets *numbering* or *annotation*, or that *each line contains ONE fff ONLY*, then this results in a very bad grade! :)



4.1.9 Example. (Some very simple Hilbert Proofs)

(a) We verify that “ $A, A \equiv B \vdash B$ ” (goes without saying, *for all wff A and B*).

Well, just write a proof of B with “ Γ ” being $\{A, A \equiv B\}$.

BTW, we indicate a finite “ Γ ” like $\{A, A \equiv B\}$ without the braces “ $\{ \}$ ” when writing it to the left of “ \vdash ”.

- (1) A $\langle \text{hypothesis} \rangle$
- (2) $A \equiv B$ $\langle \text{hypothesis} \rangle$
- (3) B $\langle (1) + (2) + (Eqn) \rangle$



Incidentally, *members of Γ are annotated as “hypotheses” and going forward we just write “hyp”*.

Members of Λ we annotate as “Axioms”.





Since A and B are arbitrary *undisclosed* wff, *the expression $A, A \equiv B \vdash B$ is a Theorem Schema* (*a theorem, no matter what formulas we plug into A and B*).



(b) Next verify the Theorem Schema

I will skip in class, but you study this!

$$A \equiv B \vdash C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$$

Here you go:

- (1) $A \equiv B$ $\langle \text{hyp} \rangle$
- (2) $C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$ $\langle (1) + \text{Leib} \rangle$

C can be any wff (and p any actual Boolean variable)
 so from ONE hypothesis for fixed A and B we can
 derive an infinite number of theorems of the “shape”
 $C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$.

(c) Something more substantial. *Our First Derived Rule!*

We establish the following *Theorem Schema* that we will refer to as *Transitivity* of \equiv —or simply “*Trans*”.
How? *We write a Hilbert proof!*

$$A \equiv B, B \equiv C \vdash A \equiv C \quad (Trans)$$

- (1) $A \equiv B$ $\langle \text{hyp} \rangle$
- (2) $B \equiv C$ $\langle \text{hyp} \rangle$
- (3) $(A \equiv B) \equiv (A \equiv C)$ $\langle (2) + (Leib), \text{Denom. } “A \equiv \mathbf{p}”; \mathbf{p} \text{ “fresh”} \rangle$
- (4) $A \equiv C$ $\langle (1) + (3) + (Eqn) \rangle$

What is fresh? Can I always have it? Why must \mathbf{p} be fresh?

Say A is $\mathbf{p} \wedge \mathbf{q}$.

Then, feeding B to \mathbf{p} of “ $A \equiv \mathbf{p}$ ” the latter becomes

$$B \wedge \mathbf{q} \equiv B$$

which is NOT the SAME STRING AS $A \equiv B$.

$$\overbrace{B \wedge \mathbf{q}}^{\text{this is NOT } A} \equiv B$$

(d) *And a Tricky One!* Verify that “ $A \equiv A$ ” is an absolute theorem for all A . That is,

$$\vdash A \equiv A$$

No “HYP” in the proof below!!

- (1) $A \vee A \equiv A$ $\langle \text{axiom} \rangle$
 - (2) $A \equiv A$ $\langle (1) + (Leib): A[\mathbf{p} := A \vee A] \equiv A[\mathbf{p} := A]$
- where \mathbf{p} is “fresh”

Can start the proof with any known equivalence that is an axiom. For example, line one could contain $\top \equiv \perp \equiv \perp$. \square

4.1.10 Metatheorem. (Hypothesis Strengthening)

If $\Gamma \vdash A$ and $\Gamma \subseteq \Delta$, then also $\Delta \vdash A$.

Proof. A Γ -proof for A is also a Δ -proof, since every time we say about a formula B in the proof “legitimate since $B \in \Gamma$ ” we can say instead “legitimate since $B \in \Delta$ ”.

□

4.1.11 Metatheorem. (Transitivity of \vdash) Assume $\Gamma \vdash B_1, \Gamma \vdash B_2, \dots, \Gamma \vdash B_n$. Let also $B_1, \dots, B_n \vdash A$. Then we have $\Gamma \vdash A$.

Also: Using *DERIVED RULES* $B_1, \dots, B_n \vdash A$.

Proof.

We have Γ -proofs

$$\boxed{\dots, B_1} \tag{1}$$

$$\boxed{\dots, B_2} \tag{2}$$

$$\vdots$$

$$\boxed{\dots, B_n} \tag{n}$$

We also have a $\{B_1, \dots, B_n\}$ -proof

$$\boxed{\dots, B_i, \dots, A} \tag{n+1}$$

Concatenate all proofs (1)–(n) (in any order) and to the right of the result glue the proof (n + 1).

We have the following proof:

$$\begin{array}{c}
 \text{DON'T say "hyp"} \\
 \downarrow \\
 \boxed{\dots, B_1}, \boxed{\dots, B_2}, \dots, \boxed{\dots, B_n}, \boxed{\dots, B_i, \dots, A} \\
 \uparrow \\
 \text{SAY: "obtained earlier; see box } i\text{"}
 \end{array}$$

□

So if we view $B_1, \dots, B_n \vdash A$ as a (derived or “macro” rule) then this “rule” is applicable!

If the B_i are Γ -theorems and $B_1, \dots, B_n \vdash A$, then we can apply the latter as a “rule” to obtain the Γ -theorem A .

We say “ $B_1, \dots, B_n \vdash A$ is a derived rule”.

4.1.12 Corollary. *If $\Gamma \vdash A$ and also $\Gamma \cup \{A\} \vdash B$, then $\Gamma \vdash B$.*



In words, the conclusion says that A drops out as a hypothesis and we get $\Gamma \vdash B$.

That is, a THEOREM A can be invoked just like an axiom OR a hyp in a proof!



Proof. We have *two* proofs:

$$\boxed{\begin{array}{c} \text{from } \Gamma \\ \hline \dots A \end{array}}$$

and

$$\boxed{\begin{array}{c} \text{from } \Gamma \cup \{A\} \\ \hline \dots A \dots B \end{array}}$$

When the second box is *standalone*, the justification for A is “hyp”.

Now concatenate the two proofs above in the order

$$\boxed{\begin{array}{c} \text{from } \Gamma \\ \hline \dots A \end{array}} \quad \boxed{\begin{array}{c} \text{from } \Gamma \cup \{A\} \\ \hline \dots \textcolor{red}{A} \dots B \end{array}}$$

*Now change all the justifications for the *red* A in the right box from “hyp” to the same exact reason you gave to the A in box one —OR, as in the proof of 4.1.11 say about the *red* A : “obtained earlier in box 1”.*

Thus, the status of A as “hyp” is removed and B is proved from Γ alone. \square

4.1.13 Corollary. If $\Gamma \cup \{A\} \vdash B$ and $\vdash A$, then $\Gamma \vdash B$.

Proof. By hyp strengthening, I have $\Gamma \vdash A$. Now apply the previous corollary. \square

Oct. 2, 2024

4.1.14 Theorem. $A \equiv B \vdash B \equiv A$

Proof.

- (1) $A \equiv B$ $\langle \text{hyp} \rangle$
- (2) $(A \equiv B) \equiv (B \equiv A)$ $\langle \text{axiom} \rangle$
- (3) $B \equiv A$ $\langle (1,2) + \text{Eqn} \rangle$

4.1.15 Theorem. $\vdash (A \equiv (B \equiv C)) \equiv ((A \equiv B) \equiv C)$

NOTE. This is the mirror image of Axiom (1).

Proof.

- (1) $((A \equiv B) \equiv C) \equiv (A \equiv (B \equiv C))$ $\langle \text{axiom} \rangle$
- (2) $(A \equiv (B \equiv C)) \equiv ((A \equiv B) \equiv C)$ $\langle (1)+4.1.14 \rangle \square$



4.1.16 Remark. Thus, *in a chain of two “ \equiv ” we can shift brackets from left to right (axiom) but also right to left (above theorem).*

So it does not matter how brackets are inserted in such chain.

An induction proof on chain length (see course URL, bullet #4 under Notes:

<http://www.cs.yorku.ca/~gt/courses/MATH1090F23/1090.html>) extends this remark to any chain of “ \equiv ”, of any length.

\square



4.1.17 Theorem. (The other (*Eqn*)) $B, A \equiv B \vdash A$

Proof.

- (1) B $\langle \text{hyp} \rangle$
- (2) $A \equiv B$ $\langle \text{hyp} \rangle$
- (3) $B \equiv A$ $\langle (2) + 4.1.14 \rangle$
- (4) A $\langle (1, 3) + \text{original } (Eqn) \rangle$ \square

4.1.18 Corollary. $\vdash \top$

Proof.

- (1) $\top \equiv \perp \equiv \perp$ $\langle \text{axiom} \rangle$
- (2) $\perp \equiv \perp$ $\langle \text{the “} A \equiv A \text{” theorem} \rangle$
- (3) \top $\langle (1, 2) + (Eqn) \rangle$ \square

4.1.19 Theorem. $\vdash A \equiv A \equiv B \equiv B$

- (1) $(A \equiv B \equiv B) \equiv A$ \langle axiom; brackets as I please! \rangle
- (2) $A \equiv (A \equiv B \equiv B)$ $\langle(1) + 4.1.14\rangle$ \square

4.1.20 Corollary. $\vdash \perp \equiv \perp \equiv B \equiv B$ and $\vdash A \equiv A \equiv \perp \equiv \perp$

NOTE *absence of brackets in theorem AND corollary!*

4.1.21 Corollary. (Redundant \top Theorem)

$\vdash \top \equiv A \equiv A$ and $\vdash A \equiv A \equiv \top$.

Proof.

- (1) $\top \equiv \perp \equiv \perp$ $\langle \text{axiom} \rangle$
- (2) $\perp \equiv \perp \equiv A \equiv A$ $\langle \text{absolute theorem 4.1.20} \rangle$
- (3) $\top \equiv A \equiv A$ $\langle (Trans) + (1, 2) \rangle$

4.1.22 Metatheorem. (Redundant \top METAtheorem)

For any Γ and A , we have $\Gamma \vdash A$ iff $\Gamma \vdash A \equiv \top$.

Proof. Say $\Gamma \vdash A$.

Thus

- $$\begin{array}{ll} \Gamma & \\ \vdots & \\ (1) & A \quad \langle \Gamma\text{-theorem} \rangle \\ (2) & A \equiv A \equiv \top \langle \text{Red. } \top \text{ theorem; 4.1.21} \rangle \\ (3) & A \equiv \top \quad \langle (1, 2) + \text{Eqn} \rangle \end{array}$$

The other direction is similar.

□

4.2 Equational Proofs

Example from high school trigonometry.

Prove that $1 + (\tan x)^2 = (\sec x)^2$ given the identities

$$\tan x = \frac{\sin x}{\cos x} \quad (i)$$

$$\sec x = \frac{1}{\cos x} \quad (ii)$$

$$(\sin x)^2 + (\cos x)^2 = 1 \text{ (Pythagoras' Theorem)} \quad (iii)$$

Equational proof with annotation

$$\begin{aligned}
& 1 + (\tan x)^2 \\
= & \langle \text{by } (i) \rangle \\
& 1 + (\sin x / \cos x)^2 \\
= & \langle \text{arithmetic} \rangle \\
& \frac{(\sin x)^2 + (\cos x)^2}{(\cos x)^2} & (E) \\
= & \langle \text{by } (iii) \rangle \\
& \frac{1}{(\cos x)^2} \\
= & \langle \text{by } (ii) \rangle \\
& (\sec x)^2
\end{aligned}$$

An equational proof in Logic looks like:

$$\overbrace{A_1 \equiv A_2}^{\text{reason}}, \overbrace{A_2 \equiv A_3}^{\text{reason}}, \overbrace{A_3 \equiv A_4}^{\text{reason}} \dots, \overbrace{A_n \equiv A_{n+1}}^{\text{reason}} \quad (1)$$

4.2.1 Metatheorem. (Important Derived Rule!)

$$A_1 \equiv A_2, A_2 \equiv A_3, \dots, A_n \equiv A_{n+1} \vdash A_1 \equiv A_{n+1} \quad (2)$$

Proof. By induction on $n \geq 1$ using the (derived) rule (Trans).

1. *Basis* for $n = 1$. We want $A_1 \equiv A_2 \vdash A_1 \equiv A_2$. This is “ $X \vdash X$ ” done! (see 4.1.7).
2. I.H. Assume (2) for fixed unspecified n .
3. I.S. Do the case $n + 1$ for the n we fixed above, so we want (3) below:

$$A_1 \equiv A_2, A_2 \equiv A_3, \dots, A_{n+1} \equiv A_{n+2} \vdash A_1 \equiv A_{n+2} \quad (3)$$

Here it goes

Oct. 7, 2024

Generalised \equiv -Transitivity (Derived Rule)

(1)	$A_1 \equiv A_2$	$\langle \text{hyp} \rangle$
(2)	$A_2 \equiv A_3$	$\langle \text{hyp} \rangle$
	\vdots	\vdots
(n)	$A_n \equiv A_{n+1}$	$\langle \text{hyp} \rangle$
(n + 1)	$A_1 \equiv A_{n+1}$	$\langle (1 + 2 + \dots + n) + \text{I.H.} \rangle$
(n + 2)	$A_{n+1} \equiv A_{n+2}$	$\langle \text{hyp} \rangle$
(n + 3)	$A_1 \equiv A_{n+2}$	$\langle n + 1, n + 2 + \text{Trans} \rangle$

□

All Equational Proofs are based on Metatheorem 4.2.1:

4.2.2 Corollary. *In an Equational proof (from Γ) like the one in (1) of p.123 we have $\Gamma \vdash A_1 \equiv A_{n+1}$.*

Proof. So we have n Γ -proofs, for $i = 1, \dots, n$,

$$\boxed{\dots, A_i \equiv A_{i+1}}$$

Concatenate them all to get ONE Γ -proof

$$\overbrace{\boxed{\dots, A_1 \equiv A_2}}^{\Gamma\text{-proof}} \dots \overbrace{\boxed{\dots, A_i \equiv A_{i+1}}}^{\Gamma\text{-proof}} \dots \overbrace{\boxed{\dots, A_n \equiv A_{n+1}}}^{\Gamma\text{-proof}}$$

By the DERIVED RULE 4.2.1 the following is a Γ -proof of $A_1 \equiv A_{n+1}$

$$\boxed{\dots, A_1 \equiv A_2} \dots \boxed{\dots, A_i \equiv A_{i+1}} \dots \boxed{\dots, A_n \equiv A_{n+1}}, A_1 \equiv A_{n+1}$$

□

4.2.3 Corollary. *In an Equational proof (from Γ) like the one in (1) of p.123 we have $\Gamma \vdash A_1$ iff $\Gamma \vdash A_{n+1}$.*

Proof. From the above Corollary we have

$$\Gamma \vdash A_1 \equiv A_{n+1} \quad (\dagger)$$

Now split the “iff” in two directions:

- IF (\leftarrow): So we have

$$\Gamma \vdash A_{n+1}$$

This plus (\dagger) plus Eqn yield $\Gamma \vdash A_1$.

- ONLY IF (\rightarrow): So we have

$$\Gamma \vdash A_1$$

This plus (\dagger) plus Eqn yield $\Gamma \vdash A_{n+1}$.

□

Equational Proof Layout

Successive equivalences like “ $A_i \equiv A_{i+1}$ and $A_{i+1} \equiv A_{i+2}$ ” we write vertically, without repeating the shared formula A_{i+1} .

WITH annotation in $\langle \dots \rangle$ brackets

$$\begin{array}{l}
 A_1 \\
 \equiv \langle \text{annotation} \rangle \\
 A_2 \\
 \equiv \langle \text{annotation} \rangle \\
 \vdots \\
 A_{n-1} \\
 \equiv \langle \text{annotation} \rangle \\
 A_n \\
 \equiv \langle \text{annotation} \rangle \\
 A_{n+1}
 \end{array}
 \tag{ii}$$

EXCEPT FOR ONE THING!

(ii) is just ONE FORMULA, namely

$$A_1 \equiv A_2 \equiv \dots \equiv A_n \equiv A_{n+1}$$

where I can put brackets anywhere I please.

It does NOT say the same thing as (1) of p.123.

For example, “ $\top \equiv \perp \equiv \perp$ ” is NOT the same as “ $\underbrace{\top \equiv \perp}_f \wedge \underbrace{\perp \equiv \perp}_t$ ”

The former (blue) is true but the latter (red) is false.

What do we do?

We introduce a metasymbol for an equivalence that acts ONLY on TWO formulas!

AND

Such equivalences CANNOT be chained to form a SINGLE formula.

The symbol we will use for such UNCHAINABLE equivalences is “ \Leftrightarrow ” and thus

“ $A \Leftrightarrow B \Leftrightarrow C$ ” MEANS “ $A \equiv B$ AND $B \equiv C$ ”, NOT “ $A \equiv B \equiv C$ ”.

We say that “ \Leftrightarrow ” is CONJUNCTIONAL while “ \equiv ” is associative.

So the final layout is:

$$\begin{array}{c}
 A_1 \\
 \Leftrightarrow \langle \text{annotation} \rangle \\
 A_2 \\
 \Leftrightarrow \langle \text{annotation} \rangle \\
 \vdots \\
 A_{n-1} \\
 \Leftrightarrow \langle \text{annotation} \rangle \\
 A_n \\
 \Leftrightarrow \langle \text{annotation} \rangle \\
 A_{n+1}
 \end{array}$$

A Lot of Practice Examples Now!

If I say “I will skip this, but you should do it (do study it)”, then indeed you do it and you must know/remember both the proof technique AND the theorem!

You can use the latter in assignments, midterm, exam. As is!

4.2.4 Theorem. $\vdash \neg(A \equiv B) \equiv A \equiv \neg B$

Proof. (Equational)

$$\begin{aligned}
 & \neg(A \equiv B) \\
 \Leftrightarrow & \langle \text{axiom} \rangle \\
 & A \equiv B \equiv \perp \\
 \Leftrightarrow & \langle (Leib) + \text{axiom: } \neg B \equiv B \equiv \perp; \text{Denom: } A \equiv \mathbf{p}; \mathbf{p} \text{ fresh} \rangle \\
 & A \equiv \neg B \quad \square
 \end{aligned}$$

Why do I need Leib above? Why not just use the Axiom? **Because I am replacing PART of a wff.**

► Study this next one, but I will SKIP!

4.2.5 Corollary. $\vdash \neg(A \equiv B) \equiv \neg A \equiv B$

Proof. (Equational)

$$\begin{aligned}
 & \neg(A \equiv B) \\
 \Leftrightarrow & \langle \text{axiom} \rangle \\
 & A \equiv B \equiv \perp \\
 \Leftrightarrow & \langle (Leib) + \text{axiom: } B \equiv \perp \equiv \perp \equiv B; \text{Denom: } A \equiv \mathbf{p}; \mathbf{p} \text{ fresh} \rangle \\
 & A \equiv \perp \equiv B \\
 \Leftrightarrow & \langle (Leib) + \text{axiom: } A \equiv \perp \equiv \neg A; \text{Denom: } \mathbf{q} \equiv B; \mathbf{q} \text{ fresh} \rangle \\
 & \neg A \equiv B
 \end{aligned}$$

□

4.2.6 Theorem. (Double Negation) $\vdash \neg\neg A \equiv A$ **Proof.** (Equational)

$$\begin{aligned}
& \neg\neg A \\
\Leftrightarrow & \langle \text{axiom } "\neg X \equiv X \equiv \perp" \rangle \\
& \neg A \equiv \perp \\
\Leftrightarrow & \langle (Leib) + \text{axiom: } \neg A \equiv A \equiv \perp; \text{Denom: } \mathbf{p} \equiv \perp \rangle \\
& A \equiv \perp \equiv \perp \\
\Leftrightarrow & \langle (Leib) + \text{axiom: } \top \equiv \perp \equiv \perp; \text{Denom: } A \equiv \mathbf{q}; \mathbf{q} \text{ fresh} \rangle \\
& A \equiv \top \\
\Leftrightarrow & \langle \text{red. } \top \text{ thm.} \rangle \\
& A
\end{aligned}$$

□

4.2.7 Theorem. $\vdash \top \equiv \neg \perp$

Proof. (Equational)

$$\begin{aligned} & \top \\ \Leftrightarrow & \langle \text{axiom} \rangle \\ & \perp \equiv \perp \\ \Leftrightarrow & \langle \text{axiom} \rangle \\ & \neg \perp \end{aligned}$$

□

4.2.8 Theorem. $\vdash \perp \equiv \neg \top$

Proof. (Equational)

$$\begin{aligned} & \neg \top \\ \Leftrightarrow & \langle \text{axiom} \rangle \\ & \top \equiv \perp \\ \Leftrightarrow & \langle \text{red. } \top \rangle \\ & \perp \end{aligned}$$

□



Practical Advise. In Equational Proofs move from the most complex side towards the least complex one.



STARTING WITH WHAT YOU ARE PROVING

4.2.9 Theorem. $\vdash A \vee \top$

Proof.

$$A \vee \top$$

$\Leftrightarrow \langle (Leib) + \text{axiom } \top \equiv \perp \equiv \perp; \text{“Denom:” } A \vee \mathbf{p}; \text{Mind brackets!} \rangle$

$$A \vee (\perp \equiv \perp)$$

$\Leftrightarrow \langle \text{axiom} \rangle$

$A \vee \perp \equiv A \vee \perp$ **Bingo!** Recognised an axiom or known theorem. \square



Recall about \equiv that, by axiom (1) and a theorem we proved in the NOTES posted in <http://www.cs.yorku.ca/~gt/courses/MATH1090F23/1090.html> (4th bullet), we have that

in a chain of **any** number of \equiv we may omit brackets.

The same holds for a chain of \vee (and \wedge) using *the same kind of proof, in the same link mentioned above.*



That is,

we do not need to show bracketing in a chain of \vee (or one of \wedge).



How about *moving* formulas around in such a chain?
(*Permuting* them).



It is OK! I prove this for \vee -chains HERE. The proof is identical for \equiv -chains and \wedge -chains (EXERCISE!!)

Prove first this theorem:

$$\vdash B \vee C \vee D \equiv D \vee C \vee B$$

Indeed here is a proof:

$$\begin{aligned} & B \vee C \vee D \\ \Leftrightarrow & \langle \vee \text{ commutes axiom (imagin brackets around } B \vee C) \rangle \\ & D \vee B \vee C \quad (*) \\ \Leftrightarrow & \langle (Leib) + \vee \text{ commutes axiom. "Denom:"} D \vee \mathbf{p} \rangle \\ & D \vee C \vee B \end{aligned}$$

More generally we CAN DO an arbitrary swap (*not only* the END-FORMULAS), that is, we have the theorem

$$\vdash \boxed{A} \vee B \vee \boxed{C} \vee D \vee \boxed{E} \equiv A \vee D \vee C \vee B \vee E$$



The boxed formulas may well be long \vee -chains!



Follows by an application of the previous special case:

$$\begin{aligned} & A \vee \overbrace{B \vee C \vee D} \vee E \\ \Leftrightarrow & \langle (Leib) + \text{special case. "Denom:"} \ A \vee \mathbf{p} \vee E \rangle \\ & A \vee \underbrace{D \vee C \vee B} \vee E \end{aligned}$$

Oct. 9, 2024

4.2.10 Theorem. $\vdash A \vee \perp \equiv A$

Proofs. (Equational)

This time we work with the entire formula, not just one of the sides of “ \equiv ”.



How do we know? We don't! It is just a matter of practice.



$$\begin{aligned}
 & A \vee \perp \equiv A \\
 \Leftrightarrow & \langle (Leib) + \text{axiom } A \equiv A \vee A; \text{“Denom.” } A \vee \perp \equiv \mathbf{p} \rangle \\
 & A \vee \perp \equiv A \vee A \\
 \Leftrightarrow & \langle \text{axiom } \vee \text{ over } \equiv \rangle \\
 & A \vee (\perp \equiv A) \\
 \Leftrightarrow & \langle (Leib) + \text{axiom: } \perp \equiv A \equiv \neg A; \text{“Denom.” } A \vee \mathbf{p} \rangle \\
 & A \vee \neg A \quad \textit{Bingo!} \quad \square
 \end{aligned}$$

Comment on “same mouth” \mathbf{p} used twice in above proof. What about freshness?

4.2.11 Theorem. $\vdash A \rightarrow B \equiv \neg A \vee B$

Proof.

$$\begin{aligned}
 & A \rightarrow B \\
 \Leftrightarrow & \langle \text{axiom} \rangle \\
 & A \vee B \equiv B \quad \quad \quad \blacktriangleleft \text{HERE} \\
 \Leftrightarrow & \langle (Leib) + 4.2.10; \text{“Denom.” } A \vee B \equiv \mathbf{p} \rangle \\
 & A \vee B \equiv \perp \vee B \\
 \Leftrightarrow & \langle \text{axiom} \rangle \\
 & (A \equiv \perp) \vee B \\
 \Leftrightarrow & \langle (Leib) + \text{axiom}; \text{“Denom.” } \mathbf{p} \vee B \rangle \\
 & \neg A \vee B \quad \quad \quad \square
 \end{aligned}$$

4.2.12 Corollary. (IMPORTANT)

$$\vdash \neg A \vee B \equiv A \vee B \equiv B$$

Proof. Start the above proof from spot marked (above)
“HERE”. □

READ THIS! I will skip. Uses IMPORTANT Corollary above twice.

4.2.13 Theorem. (de Morgan 1)

$$\vdash A \wedge B \equiv \neg(\neg A \vee \neg B)$$

Proof.

Long but obvious. Start with the most complex side!

$$\begin{aligned}
 & \neg(\neg A \vee \neg B) \\
 \Leftrightarrow & \langle \text{axiom} \rangle \\
 & \neg A \vee \neg B \equiv \perp \\
 \Leftrightarrow & \langle (Leib) + 4.2.12; \text{“Denom:” } \mathbf{p} \equiv \perp \rangle \\
 & A \vee \neg B \equiv \neg B \equiv \perp \\
 \Leftrightarrow & \langle (Leib) + \text{axiom}; \text{“Denom:” } A \vee \neg B \equiv \mathbf{p} \text{ —order does not matter!} \rangle \\
 & A \vee \neg B \equiv B \\
 \Leftrightarrow & \langle (Leib) + 4.2.12; \text{“Denom:” } \mathbf{p} \equiv B \rangle \\
 & A \vee B \equiv A \equiv B \\
 \Leftrightarrow & \langle \text{GR axiom —order does not matter} \rangle \\
 & A \wedge B
 \end{aligned}$$

□

4.2.14 Corollary. (de Morgan 2)

$$\vdash A \vee B \equiv \neg(\neg A \wedge \neg B)$$

Proof. See Text. Better still, EXERCISE!

MORE About “ \wedge ”

DO ALL that I skip regarding “ \wedge ”.

4.2.15 Theorem. $\vdash A \wedge A \equiv A$

Proof.

$$\begin{aligned}
 & A \wedge A \equiv A \\
 \Leftrightarrow & \langle \text{GR axiom —} \underline{\text{order}} \text{ does not matter} \rangle \\
 & A \vee A \equiv A \qquad \text{Bingo!} \quad \square
 \end{aligned}$$

4.2.16 Theorem. $\vdash A \wedge \top \equiv A$

Proof.

$$\begin{aligned}
 & A \wedge \top \equiv A \\
 \Leftrightarrow & \langle \text{GR axiom} \rangle \\
 & A \vee \top \equiv \top \\
 \Leftrightarrow & \langle \text{Red. } \top \text{ Thm.} \rangle \\
 & A \vee \top \qquad \text{Bingo!} \quad \square
 \end{aligned}$$

4.2.17 Theorem. $\vdash A \wedge \perp \equiv \perp$

Proof.

$$\begin{aligned} A \wedge \perp &\equiv \perp \\ \Leftrightarrow \langle \text{GR axiom} \rangle \\ A \vee \perp &\equiv A \quad \text{Bingo!} \quad \square \end{aligned}$$

► READ the following theorem and its proof!
REMEMBER the result!!!

4.2.18 Theorem. (Distributive Laws between \vee and \wedge)

$$(i) \quad \vdash A \vee B \wedge C \equiv (A \vee B) \wedge (A \vee C)$$

and

$$(ii) \quad \vdash A \wedge (B \vee C) \equiv A \wedge B \vee A \wedge C$$



The above are written in least parenthesised notation!

It is part of your tool-set!



Proof.

We just prove (i).

$$\begin{aligned}
& (A \vee B) \wedge (A \vee C) \\
\Leftrightarrow & \langle \text{GR} \rangle \\
& A \vee B \vee A \vee C \equiv A \vee B \equiv A \vee C \\
\Leftrightarrow & \langle (\text{Leib}) + \text{scramble an } \vee\text{-chain; "Denom:"} \mathbf{p} \equiv A \vee B \equiv A \vee C \rangle \\
& A \vee A \vee B \vee C \equiv A \vee B \equiv A \vee C \\
\Leftrightarrow & \langle (\text{Leib}) + \text{axiom; "Denom:"} \mathbf{p} \vee B \vee C \equiv A \vee B \equiv A \vee C \rangle \\
& A \vee B \vee C \equiv A \vee B \equiv A \vee C
\end{aligned}$$

HERE WE STOP, and try to reach this result from the other side: $A \vee B \wedge C$.

$$\begin{aligned}
& A \vee B \wedge C \\
\Leftrightarrow & \langle (\text{Leib}) + \text{GR; "Denom:"} A \vee \mathbf{p}; \text{mind brackets!} \rangle \\
& A \vee (B \vee C \equiv B \equiv C) \\
\Leftrightarrow & \langle \text{axiom} \rangle \\
& A \vee B \vee C \equiv A \vee (B \equiv C) \\
\Leftrightarrow & \langle (\text{Leib}) + \text{axiom; "Denom:"} A \vee B \vee C \equiv \mathbf{p} \rangle \\
& A \vee B \vee C \equiv A \vee B \equiv A \vee C
\end{aligned}$$

□

Please Read. I Skip.

4.2.19 Theorem. (Important: “Proof by cases”)

$$\vdash A \vee B \rightarrow C \equiv (A \rightarrow C) \wedge (B \rightarrow C)$$

Proof.

$$\begin{aligned}
 & A \vee B \rightarrow C \\
 \Leftrightarrow & \langle 4.2.11 \rangle \\
 & \neg(A \vee B) \vee C \\
 \Leftrightarrow & \langle (Leib) + 4.2.14; \text{“Denom.” } \neg \mathbf{p} \vee C \rangle \\
 & \neg\neg(\neg A \wedge \neg B) \vee C \\
 \Leftrightarrow & \langle (Leib) + \text{double neg.}; \text{“Denom.” } \mathbf{p} \vee C \rangle \\
 & (\neg A \wedge \neg B) \vee C \\
 \Leftrightarrow & \langle 4.2.18 \rangle \\
 & (\neg A \vee C) \wedge (\neg B \vee C) \\
 \Leftrightarrow & \langle \text{obvious } (Leib), \text{ twice } + 4.2.11 \rangle \\
 & (A \rightarrow C) \wedge (B \rightarrow C) \quad \square
 \end{aligned}$$

Until now we only proved absolute theorems Equationally.

► How about theorems with HYPOTHESES?

To do so we use the Redundant \top METAtheorem:

$$\Gamma \vdash A \text{ iff } \Gamma \vdash A \equiv \top$$

The Technique is demonstrated via Examples!

This “trick” converts a Γ -theorem to an equivalence that is a Γ -theorem.

Such equivalences help: They allow the use of Leib!

4.2.20 Example.

- (1) $A, B \vdash A \wedge B$
- (2) $A \vee A \vdash A$
- (3) $A \vdash A \vee B$
- (4) $A \wedge B \vdash A$

For (1):

$$\begin{aligned}
 & A \wedge B \\
 \Leftrightarrow & \langle (Leib) + \text{hyp } B + \text{Red. } \top \text{ META; "Denom:"} \ A \wedge \mathbf{p} \rangle \\
 & A \wedge \top \\
 \Leftrightarrow & \langle 4.2.16 \rangle \\
 & A
 \end{aligned}$$

Bingo!

NOTES:

► $A, B \vdash B$. Hence $A, B \vdash \boxed{B \equiv \top}$

For (2):

$$\begin{array}{c}
 A \\
 \Leftrightarrow \langle \text{axiom} \rangle \\
 A \vee A \quad \text{Bingo!}
 \end{array}$$

For (3):

$$\begin{array}{c}
 A \vee B \\
 \Leftrightarrow \langle (Leib) + \text{Hyp } A + \text{Red-}\top\text{-META; "Denom.:" } \mathbf{p} \vee B \rangle \\
 \top \vee B \quad \quad \quad \langle \text{Bingo!} \rangle
 \end{array}$$

(4) is a bit trickier:

$$\begin{aligned}
 & A \\
 \Leftrightarrow & \langle 4.2.16 \rangle \\
 & A \wedge \top \\
 \Leftrightarrow & \langle (Leib) + \text{Hyp } A \wedge B + \text{Red-}\top\text{-META; "Denom:"} \ A \wedge \mathbf{p} \rangle \\
 & A \wedge A \wedge B \\
 \Leftrightarrow & \langle (Leib) + 4.2.15; \text{"Denom:"} \ \mathbf{p} \wedge B \rangle \\
 & A \wedge B \quad \text{Bingo!} \quad \square
 \end{aligned}$$

I SKIP TO CUT-RULE! You read all :)



4.2.21 Metatheorem. (Hypothesis splitting/merging)

For any wff A, B, C and hypotheses Γ , we have $\Gamma \cup \{A, B\} \vdash C$ iff $\Gamma \cup \{A \wedge B\} \vdash C$.

Proof. (Hilbert-style)

(I) *ASSUME* $\Gamma \cup \{A, B\} \vdash C$ and *PROVE* $\Gamma \cup \{A \wedge B\} \vdash C$.

So, armed with Γ and $A \wedge B$ as *hypotheses* I have to prove C . OK, start!

- (1) $A \wedge B$ $\langle \text{hyp} \rangle$
- (2) A $\langle (1) + A \wedge B \vdash A \text{ rule} \rangle$
- (3) B $\langle (1) + A \wedge B \vdash B \text{ rule} \rangle$
- (4) C $\langle \text{using HYP } \Gamma + (2) \text{ and } (3) \rangle$

(II) *ASSUME* $\Gamma \cup \{A \wedge B\} \vdash C$ and *PROVE* $\Gamma \cup \{A, B\} \vdash C$.

Exercise, or see Text.



4.2.22 Theorem. (Modus Ponens) $A, A \rightarrow B \vdash B$

Proof.

$$\begin{aligned}
 & A \rightarrow B \\
 \Leftrightarrow & \langle \neg\vee\text{-theorem} \rangle \\
 & \neg A \vee B \\
 \Leftrightarrow & \langle (Leib) + \text{hyp } A + \text{Red-}\top\text{-META; "Denom.:" } \neg\mathbf{p} \vee B \rangle \\
 & \neg\top \vee B \\
 \Leftrightarrow & \langle (Leib) + \text{theorem from class; "Denom.:" } \mathbf{p} \vee B \rangle \\
 & \perp \vee B \\
 \Leftrightarrow & \langle \text{thm from class} \rangle \\
 & B
 \end{aligned}$$

□

4.2.23 Theorem. (Cut Rule) $A \vee B, \neg A \vee C \vdash B \vee C$

Proof. We start with an AUXILIARY theorem — a Lemma— which makes the most complex hypothesis $\neg A \vee C$ usable (turns it into an EQUIVALENCE).

$$\begin{aligned} & \neg A \vee C \\ \Leftrightarrow & \langle \text{how to lose a NOT} \rangle \\ & A \vee C \equiv C \end{aligned}$$

Since $\neg A \vee C$ is a HYP hence also a THEOREM, the same is true for $A \vee C \equiv C$ from the Equational proof above. **Remember this below!**

$$\begin{aligned} & B \vee C \\ \Leftrightarrow & \langle (Leib) + \text{Lemma; “Denom:” } B \vee \mathbf{p} \rangle \\ & B \vee (A \vee C) \\ \Leftrightarrow & \langle \text{inserting brackets to our advantage AND swapping wff} \rangle \\ & (A \vee B) \vee C \\ \Leftrightarrow & \langle (Leib) + \text{HYP } A \vee B + \text{Red-}\top\text{-Meta; “Denom:” } \mathbf{p} \vee C \rangle \\ & \top \vee C \quad \text{Bingo!} \quad \square \end{aligned}$$

READ ME! SPECIAL CASES of CUT:

4.2.24 Corollary. $A \vee B, \neg A \vee B \vdash B$

Proof. From 4.2.23 we get $A \vee B, \neg A \vee B \vdash B \vee B$.

We have also learnt the rule $B \vee B \vdash B$.

Apply this rule to the proof above that ends with “ $B \vee B$ ” to get B .

Arrange this wordiness into a neat Hilbert proof! (**Exercise!**) □

4.2.25 Corollary. $A \vee B, \neg A \vdash B$

Proof. Apply the rule $\neg A \vdash \neg A \vee B$.

We now can use the above Corollary! How exactly?

□

4.2.26 Corollary. $A, \neg A \vdash \perp$ **Proof.** Hilbert-style.

- (1) A $\langle \text{hyp} \rangle$
- (2) $\neg A$ $\langle \text{hyp} \rangle$
- (3) $A \vee \perp$ $\langle 1 + \text{rule } X \vdash X \vee Y \rangle$
- (4) $\neg A \vee \perp$ $\langle 2 + \text{rule } X \vdash X \vee Y \rangle$
- (5) \perp $\langle 3 + 4 + \text{rule 4.2.24} \rangle$

Can you do the above Equationally without using CUT?

□

SKIP this PROOF, but memorise the result!

4.2.27 Corollary. (Transitivity of \rightarrow) $A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$

Proof. (Hilbert style)

- (1) $A \rightarrow B$ $\langle \text{hyp} \rangle$
- (2) $B \rightarrow C$ $\langle \text{hyp} \rangle$
- (3) $A \rightarrow B \equiv \neg A \vee B$ $\langle \neg \vee \text{ thm} \rangle$
- (4) $B \rightarrow C \equiv \neg B \vee C$ $\langle \neg \vee \text{ thm} \rangle$
- (5) $\neg A \vee B$ $\langle (1, 3) + (Eqn) \rangle$
- (6) $\neg B \vee C$ $\langle (2, 4) + (Eqn) \rangle$
- (7) $\neg A \vee C$ $\langle (5, 6) + \text{CUT} \rangle$ \square

The last line is provably equivalent to $A \rightarrow C$ by the $\neg \vee$ theorem.

Chapter 5

A Weak Post's Theorem and the Deduction Theorem Retold

Oct. 21, 2024

This note is about the *Soundness* and *Completeness* (the latter is also known as “Post's Theorem”) in Boolean logic.

5.1 Soundness of Boolean Logic

MEMORISE RESULTS (but SKIP PROOFS):

1. SOUNDNESS of Boolean Logic.
2. POST'S THEOREM (completeness of Boolean Logic).
3. DEDUCTION THEOREM.

Soundness is the Property expressed by the statement of the *metatheory* below —which in English says “**Boolean Logic tells ONLY the truth**”:

$$\text{If } \Gamma \vdash A, \text{ then } \Gamma \models_{\text{taut}} A \quad (1)$$

5.1.1 Definition. The statement “**Boolean logic is Sound**” means that Boolean logic satisfies (1).

For $\Gamma = \emptyset$ it means this:

$$\text{If } \vdash A, \text{ then } \models_{\text{taut}} A \quad (2)$$

□

SKIP TO STATEMENT OF POST'S THEOREM (5.2) AND EXAMPLE BEFORE IT.

To prove soundness is an easy induction on the length of Γ -proofs:

We prove that proofs preserve truth.

5.1.2 Lemma. *Eqn and Leib preserve truth, that is,*

$$A, A \equiv B \models_{\text{taut}} B \quad (2)$$

and

$$A \equiv B \models_{\text{taut}} C[\mathbf{p} := A] \equiv C[\mathbf{p} := B] \quad (3)$$

Proof. (2) is trivial.

We prove (3) here:

So, let a state s make $A \equiv B$ true (\mathbf{t}).

Thus,

$$\bar{s}(A) = \bar{s}(B)$$

We will show that for any choice of instances of \mathbf{p}, C we have

$$C[\mathbf{p} := A] \equiv C[\mathbf{p} := B] \text{ is } \mathbf{t} \text{ in state } s \quad (4)$$

If \mathbf{p} is not in C then (4) is $C \equiv C$, a tautology, so is true in ANY STATE, THUS under s in particular.

Let then the distinct $\mathbf{p}, \mathbf{q}, \mathbf{r}, \mathbf{r}', \mathbf{r}'', \dots$ all occur in C .

Now *in the lhs of (4)* \mathbf{p} gets the value $\bar{s}(A)$, while $\mathbf{q}, \mathbf{r}, \mathbf{r}', \mathbf{r}'', \dots$ get their values **DIRECTLY** from s —just as we did with A and B .

Similarly, *in the RHS of (4)* \mathbf{p} gets the value $\bar{s}(B)$, while $\mathbf{q}, \mathbf{r}, \mathbf{r}', \mathbf{r}'', \dots$ STILL get their values DIRECTLY from s .

► But $\bar{s}(A) = \bar{s}(B)$.

So both the lhs and rhs VARIABLES of (4) end up with the same truth value assignments after the indicated substitutions and also directly from s (into $\mathbf{q}, \mathbf{r}, \mathbf{r}', \mathbf{r}'', \dots$).

Then the computed values of lhs and rhs in (4) are the same.

In short, the equivalence is true.



We can now prove:

5.1.3 Metatheorem. *Boolean logic is Sound, that is, (1) on p.162 holds.*

Proof. By induction on the length of proof, n , needed to obtain $\Gamma \vdash A$ we prove

$$\Gamma \models_{\text{taut}} A \quad (\dagger)$$

So pick a state s that satisfies Γ . (\ddagger)

1. *Basis.* $n = 1$. Then we have just A in the proof, and no other wff.

If $A \in \Lambda$, then it is a tautology, so in particular *is true under s* . We have (\dagger).

If $A \in \Gamma$, then s satisfies A by (\ddagger). Again we have (\dagger).

I.H. Assume claim for all proofs of length $\leq n$.

I.S. Prove the theorem in the case $\Gamma \vdash A$ *was established by a proof* of length $n + 1$:

$$\underbrace{\overbrace{\dots}^{\text{length} \leq n}, A, \dots}_{\text{length} = n+1} \quad (\P)$$

Now if A is in $\Lambda \cup \Gamma$ we are back to the Basis. Done.

If not

- Case where A is the result of *EQN* on X and $X \equiv Y$ (so A is the same wff as Y) that are found in the *left* “...-area” of (\P) .

By the I.H. (explain!) s satisfies X and $X \equiv Y$ hence, by the Lemma, satisfies Y , that is, A .

- Case where A is the result of *LEIB* on $X \equiv Y$ that is found in the *left* “...-area” of (\P) .

By the I.H. (explain!) s satisfies $X \equiv Y$ hence, by the Lemma, satisfies A . \square

5.1.4 Corollary. *If $\vdash A$ then $\models_{\text{taut}} A$. A is a tautology.*

Proof. $\Gamma = \emptyset$ here. By the above, $\emptyset \models_{\text{taut}} A$.

BUT, $\emptyset \models_{\text{taut}} A$ says EXACTLY $\models_{\text{taut}} A$ (**EXERCISE**). \square



5.1.5 Example. Soundness allows us to disprove formulas: To show they are NOT theorems.

- The statement “ $\vdash \mathbf{p}$ ” is false. If this were true, then \mathbf{p} would be a tautology!
- The statement “ $\vdash \perp$ ” is false! Because \perp is not a tautology!
- The statement “ $p \vdash p \wedge q$ ” is false. Because if it were true I’d have to have $p \models_{\text{taut}} p \wedge q$.

Not so: Take a state s such that $s(p) = \mathbf{t}$ and $s(q) = \mathbf{f}$.

□



5.2 Completeness of Boolean logic (“Post’s Theorem”)

We prove here

- (1) A weak form of Post’s theorem: If Γ is finite and $\Gamma \models_{\text{taut}} A$, then $\Gamma \vdash A$
 and derive as a corollary the *Deduction Theorem*:
 (2) If $\Gamma, A \vdash B$, then $\Gamma \vdash A \rightarrow B$.

(1) can be formulated as

IF $A_1, \dots, A_n \models_{\text{taut}} B$ THEN $A_1, \dots, A_n \vdash B$

That is,

Every \models_{taut} SCHEMA can be read as a DERIVED RULE.

(2) **Is the way practicing mathematicians prove theorems like $\Gamma \vdash A \rightarrow B$ in everyday work.**

SKIP THIS METAPROOF! (Jump to 5.3.1)

We will employ ONE TOOL from class, *RETOLD* below:

5.2.1 Theorem. $\neg C \vee E, \neg D \vee E \vdash \neg(C \vee D) \vee E$.

Proof. Translating via the “ $\neg\vee$ -theorem”, the above says

$$C \rightarrow E, D \rightarrow E \vdash (C \vee D) \rightarrow E \quad (\dagger)$$

Here is an Equational proof of (\dagger) :

$$\begin{aligned} & (C \vee D) \rightarrow E \\ \iff & \langle 4.2.19 \rangle \\ & (C \rightarrow E) \wedge (D \rightarrow E) \end{aligned}$$

Hence

$$(C \rightarrow E) \wedge (D \rightarrow E) \vdash (C \vee D) \rightarrow E$$

and thus

$$C \rightarrow E, D \rightarrow E \vdash (C \vee D) \rightarrow E$$

by *hypothesis splitting* (4.2.21). □

5.2.2 Main Lemma. (Special Case) *Suppose that A contains none of the symbols $\top, \perp, \rightarrow, \wedge, \equiv$. If $\models_{\text{taut}} A$, then $\vdash A$.*

Proof. The proof is long BUT EASY!

Under the assumption, A is *an \vee -chain*, that is, it has the form

$$A_1 \vee A_2 \vee A_3 \vee \dots \vee A_i \vee \dots \vee A_n \quad (1)$$

where none of the A_i has the form $B \vee C$.

In (1) we assume WITHOUT LOSS OF GENERALITY that $n > 1$, due to the axiom $X \vee X \equiv X$ —that is, *in the contrary case* we can use $A \vee A$ instead, which is a tautology as well and $n = 2$.

Moreover, (1), that is, A , is written in *least parenthesised notation*.

DEFINITION! Let us call an A_i in (1) *reducible* iff it has the form $\neg(C \vee D)$ or $\neg(\neg C)$.



“Reducible”, since A_i is not alone in the \vee -chain, will be synonymous to simplifiable without changing the meaning of A_i , or indeed of A .



Otherwise we say that A_i is *irreducible*. *Not simplifiable.*

Thus, the only possible *irreducible* A_i have the form \mathbf{p} or $\neg \mathbf{p}$ (where \mathbf{p} is a variable).

*By **definition** we will say that A is irreducible iff **all** its A_i are.*



We define the *reducibility degree*, of EACH A_i —in symbols, $rd(A_i)$ — to be the total number, counting repetitions of the \neg and \vee connectives in it, **not counting a possible leading \neg** .



The reducibility degree of the entire A is the sum of the reducibility degrees of all its A_i .

For Example, $rd(p) = 0$, $rd(\neg p) = 0$, $rd(\neg(\neg p \vee q)) = 2$, $rd(\neg(\neg p \vee \neg q)) = 3$, $rd(\neg p \vee q) = 0$.

We say that \mathbf{p} “*occurs positively* in $\dots \vee \mathbf{p} \vee \dots$ ”, while it “*occurs negatively* in $\dots \vee \neg \mathbf{p} \vee \dots$ ”.

In, for example, $\mathbf{p} \vee \neg \mathbf{p}$ it occurs *both* positively and negatively.

By induction on $rd(A)$ we now prove the main lemma, that $\vdash A$ follows the stated hypothesis that $\models_{\text{taut}} A$.

For the *Basis*, let A be an *irreducible* tautology —so, $rd(A) = 0$.



So $rd(A_i) = 0$ for all i



It must be that A is a string of the form

“ $\dots \vee \mathbf{p} \vee \dots \neg \mathbf{p} \vee \dots$ ”

for some \mathbf{p} , otherwise,

if no \mathbf{p} appears both “positively” and “negatively”,

then we can find a truth-assignment that makes A false (**f**) —a contradiction to its *tautologyhood*.

To see that we can do this, just assign **f** to \mathbf{p} ’s that occur *positively only*, and **t** to those that occur *negatively only*.

Now

$$\begin{aligned}
 & A \\
 \Leftrightarrow & \left\langle \text{commuting the terms of an } \vee\text{-chain} \right\rangle \\
 & \mathbf{p} \vee \neg \mathbf{p} \vee B \quad (\text{what is “} B \text{”}?) \\
 \Leftrightarrow & \left\langle \text{Leib} + \text{axiom} + \text{Red. } \top \text{ META; Denom: } \mathbf{r} \vee B; \text{ fresh } \mathbf{r} \right\rangle \\
 & \top \vee B \quad \text{bingo!}
 \end{aligned}$$

Thus $\vdash A$, which settles the *Basis*-case: $rd(A) = 0$.



We now argue the case where $rd(A) = m + 1$,

on the I.H. that for any formula Q with $rd(Q) \leq m$,
we *do* have that $\models_{\text{taut}} Q$ implies $\vdash Q$.



Since we can shuffle an \vee -chain any way we please, we assume *without restricting generality* that $rd(A_1) > 0$.

We have two cases:

(1) A_1 is the string $\neg\neg C$, hence A has the form $\neg\neg C \vee D$.

Clearly $\models_{\text{taut}} C \vee D$ as well.

Moreover, $rd(C \vee D) < rd(\neg\neg C \vee D)$, hence (by I.H.)

$$\vdash C \vee D \quad (\dagger)$$

But,

$$\neg\neg C \vee D$$

$$\Leftrightarrow \left\langle \text{Leib} + \vdash \neg\neg X \equiv X; \text{Denom: } \mathbf{r} \vee D; \text{fresh } \mathbf{r} \right\rangle$$

$$C \vee D \quad \text{“bingo” by } (\dagger) \text{ above!}$$

Hence, $\vdash \neg\neg C \vee D$, that is, $\vdash A$ in this case.

One more case to go:

(2) A_1 is the string $\neg(C \vee D)$, hence A has the form $\neg(C \vee D) \vee E$.

We want: $\vdash \neg(C \vee D) \vee E$ (i)

We are given

$\models_{\text{taut}} \neg(C \vee D) \vee E$ (i')

THAT IS $\models_{\text{taut}} A$.

We immediately get that

$\models_{\text{taut}} \neg C \vee E$ (ii)

and

$\models_{\text{taut}} \neg D \vee E$ (iii)

from truth tables.

Check it!!!

Hint. To show (ii) let v be any state. Consider cases: (1) where $\bar{v}(C) = \mathbf{f}$ and (2) where $\bar{v}(C) = \mathbf{t}$. In the second case use (i') to show $\bar{v}(E) = \mathbf{t}$.

Since the rd of each of (ii) and (iii) is $< rd(A)$, the I.H. yields $\vdash \neg C \vee E$ AND $\vdash \neg D \vee E$.

Apply the TOOL 5.2.1 to the above two theorems to get (i).

We are done, **except for one small detail:**

If we had changed the “original” A into $A \vee A$ (cf. the “without loss of generality” remark just below (1) on p.172), then all we proved above is $\vdash A \vee A$.

The *contraction* rule from Notes, and Text then yield $\vdash A$. □



Do you see now why we wanted $n \geq 2$?



But ALL this only proves “ $\models_{taut} A$ implies $\vdash A$ ”

► when A does *not* contain any of $\wedge, \rightarrow, \equiv, \perp, \top$.

WHAT IF IT DOES?

We are now removing the restriction on A regarding its connectives and constants:

5.2.3 Metatheorem. (Post's Theorem) *If $\models_{\text{taut}} A$, then $\vdash A$.*

Proof. First, we note the following *theorems* stating equivalences, *where \mathbf{p} is fresh for A .*

The proof of the last one is in the notes and text but it was too long (but easy) to cover in class.

$$\begin{aligned}
 \vdash \top &\equiv \neg \mathbf{p} \vee \mathbf{p} \\
 \vdash \perp &\equiv \neg(\neg \mathbf{p} \vee \mathbf{p}) \\
 \vdash C \rightarrow D &\equiv \neg C \vee D \\
 \vdash C \wedge D &\equiv \neg(\neg C \vee \neg D) \\
 \vdash (C \equiv D) &\equiv ((C \rightarrow D) \wedge (D \rightarrow C))
 \end{aligned} \tag{2}$$

Using (2) above, we eliminate, in order, all the \equiv , then all the \wedge , then all the \rightarrow and finally all the \perp and all the \top .

Let us assume that our process eliminates one unwanted symbol at a time.



This leads to *the Equational Proof below*.

Starting from A we will generate a sequence of formulae

$$F_1 (\text{same as } A), F_2, F_3, \dots, F_n (\text{same as } A')$$

where the last, F_n , contains no $\top, \perp, \wedge, \rightarrow, \equiv$.



I am using here F_1 as an alias for A . We will also give to F_n an alias A' .

$$\begin{array}{c}
 A \\
 \Leftrightarrow \langle \text{Leib from (2)} \rangle \\
 F_2 \\
 \Leftrightarrow \langle \text{Leib from (2)} \rangle \\
 F_3 \\
 \Leftrightarrow \langle \text{Leib from (2)} \rangle \\
 F_4 \\
 \vdots \\
 \Leftrightarrow \langle \text{Leib from (2)} \rangle \\
 A' (F_n)
 \end{array}$$

Thus, $\vdash A' \equiv A$ (*)

By **soundness**, we also have $\models_{\text{taut}} A' \equiv A$ (**)

So, say $\models_{\text{taut}} A$. By (**) we have $\models_{\text{taut}} A'$ as well, and by the Main Lemma 5.2.2 we obtain $\vdash A'$.

So bottom line, A' , being a theorem is a “*bingo!*” hence the top line, A , is also a theorem. □



Post's theorem is the “*Completeness Theorem*”[†] of Boolean Logic.

It shows that the syntactic manipulation apparatus — **proofs** — *DOES certify* the “whole truth” (tautologyhood) in the Boolean case.



[†]Which is really a *Metatheorem*, right?

5.2.4 Corollary. *If*

$$\overbrace{A_1, \dots, A_n}^{\text{finite } \Gamma} \models_{\text{taut}} B$$

then

$$A_1, \dots, A_n \vdash B$$

Proof. It is an easy semantic exercise to see that the assumption implies

$$\models_{\text{taut}} A_1 \rightarrow \dots \rightarrow A_n \rightarrow B$$

By 5.2.3,

$$\vdash A_1 \rightarrow \dots \rightarrow A_n \rightarrow B$$

hence (hypothesis strengthening)

$$A_1, A_2, \dots, A_n \vdash A_1 \rightarrow A_2 \rightarrow \dots \rightarrow A_n \rightarrow B \quad (1)$$

Applying modus ponens n times to (1) we get

$$A_1, \dots, A_n \vdash B \quad \square$$



The above corollary is very convenient.

It says that every (correct) schema $A_1, \dots, A_n \models_{\text{taut}} B$ leads to a derived rule of inference, $A_1, \dots, A_n \vdash B$.



In particular, combining with the transitivity of \vdash metatheorem, we get

5.2.5 Corollary. *If $\Gamma \vdash A_i$, for $i = 1, \dots, n$, and if $A_1, \dots, A_n \models_{\text{taut}} B$, then $\Gamma \vdash B$.*



Thus —*unless otherwise required!*— we can, from now on, *rigorously* mix syntactic with semantic justifications of our proof steps.

For example, we have at once $A \wedge B \vdash A$, because (trivially) $A \wedge B \models_{\text{taut}} A$ (compare with our earlier, much longer, proof given in class).



5.3 Deduction Theorem and Proof by Contradiction

5.3.1 Metatheorem. (The Deduction Theorem) *If $\Gamma, A \vdash B$, then $\Gamma \vdash A \rightarrow B$, where “ Γ, A ” means “all the assumptions in Γ , plus the assumption A ” (in set notation this would be $\Gamma \cup \{A\}$).**

Proof. Let $G_1, \dots, G_n \subseteq \Gamma$ be a *finite* set of formulae from Γ that *were used* in a Γ, A -proof of B .

Thus we also have $G_1, \dots, G_n, A \vdash B$.

By *Boolean Soundness*, $G_1, \dots, G_n, A \models_{\text{taut}} B$.

But then,

$$\overbrace{G_1, \dots, G_n}^{\text{finite!}} \models_{\text{taut}} A \rightarrow B \quad \textbf{Verify!}$$

By 5.2.4, $G_1, \dots, G_n \vdash A \rightarrow B$ and hence $\Gamma \vdash A \rightarrow B$ by *hypothesis strengthening*. \square

*We can also write $\Gamma + A$.



The mathematician, or indeed the mathematics practitioner, uses the Deduction theorem all the time, without stopping to think about it. Metatheorem 5.3.1 above makes an honest person of such a mathematician or practitioner.

The everyday “style” of applying the Metatheorem goes like this:

Say we have all sorts of assumptions and we want, *under these assumptions*, to “prove” that “if A , then B ” (verbose form of “ $A \rightarrow B$ ”).

We start by **adding** A to our assumptions, often with the words, “Assume A ”. We then proceed and prove *just* B (not $A \rightarrow B$), and at that point we rest our case.

Thus, we may view an application of the Deduction theorem as a simplification of the proof-task. It allows us to “split” an implication $A \rightarrow B$ that we want to prove, moving its premise to join our other assumptions. We now have to prove a *simpler formula*, B , with the help of *stronger* assumptions (that is, all we knew so far, plus A). That often makes our task so much easier!



An Example. Prove

$$\vdash (A \rightarrow B) \rightarrow A \vee C \rightarrow B \vee C$$

By DThm, suffices to prove

$$A \rightarrow B \vdash A \vee C \rightarrow B \vee C$$

instead.

Again By DThm, suffices to prove

$$A \rightarrow B, A \vee C \vdash B \vee C$$

instead.

Let's do it:

- | | | |
|----|--|---|
| 1. | $A \rightarrow B$ | $\langle \text{hyp} \rangle$ |
| 2. | $A \vee C$ | $\langle \text{hyp} \rangle$ |
| 3. | $A \rightarrow B \equiv \neg A \vee B$ | $\langle \neg \vee \text{ thm} \rangle$ |
| 4. | $\neg A \vee B$ | $\langle 1 + 3 + \text{Eqn} \rangle$ |
| 5. | $B \vee C$ | $\langle 2 + 4 + \text{Cut} \rangle$ |

□

5.3.2 Definition. A set of formulas Γ is *inconsistent* or *contradictory* iff Γ proves every A in WFF. \square



Why “contradictory”? For if Γ proves *everything*, then it also proves the *contradiction* $\mathbf{p} \wedge \neg \mathbf{p}$.



5.3.3 Lemma. Γ is inconsistent iff $\Gamma \vdash \perp$

Proof. only if-part. If Γ is as in 5.3.2, then, in particular, it proves \perp since the latter is a well formed formula.

if-part. Say, conversely, that we have

$$\Gamma \vdash \perp \quad (9)$$

Let now A be any formula in WFF whatsoever. We have

$$\perp \models_{\text{taut}} A \quad (10)$$

Pause. Do you believe (10)?

By Post (5.2), $\Gamma \vdash A$ follows from (9) and (10). \square

5.3.4 Metatheorem. (Proof by contradiction) $\Gamma \vdash A$ iff $\Gamma \cup \{\neg A\}$ is inconsistent.

Proof. if-part. So let (by 5.3.3)

$$\Gamma, \neg A \vdash \perp$$

Hence

$$\Gamma \vdash \neg A \rightarrow \perp \quad (1)$$

by the Deduction theorem. However $\neg A \rightarrow \perp \models_{\text{taut}} A$, hence, by Corollary 5.2.5 and (1) above, $\Gamma \vdash A$.

only if-part. So let

$$\Gamma \vdash A$$

By hypothesis strengthening,

$$\Gamma, \neg A \vdash A \quad (2)$$

Moreover, trivially,

$$\Gamma, \neg A \vdash \neg A \quad (3)$$

Since $A, \neg A \models_{\text{taut}} \perp$, (2) and (3) yield $\Gamma, \neg A \vdash \perp$ via Corollary 5.2.5, and we are done by 5.3.3. \square



5.3.4 legitimises the tool of “proof by contradiction” that goes all the way back to the ancient Greek mathematicians: To prove A assume instead the “opposite”, $\neg A$. Proceed then to obtain a contradiction. This being accomplished, it is as good as having proved A .



Chapter 6

Resolution

Proof by Resolution is an easy and self-documenting 2-dimensional proof style.

It is essentially a Hilbert style proof that needs no numbering due to its graphical presentation, where the *annotation is depicted by drawing certain lines*.

The technique is used in “automatic theorem proving”, i.e., special computer systems (programs) that prove theorems.

It is based on the *proof by contradiction metatheorem*:

6.0.1 Metatheorem.

$$\Gamma, \neg A \vdash \perp \quad (1)$$

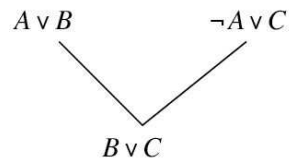
iff

$$\Gamma \vdash A \quad (2)$$

Thus, instead of proving (2) prove (1).

(1) is proved using (almost) exclusively the CUT Rule.

The *self-annotating* diagram below says “apply the CUT rule to premises $A \vee B$ and $\neg A \vee C$ to obtain $B \vee C$ ”.



The technique can be best learnt via examples:

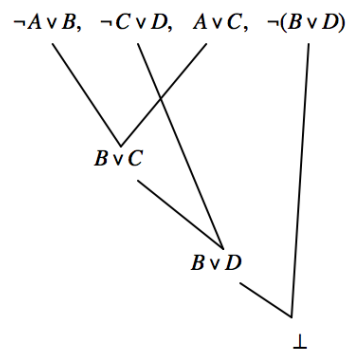
6.0.2 Example. Use Resolution to prove (1) below:

$$A \rightarrow B, C \rightarrow D \vdash A \vee C \rightarrow B \vee D \quad (1)$$

by DThm prove instead:

$$A \rightarrow B, C \rightarrow D, A \vee C \vdash B \vee D$$

By 6.0.1 prove instead that the “ Γ ” in the top line below proves \perp



□

6.0.3 Example. Next prove

$$\vdash (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$$

By the DThm prove instead

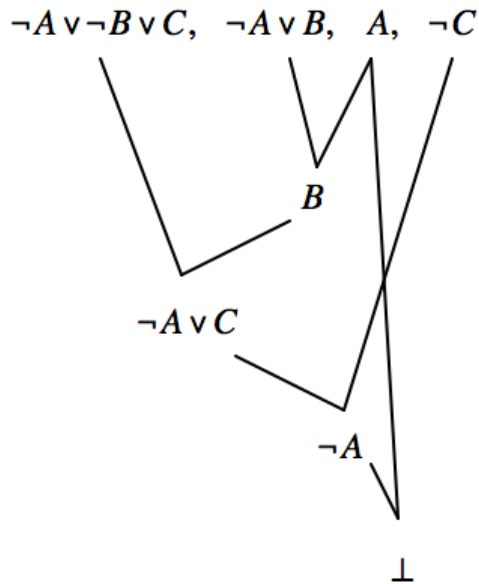
$$A \rightarrow (B \rightarrow C) \vdash (A \rightarrow B) \rightarrow (A \rightarrow C)$$

Two more applications of the DThm simplify what we will prove into the following:

$$A \rightarrow (B \rightarrow C), A \rightarrow B, A \vdash C$$

By 6.0.1, prove instead that $\Gamma \vdash \perp$ where

$$\Gamma = \{\neg A \vee \neg B \vee C, \neg A \vee B, A, \neg C\}$$



□

6.0.4 Example. Prove

$$\vdash (A \wedge \neg B) \rightarrow \neg(A \rightarrow B)$$

By DThm do instead: $A \wedge \neg B \vdash \neg(A \rightarrow B)$.

By 6.0.1 do instead

$$A \wedge \neg B, A \rightarrow B \vdash \perp$$

or

$$A \wedge \neg B, \neg A \vee B \vdash \perp$$

Use HYP Splitting, so do instead

$$A, \neg B, \neg A \vee B \vdash \perp$$

$$A, \neg B, \neg A \vee B$$

To this end, cut 1st and 3rd to get B .

Cut the latter with $\neg B$ to get \perp .

□

6.0.5 Example. *Annotating hypothesis splitting* and equivalence graphically: We do not annotate the equivalence or split lines any more than we annotate the CUT lines!

□

Chapter 7

Predicate Logic

Extending Boolean Logic

Boolean Logic can deal only with the Boolean glue:
properties and behaviour.

Can certify tautologies, but it misses *many other truths*
as we will see, *like $x = x$ where x stands for a mathe-*
matical object like a matrix, string, array, number, etc.

One of the obvious reasons is that Boolean logic can-
not even “see” or “speak” about mathematical objects.



*If it cannot see or speak about them, then naturally cannot reason
about them either!*



E.g, we *cannot even state* inside Boolean logic the sentence “every natural number greater than 1 has a prime factor”.

Boolean Logic *does not know* what “every” means or what a “number” is, what “natural” means, what is “1”, what “greater” means, what “prime” is, or what “factor” is.

In fact it is worse than not “knowing”: It cannot even say any one of the concepts listed above.

Its alphabet and language are extremely limited.

We need a richer language!

7.0.1 Example. Look at these two math statements. The first says that *two sets are equal IF they have the same elements*. The second says that *any object is equal to itself*.

We read “ $(\forall x)$ ” below as “*for all values of x* ”, usually said MORE SIMPLY as, “*for all x* ”.

$$(\forall y)(\forall z) \left((\forall x)(x \in y \equiv x \in z) \rightarrow y = z \right) \quad (1)$$

and

$$x = x \quad (2)$$

Boolean Logic is a *very high level* (= very non-detailed) abstraction of Mathematics.

Since Boolean Logic cannot see object variables x, y, z , cannot see \forall or $=$, *nor can penetrate inside the so-called “scope” of $(\forall z)$* —that is, the big brackets above it myopically *understands* (*perceives*) each of (1) and (2) as atomic statements p and q (not seeing inside the scope it sees NO “glue”).

Thus Boolean logic, if forced to opine about the above it will say none of the above is a theorem (by soundness).

Yet, (1) is an axiom of *Set Theory* and (2) is an *axiom in ALL mathematics*.

Says: “**Every object is equal to itself.**”

□

Enter First-Order Logic or Predicate Logic.

Predicate logic is *the language AND logic* of mathematics and mathematical sciences.

In it we CAN “speak” (1) and (2) above and reason about them.

7.1 The language of First-Order Logic

What symbols are *absolutely necessary* to *include* in the Alphabet, \mathcal{V}_1 —the subscript “1” for “1st-order”— of Predicate Logic?

Well, let us enumerate:

7.1.1 Definition. (The 1st-order alphabet; first part)

1. First of all, we are *EXTENDING*, *NOT* discarding, *Boolean Logic*. So we include in \mathcal{V}_1 *all of Boolean Logic’s symbols*

$$\mathbf{p}, \perp, \top, (,), \neg, \wedge, \vee, \rightarrow, \equiv$$

where \mathbf{p} stands for **any** of the infinitely many Boolean variables.

2. Then we need *object variables*—that is, variables that stand for *mathematical objects*— x, y, z, u, v, w *with or without primes or subscripts*. These are infinitely many.

Metanotation that stands for any of them will be bold face, but using the same letters with or without primes or subscripts: $\mathbf{x}, \mathbf{x}''_5, \mathbf{y}, \mathbf{w}'''_{123}$, etc.

3. *Equality* between mathematical objects: $=$

4. *New glue*: \forall

We call this glue *universal quantifier*. It is pronounced “**for all**”.

Is that all? No. But let's motivate with two examples.

□

7.1.2 Example. (Set theory) The language of set theory needs also a binary relation or *predicate up in front*: Denoted by “ \in ”. BUT nothing else.

All else is “*manufactured*” in the theory, that is, introduced by definitions.

The manufactured symbols include *constants* like our familiar \mathbb{N} (the *set of natural numbers*, albeit set theorists often prefer the symbol “ ω ”), our familiar *constant* “ \emptyset ” (the empty set).

Also include *functions* like \cup, \cap and relations or *predicates* like \subset, \subseteq .

So set theory needs no constants or functions up in front to start “operating” (proving theorems, that is). \square

7.1.3 Example. (Number theory) The language of Number theory —also called *Peano arithmetic*— needs —in order to get started:

- A *constant*, the *number zero*: 0
- A binary *predicate* (“less than”): $<$
- A unary *function*: “ S ”. (This, informally/intuitively is the “successor function” which with input x produces output $x + 1$.)
- Two binary *functions*, “ $+$, \times ” with the obvious meaning.

All else is “*manufactured*” in the theory, that is, *introduced by definitions*.

The manufactured symbols include *constants* like our familiar 1, 2, 1000234000785:

$$S0, SS0, \overbrace{SS \dots S}^{1000234000785 \text{ } S \text{ symbols}} 0$$

Also include *functions* like x^y , $\lfloor x/y \rfloor$ and more relations or *predicates* like \leq . \square

We will do logic for the user, that is, we are aiming to *teach the USE of logic*.

But will do so *without having to learn and do set theory or number theory or any specific mathematical theory* (geometry, algebra, etc.).

So equipped with our observations from the examples above, we note that various theories start up with *DIF-FERENT* sets of constants, functions and predicates — according to their specific needs.

So we will complete the Definition 7.1.1 in a *UNI-FIED* way that *APPLIES TO ANY AREA OF MATHEMATICAL APPLICATION*.

Oct. 30, 2024

7.1.4 Definition. (The 1st-order alphabet; part 2)

Our 1st-order alphabet also includes the following symbols

- (1) Symbols for zero or more *constants*. *Generically*, we use a, b, c, d with or without primes or subscripts for constants.
- (2) Symbols for zero or more *functions*. *Generically* we use f, g, h with or without primes or subscripts for functions.

Each such symbol will have the need for a certain number of arguments, this number called the function's "*arity*" (must be ≥ 1). For example, S has arity 1; it is unary. Each of $+, \times$ have arity two; they are binary.

You see where the word "arity" comes from?

- (3) Symbols for zero or more *predicates*, *generically* denoted as ϕ (" $f\bar{e}$ ", as in "see"), ψ (" $ps\bar{e}$ "), with or without primes or subscripts.

Each predicate symbol will have the need for a certain number of arguments called its "*arity*" (must be ≥ 1). For example, $<$ has arity 2. \square

The first-order *LANGUAGE* is a set of strings of two types —*terms* and *formulas*— over the *alphabet* 7.1.1 AND 7.1.4.

By now we should feel comfortable with *first-order inductive definitions*.

In fact we gave inductive definitions of *first-order Boolean formulas* and used it quite a bit, but also more recently gave an inductive definition of Boolean *proofs*.

Thus we inductively introduce first-order Terms that denote objects —the notation is that of function calls— and first-order formulas, that denote statements, in two separate definitions.

First terms:

7.1.5 Definition. (Terms)

A term is a string over the alphabet \mathcal{V}_1 that satisfies ONE of:

- (1) It is just an *object variable* x (recall that x is metanotation and stands for *any* object variable).



BTW, we drop the qualifier “object” from “object variable” from now on, but *RETAIN* the qualifier “Boolean” in “Boolean variable”.



- (2) An *object constant* a (this stands for any constant —generically).



BTW, we ALSO drop the qualifier “object” from “object constant” from now on, but *RETAIN* the qualifier “Boolean” in “Boolean constant”.



- (3) **General case.** It is a string of the form $ft_1t_2\dots t_n$ where the function symbol f has *arity* n and the t_i **are** (I mean, **STAND FOR**) terms.

As noted already, objects —with the exception of trivial ones, like variables and constants, these are denoted by function calls.

Surprised? Function calls do return as values objects!

We will denote arbitrary terms generically by the meta-symbols t, s with or without primes or subscripts just as we did above. □



We will often abuse notation and write “ $f(t_1, t_2, \dots, t_n)$ ” for “ $ft_1t_2 \dots t_n$ ”.

This is one (rare) case where *the human eye prefers extra brackets!* Be sure to note that the comma “,” is not in our alphabet!



Examples from number theory.

$x, 0$ are terms. $x + 0$ is a term (abuse of the actual “ $+x0$ ” notation).

$(x + y) \times z$ is a term (abuse of the actual $\times + xyz$).

With the concept of terms out of the way we now define 1st-order formulas:

First the Atomic Case:

Reminder: Arbitrary (non specific) terms are denoted by the letters t and s with or without subscripts or primes.

7.1.6 Definition. (1st-order Atomic formulas) The following are the *atomic —that is, glue-less— formulas of 1st-order logic*:

- (i) Any *Boolean* atomic formula.
- (ii) The *expression* (*string*) “ $t = s$ ”, for any choice of t and s (probably, the t and s name the same term).
- (iii) For any predicate ϕ of *arity* n , and any n terms t_1, t_2, \dots, t_n , the string “ $\phi t_1 t_2 \dots t_n$ ”.

We denote *the set of all atomic formulas* here defined by **AF**. □

In practice, we prefer writing $x < y$ (infix) rather than $< xy$ (prefix)



7.1.7 Remark.

(1) As in the case of “complex” terms $ft_1t_2 \dots t_n$, we often abuse notation using “ $\phi(t_1, t_2, \dots, t_n)$ ” in place of the correctly written “ $\phi t_1t_2 \dots t_n$ ”.

(2) The symbol “=” is a binary predicate and is always written as it is here (never “ ϕ, ψ ”).

(3) We absolutely NEVER confuse “=” with the Boolean “glue” “ \equiv ”.

They are more different than apples and oranges! □

7.1.8 Definition. (1st-order formulas) A first-order formula A —or **wff** A — is one of



We let context fend for us as to *what formulas we have in mind when we say “wff”: 1st Order or Boolean?*

From here on it is 1st-order ones!

If we want to talk about Boolean wff we *WILL ALWAYS USE* the qualifier “Boolean”!



- (1) A *member of 1st-order AF set* —in particular it could be a *Boolean* atomic wff!
- (2) $(\neg B)$ if B is a wff.
- (3) $(B \circ C)$ if B and C are wff, and \circ is one of $\wedge, \vee, \rightarrow, \equiv$.
- (4) $((\forall \mathbf{x})B)$, where B is a wff and \mathbf{x} any variable.



TWO things: (1) we already agreed that “variable” means *object variable* otherwise I’d say “Boolean variable”. (2) *Nowhere in the definition —of item (4)— is required that \mathbf{x} occurs in B as a substring.*



We call “ \forall ” the *universal quantifier*.

The configuration $(\forall \mathbf{x})$ is pronounced “for all \mathbf{x} ” — intuitively meaning “*for all values of \mathbf{x}* ” rather than

“for all variables $x, y'', z'''_{1234009}, \dots$ that \mathbf{x} may stand for”.

We say that the part of A between the two large red brackets above is the scope of $(\forall \mathbf{x})$.

Thus the \mathbf{x} in $(\forall \mathbf{x})$ and the entire B are in this scope.

□



The “in particular” observation in case (1) along with the cases (2) and (3) make it clear that *every Boolean wff is also a (1st-order) wff*.

Thus first-order logic can “speak” Boolean (but not the other way around, as we made abundantly clear!)



7.1.9 Example. $x = y$, \perp and p are wff. In fact, Atomic.

The last two are also Boolean wff.

$((\forall x)((\forall y)(\neg x = y)))$ is a wff. Note that \neg in $(\neg x = y)$ *applies to $x = y$* NOT to x !

Glue cannot apply to an object like x . Must apply to a statement (a wff)!

$((\forall y)((\neg x = y) \wedge p))$ and $((\forall y)(\neg x = y) \wedge p)$ are also formulas.

BTW, in the two last examples: *p is in the scope of $(\forall y)$ in the first*, but not so in the second. \square

7.1.10 Definition. (Existential quantifier)

It is convenient —**but NOT NECESSARY**— to introduce the “*existential quantifier*”, \exists .

This is only a *metatheoretical abbreviation* symbol that we introduce by this *Definition*, that is, by a “*naming*”

For any wff A , we define $((\exists \mathbf{x})A)$ to be *a short name for*

$$\left(\neg \left((\forall \mathbf{x}) (\neg A) \right) \right) \quad (1)$$

We pronounce $((\exists \mathbf{x})A)$ “for some (value of) \mathbf{x} , A holds”.

The intuition behind this $((\exists \mathbf{x})A)$ *naming* is captured by the diagram below

$$\left(\overbrace{\neg}^{\text{it is not the case that}} \left(\underbrace{(\forall \mathbf{x})}_{\text{all values of } x} \overbrace{(\neg A)}^{\text{make } A \text{ false}} \right) \right)$$

So SOME value of x makes A TRUE.

The *scope of* $(\exists \mathbf{x})$ in

$$((\exists \mathbf{x})A) \quad (2)$$

is the area between the two red brackets.

In particular, the leftmost \mathbf{x} in (2) is in the scope. \square

Priorities Revisited

We *augment* our priorities *table, from highest to lowest*:

$$\overbrace{\forall, \exists, \neg}^{\text{equal priorities}}, \wedge, \vee, \rightarrow, \equiv$$

Associativities *remain right*! Thus, $\neg(\forall x)\neg A$ is *a short form* of (1) in 7.1.10.

Another example: $(u = v \rightarrow ((\forall x)x = a) \wedge p)$ simplifies into

$$u = v \rightarrow (\forall x)x = a \wedge p$$

More examples:

(2) Instead of $((\forall z)(\neg x = y))$ we write

$$(\forall z)\neg x = y$$

(3) Instead of $((\forall x)((\forall x)x = y))$ we write

$$(\forall x)(\forall x)x = y$$

BOUND vs FREE.

7.1.11 Definition. A variable \mathbf{x} *occurs free* in a wff A iff *it is NOT inside the scope of a $(\forall \mathbf{x})$ or $(\exists \mathbf{x})$* —otherwise it occurs bound.

We say that a bound variable \mathbf{x} in $\boxed{(\forall \mathbf{x})A}$ other than the one in the displayed $(\forall \mathbf{x})$, belongs to the displayed leftmost “ $(\forall \mathbf{x})$ ” iff \mathbf{x} *occurs free in A* —thus it was this leftmost “ $(\forall \mathbf{x})$ ”, which we added to the left of A that *did the bounding!*

The terminology “belongs to” is now clear.

We apply this criterion to *subformulas* of A of the form $(\forall \mathbf{x})(\dots)$ to determine where various bound \mathbf{x} found inside A belong. \square

7.1.12 Example. Consider

$$(\forall x) \overbrace{(x = y \rightarrow (\forall x)x = z))^A$$

Here the red x in A belongs to the red $\forall x$. The black x belongs to the black $\forall x$. \square



7.1.13 Remark. We saw that *a Boolean wff, is also a 1st-order wff.*

We view Boolean formulas as abstractions of 1st-order ones.

How is this Abstraction accomplished?



Well, in any given 1st-order wff we just “hide” all 1st-order features inside certain *glue-less subformulas* that take up *the role of NEW Boolean variables*.



That is, we view any wff among the following three forms as Boolean variables—none of them has exposed glue

These have no exposed glue so are viewed as Boolean **variables**

1. $t = s$
2. $\phi t_1 t_2 t_3 \dots t_n$
3. $(\forall x)A$

WAIT! I do see “glue” in $(\forall x)(A \rightarrow B)$; don’t I ???

No, you don’t **if you are a citizen of Boolean!** The “ \rightarrow ” is hidden inside the scope of $(\forall x)$.

So an inhabitant of Boolean logic has an INTEREST in the above “Boolean variables” if and only if they are connected with *VISIBLE* Boolean glue to form Boolean wff.



Of course, Boolean logic whose job is to certify tautologies —by either truth tables or proofs— has no use for isolated Boolean variables, that is, ones that are not glued to anything!



Examples.

- In Boolean Country you see this “ $x = y \rightarrow x = y \vee x = z$ ” as “ $\boxed{x = y} \rightarrow \boxed{x = y} \vee \boxed{x = z}$ ” where the first and second box is the same —say variable p — while the last one is different. You recognize a tautology!
- You see this “ $x = x$ ” as “ $\boxed{x = x}$ ”. *Just a Boolean variable.*
Not a tautology.
- The same goes for this “ $(\forall x)x = y \rightarrow x = y$ ” which the Boolean citizen views as “ $\boxed{(\forall x)x = y} \rightarrow \boxed{x = y}$ ”, that is, a Boolean wff $p \rightarrow q$. Not a tautology.

Process of abstraction: We only abstract (**that is, we see as “Boolean variables”**) the expressions 1.–3. above in order to turn a 1st-order wff into a Boolean wff.

The three forms above are known in logic as **Prime Formulas**.

More Boolean abstraction examples:

- If A is

$$p \rightarrow x = y \vee (\forall x)\phi x \wedge q \quad (\text{note that } q \text{ is not in the scope of } (\forall x))$$

then we abstract as

$$p \rightarrow \boxed{x = y} \vee \boxed{(\forall x)\phi x} \wedge q \quad (1)$$

so the Boolean citizen sees

$$p \rightarrow p' \vee p'' \wedge q$$



If we ask “*show ALL the prime formulas in A by boxing them*” then we —*who understand 1st-order language and we can see inside scopes*— would have also boxed ϕx above. The Boolean citizen cannot see ϕx in the scope of $(\forall x)$ anyway so the boxing done by such a citizen would be exactly as we gave it in (1)



- First box all prime formulas in (2) below.

$$(\forall x)(x = y \rightarrow (\forall z)z = a \vee q) \quad (2)$$

Here it is.

$$\boxed{(\forall x)(\boxed{x = y} \rightarrow \boxed{(\forall z)\boxed{z = a}} \vee q)}$$

Now abstract the above as if you were a Boolean citizen:

$$\boxed{(\forall x)(x = y \rightarrow (\forall z)z = a \vee q)}$$

You see no glue at all because you cannot see inside the scope of the leftmost $(\forall x)$!

The abstraction is something like

“ p ”

- $x = y \rightarrow x = y$ abstracts as $\boxed{x = y} \rightarrow \boxed{x = y}$. That is, $p \rightarrow p$ — *a tautology*.

Why bother with abstractions? Well, the last example is a tautology so a Boolean citizen can prove it.

However $\boxed{x = x}$ and $\boxed{(\forall x)x = y} \rightarrow \boxed{x = y}$ are not tautologies and we need predicate logic techniques to settle their theoremhood.



Nov. 4, 2024

We can now define:

7.1.14 Definition. (Tautologies and Tautological Implications)

We say that a (1st-order) wff, A , *is a tautology and write $\models_{\text{taut}} A$ iff its Boolean abstraction is.*

In 1st-order Logic $\Gamma \models_{\text{taut}} A$ is applied to the Boolean abstraction of A and to the abstractions of the wff in Γ .

Goes without saying that ALL the *identical* occurrences of *Prime Formulas* ... in $\Gamma \cup \{A\}$ will stand for the same Boolean variable.

For example, $x = y \models_{\text{taut}} x = y \vee z = v$ is correct as we see from

$$\overbrace{\boxed{x = y}}^p \models_{\text{taut}} \overbrace{\boxed{x = y}}^p \vee \overbrace{\boxed{z = v}}^q$$

□



Definition. A 1st-Order formula Q is a Tautology IFF its Boolean Abstraction is.

It is a tautological implication of $\{A, B, C, \dots\}$ iff the Abstractions of A, B, C, \dots tautologically imply the Abstraction of Q .



Nov.4, 2024

Substitutions

A substitution is a *textual substitution*: **Find and Replace.**

In $A[\mathbf{x} := t]$ we will replace all occurrences of a *free* \mathbf{x} in A by the term t : *Find and replace.*

In $A[\mathbf{p} := B]$ we will replace all occurrences of a \mathbf{p} in A by B : *Find and replace.*



7.1.15 Example. (What to avoid) Consider the substitution below

$$\left((\exists x) \neg x = y \right) [y := x]$$

If we go ahead with it *as a brute force “find and replace” asking no questions*, then we are met by a *serious problem*:

The result

$$(\exists x) \neg x = x \tag{1}$$

says *something DIFFERENT than* what the original formula says!

The original $—((\exists x) \neg x = y)—$ says “no matter what the value of y is, there is a *different* x -value”.

The above is true in any application of logic *where we have infinitely many objects*. For example, it is true of real numbers and natural numbers.


On the other hand, (1) is *NEVER* true! It says that there is an object that is *different from itself*! \square



7.1.16 Definition. (Substitution) Each of

1. In $A[\mathbf{x} := t]$ replace all occurrences of a free \mathbf{x} in A by the term t : *Find and replace*.
2. In $A[\mathbf{p} := B]$ replace all occurrences of a \mathbf{p} in A by B : *Find and replace*.

However we *abort* the substitution 1 or 2 if it so happens that going ahead with it

 *forces a free variable \mathbf{y} of t or B to become bound* because *t or B ended up inside the scope of a $(\forall \mathbf{y})$ or $(\exists \mathbf{y})$.*



We say that the substitution is undefined in such cases, and that the reason is that *we had a “free variable capture”*.

There is a variant of substitution 2, above:

3. In $A[\mathbf{p} \setminus B]$ replace all occurrences of a \mathbf{p} in A by B : *Find and replace*.

 For technically justified reasons to be learnt later, *we never abort this one, capture or not.*



We call the substitutions 1. and 2. *conditional* or *constrained*, while the substitution 3. unconditional or *unconstrained*.

There is NO unconditional version of 1.

PRIORITIES (AGAIN!)

$[\mathbf{x} := t], [\mathbf{p} := B], [\mathbf{p} \setminus B]$ have higher priority than all connectives $\forall, \exists, \neg, \wedge, \vee, \rightarrow, \equiv$. They associate from LEFT to RIGHT that is $A[\mathbf{x} := t][\mathbf{p} := B]$ means

$$\left(\left(A[\mathbf{x} := t] \right) [\mathbf{p} := B] \right)$$

□

7.1.17 Example. Several substitutions based on Definition 7.1.16.

$$(1) \text{ } (y = x)[y := x].$$

The red brackets are META brackets. I need them to show the substitution applies to the whole formula.

The result is $x = x$.

(2) $((\forall x)x = y)[y := x]$. By 7.1.16, this is undefined because if I go ahead then x is captured by $(\forall x)$.

(3) $(\forall x)(x = y)[y := x]$. According to priorities, this means $(\forall x)\{(x = y)[y := x]\}$.

That is, “apply the quantifier $(\forall x)$ to $x = x$ ”, **which is all right**.

Result is $(\forall x)x = x$.

(4) $((\forall x)(\forall y)\phi(x, y))[y := x]$. This says

- Do $((\forall x)((\forall y)\phi(x, y)))[y := x]$
- This is all right since y is not free in $((\forall y)\phi(x, y))$
—so *not found; no replace!*

Result is the original formula UNCHANGED.

(5) $\left(z = a \vee ((\forall x)x = y)\right)[y := x]$. *Abort*: x is captured when we attempt substitution in the **subformula** $(\forall x)x = y$.

(6) $\left((\forall x)p\right)[p \setminus x = y]$ Unconditional substitution.
Just find and replace, no questions asked!

Result: $(\forall x)x = y$.

(7) $\left((\forall x)p\right)[p := x = y]$ Undefined. *x in $x = y$ will get captured if you go ahead!* \square

7.1.18 Definition. (Partial Generalisation) We say that B is *a partial generalisation* of A if B is formed *by adding as a PREFIX to A zero or more* strings of the form $(\forall \mathbf{x})$ for any choices whatsoever of the variable \mathbf{x} —*repetitions allowed*. \square

7.1.19 Example. Here is a small list of partial generalisations of the formula $x = z$:

$$x = z,$$

$$(\forall w)x = z,$$

$$(\forall x)(\forall x)x = z,$$

$$(\forall x)(\forall z)x = z,$$

$$(\forall z)(\forall x)x = z,$$

$$(\forall z)(\forall y)(\forall z)(\forall x)(\forall u)x = z.$$

\square



7.2 Axioms and Rules for Predicate Logic

7.2.1 Definition. (1st-Order Axioms) These are **all the partial generalisations** of *all the instances of the following schemata*.

1. All tautologies (e.g., $x = y \rightarrow x = y$ is here)
2. $(\forall \mathbf{x})A \rightarrow A[\mathbf{x} := t]$

 Note that *we get an instance of this schema ONLY IF the substitution is not aborted.* 

3. $A \rightarrow (\forall \mathbf{x})A$ — *PROVIDED \mathbf{x} is not free in A .*
4. $(\forall \mathbf{x})(A \rightarrow B) \rightarrow (\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})B$
5. $\mathbf{x} = \mathbf{x}$
6. $t = s \rightarrow (A[\mathbf{x} := t] \equiv A[\mathbf{x} := s])$

 Note that *we get an instance of this schema ONLY IF none of the substitutions above is aborted.* 

The set of all first-order axioms is named “ Λ_1 ” — “1” for 1st-order.

□

Our only **INITIAL** (or *Primary* or *Primitive*) rule is **Modus Ponens**:

$$\frac{A, A \rightarrow B}{B} \quad (MP)$$

You may think that including all tautologies as axioms is overkill.

However

1. It is customary to do so in the literature ([Tou08, Sho67, End72, Man77, Tou03a])
2. After Post's Theorem we do know that every tautology is a theorem of Boolean logic.

Adopting axiom 1. makes every tautology also a theorem of Predicate Logic outright!

This is the easiest way (a literature favourite) to incorporate Boolean logic as a *sublogic* of 1st-order logic.

7.3 First-order Proofs and Theorems

A Hilbert-style proof from Γ (Γ -proof) is *exactly as defined in the case of Boolean Logic*. Namely:



It is a finite sequence of *wff*

$$A_1, A_2, A_3, \dots, A_i, \dots, A_n$$

such that each A_i is ONE of

1. Axiom from Λ_1 **OR** a member of Γ

OR

2. Is obtained by *MP* from $X \rightarrow Y$ and X that appear to the LEFT of A_i (*A_i is the same string as Y then.*)

However, here “*wff*” is *1st-order*, and Λ_1 is a DIFFERENT set of axioms than the old Λ . Moreover we have ONLY one rule up in front.

As in Boolean definitions, a 1st-order theorem from Γ (Γ -theorem) is *a formula that occurs in a 1st-order Γ -proof*.

As before we write “ $\Gamma \vdash A$ ” to say “ A is a Γ -theorem” and write “ $\vdash A$ ” to say “ A is an absolute theorem”.



Hilbert proofs in 1st-order logic are written vertically as well, with line numbers and annotation.

The following metatheorems about proofs and theorems

- ▶ proof tail removal,
- ▶ proof concatenation,
- ▶ a wff is a Γ -theorem iff it occurs at the end of a proof
- ▶ hypothesis strengthening,
- ▶ hypothesis splitting,
- ▶ Transitivity of \vdash (4.1.11),
and hence
- ▶ usability of derived rules,
- ▶ usability of previously proved theorems

hold with the same metaproofs as in the Boolean case.

We TRIVIALY have Post's Theorem (the weak form that we proved for Boolean logic).

7.3.1 Metatheorem. (Weak Post's Theorem for 1st-order logic)

If $A_1, \dots, A_n \models_{\text{taut}} B$ then $A_1, \dots, A_n \vdash B$

*Proof. Exactly the same as in Boolean logic, see 5.2.4 and 5.2.3, since —as in class— the assumption yields $\models_{\text{taut}} A_1 \rightarrow \dots \rightarrow A_n \rightarrow B$ and hence we have $\vdash A_1 \rightarrow \dots \rightarrow A_n \rightarrow B$, by **Axiom 1** —NO POST USED!*

For the rest see 5.2.4. □



Thus we may use

$$A_1, \dots, A_n \vdash B$$

as a *DERIVED rule in any 1st-order proof, if we know that*

$$A_1, \dots, A_n \models_{\text{taut}} B$$

.

THIS IS DONE A LOT IN Hilbert PROOFS!



7.4 Deduction Theorem

Nov. 6, 2024

This Metatheorem of First-Order Logic says:

7.4.1 Metatheorem. *If $\Gamma, A \vdash B$, then also $\Gamma \vdash A \rightarrow B$*

OR

7.4.2 Metatheorem. *If I want to prove $\Gamma \vdash A \rightarrow B$ it is enough to prove $\Gamma, A \vdash B$ instead.*



WAIT! Did we not already prove this for Boolean Logic? Yes, but to do so we used in an essential way *Boolean Soundness*. See proof of 5.3.1, where we write “By *Boolean Soundness*, etc.”

Boolean Semantics will *NOT help* in Predicate Logic, and First-Order Semantics are tricky, AND DIFFERENT, in particular 1st-order Soundness is totally unlike Boolean Soundness, and we will do it at the end of the course!

So here we use a direct proof by Induction on the *length* of First-Order Proofs from $\Gamma + A$.^{*}



^{*}Recall that “ $\Gamma \cup \{A\}$ ”, “ Γ, A ” and “ $\Gamma + A$ ” are *alternative notations* for the *same* set of wff!

Proof. We do Induction on the proof length L that we used for $\Gamma, A \vdash B$:

1. Proof of length $L = 1$ (*Basis*). There is only one formula in the proof: The proof must be

$$B$$

Only three subcases apply:

- $B \in \Gamma$. Then $\Gamma \vdash B$. Hence $\Gamma \vdash A \rightarrow B$ by Post .
- B IS A . So, $A \rightarrow B$ is a tautology hence axiom hence $\Gamma \vdash A \rightarrow B$.
- $B \in \Lambda_1$. Then $\Gamma \vdash B$. Conclude as in the *first* bullet.

2. *Assume (I.H.) the claim for all proofs of lengths $L \leq n$.*
3. *I.S.:* The proof has length $L = n + 1$:

$$\overbrace{\dots, B}^{n+1}$$

Let $B \in \Gamma \cup \{A\} \cup \Lambda_1$. Done as in the *Basis*.

Assume instead that it is the result of MP on formulas *to the left of B* :

$$\underbrace{\overbrace{\dots, X, \dots, X \rightarrow B, \dots, B}^{\Gamma + A\text{-proof where } L=n+1}}_{\substack{\leq n \\ n}}$$

By the I.H. the metatheorem is true for proofs of length $L \leq n$. Thus we have

$$\Gamma \vdash A \rightarrow X \quad (*)$$

and

$$\Gamma \vdash A \rightarrow (X \rightarrow B) \quad (**)$$

The following Hilbert proof concludes the case and the entire proof:

- | | |
|--------------------------------------|--|
| 1) $A \rightarrow X$ | $\langle \Gamma\text{-thm by } (*) \rangle$ |
| 2) $A \rightarrow (X \rightarrow B)$ | $\langle \Gamma\text{-thm by } (**) \rangle$ |
| 3) $A \rightarrow B$ | $\langle 1 + 2 + \text{Post} \rangle$ |

The last line proves the metatheorem. □

We learn here HOW exactly to handle the quantifier \forall .

7.5 Adding (Removing) “ $(\forall \mathbf{x})$ ” to (from) the beginning of a wff.

7.5.1 Metatheorem. (Weak Generalisation) *Suppose that no wff in Γ has any free occurrences of \mathbf{x} .*

Then if we have $\Gamma \vdash A$, we will also have $\Gamma \vdash (\forall \mathbf{x})A$.

The last line above does **NOT** say that somehow A “implies” $(\forall \mathbf{x})A$.

It rather says that from a proof of A from Γ a proof of $(\forall \mathbf{x})A$ —probably quite different— can be found, *in fact, constructed*.



It is normally the case and of no adverse consequence that A does have free occurrences of \mathbf{x} , else $(\forall \mathbf{x})A$ would be trivial.



Proof. Induction on the length L of the Γ -proof used for A .

1. $L = 1$ (*Basis*). There is only one formula in the proof: The proof must be the 1-wff sequence


$$A$$

Only two subcases apply:

- $A \in \Gamma$. Then A has *no free \mathbf{x}* , *hence* $A \rightarrow (\forall \mathbf{x})A$ is axiom 3. Thus, we have a Hilbert proof (written horizontally for speed),


$$\underbrace{A}_{\Gamma\text{-proved}}, \underbrace{A \rightarrow (\forall \mathbf{x})A}_{\text{axiom}}, \quad \text{MP on the previous two} \quad \underbrace{(\forall \mathbf{x})A}$$

- $A \in \Lambda_1$. Then so is $(\forall \mathbf{x})A \in \Lambda_1$ by partial generalisation:

 Note that if *axiom* A is $(\forall \mathbf{z})(\forall \mathbf{z}') \dots (\forall \mathbf{z}_1) \underbrace{B}_{\text{axiom schema inst.}}$,
then $(\forall \mathbf{x})A$ is $(\forall \mathbf{x})(\forall \mathbf{z})(\forall \mathbf{z}') \dots (\forall \mathbf{z}_1)B$. Hence an
axiom *too*.



Hence $(\forall \mathbf{x})A$ is in Λ_1 , thus $\Gamma \vdash (\forall \mathbf{x})A$ once more.
(*Definition of Γ -proof.*)

 **AHA! So that's what “partial generalisation” does for us!**



2. *Assume (I.H.) the claim for all proofs of lengths $L \leq n$.*
3. *I.S.:* The proof has length $L = n + 1$:

$$\overbrace{\dots, A}^{n+1}$$

If $A \in \Gamma \cup \Lambda_1$ then we are done by the argument in 1.

Assume instead that A is the result of MP on formulas to the left of it:

$$\underbrace{\overbrace{\dots, X, \dots, X \rightarrow A}^{n+1}, \dots, A}_{\substack{\leq n \\ n}}$$

By the I.H. we have

$$\Gamma \vdash (\forall \mathbf{x})X \quad (*)$$

and

$$\Gamma \vdash (\forall \mathbf{x})(X \rightarrow A) \quad (**)$$

The following Hilbert proof concludes this case and the entire proof:

- | | | |
|----|---|--|
| 1) | $(\forall \mathbf{x})X$ | $\langle \Gamma\text{-thm by } (*) \rangle$ |
| 2) | $(\forall \mathbf{x})(X \rightarrow A)$ | $\langle \Gamma\text{-thm by } (**) \rangle$ |
| 3) | $(\forall \mathbf{x})(X \rightarrow A) \rightarrow (\forall \mathbf{x})X \rightarrow (\forall \mathbf{x})A$ | $\langle \text{axiom 4} \rangle$ |
| 4) | $(\forall \mathbf{x})X \rightarrow (\forall \mathbf{x})A$ | $\langle 2 + 3 + \text{MP} \rangle$ |
| 5) | $(\forall \mathbf{x})A$ | $\langle 1 + 4 + \text{MP} \rangle$ |

The last line proves the metatheorem. \square

7.5.2 Corollary. *If $\vdash A$, then $\vdash (\forall \mathbf{x})A$.*

Proof. The condition that no X in Γ has free \mathbf{x} is met: Vacuously. Γ is empty! \square



7.5.3 Remark.

- 1. HOW TO USE Generalisation:** So, the Metatheorem says that if A is a Γ -theorem then so is $(\forall \mathbf{x})A$ as long as the restriction of 7.5.1 is met.

But then, *since I can invoke Γ -THEOREMS* (not only axioms and hypotheses) in a proof, I can insert the Γ -theorem $(\forall \mathbf{x})A$ anywhere AFTER A in any Γ -proof of A where Γ obeys the restriction on \mathbf{x} .

insert at any time after A
 $\dots, A, \dots, \overbrace{(\forall \mathbf{x})A}^{\text{insert at any time after } A}, \dots$


- 2. Why “weak”?** Because I need to know how the A was obtained[†] before I may use $(\forall \mathbf{x})A$. \square



[†]From an \mathbf{x} -less Γ .

7.5.4 Metatheorem. (Specialisation Rule)

$$\left((\forall \mathbf{x}) A \right) \vdash A[\mathbf{x} := t]$$

 Goes without saying that *IF* the expression $A[\mathbf{x} := t]$ is undefined (*in the event of “capture”*), then we have nothing to prove.



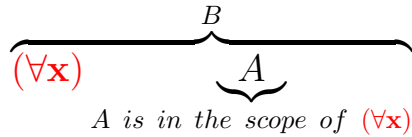
Proof.

- | | | | |
|-----|---|-------------------------------------|-----------|
| (1) | $(\forall \mathbf{x}) A$ | $\langle \text{hyp} \rangle$ | |
| (2) | $(\forall \mathbf{x}) A \rightarrow A[\mathbf{x} := t]$ | $\langle \text{axiom 2} \rangle$ | |
| (3) | $A[\mathbf{x} := t]$ | $\langle 1 + 2 + \text{MP} \rangle$ | \square |

7.5.5 Corollary. $(\forall \mathbf{x}) A \vdash A$

Proof. This is the special case where t is \mathbf{x} . \square

Specialisation removes a $(\forall \mathbf{x})$ iff the quantifier is the *very first symbol*^a of a formula B and the entire remaining part of the formula is the scope of that leading $(\forall \mathbf{x})$:



^a“($\forall \mathbf{x}$)” is ONE compound symbol.

The $(\forall x)$ in the following two CANNOT be removed:
 $(\forall x)A \vee B, \quad A \vee (\forall x)B.$



Really Important! The metatheorems 7.5.5 and 7.5.1 (or 7.5.2) —which we nickname “*spec*” and “*gen*” respectively— are tools that make our life easy in Hilbert proofs where handling of \forall is taking place.

7.5.5 *with no restrictions* allows us to **REMOVE** a leading “ $(\forall \mathbf{x})$ ”.

Doing so *we might uncover Boolean glue* and thus benefit from applications of “Post” (7.3.1).

If we need to *re-INSERT* $(\forall \mathbf{x})$ before the end of proof, we employ 7.5.1 to do so.

This is a good recipe for success in 1st-order proofs!



7.5.1 Examples

**Ping-Pong proofs.**

Hilbert proofs are not well-suited to handle equivalences.

However, trivially

$$A \rightarrow B, B \rightarrow A \models_{\text{taut}} A \equiv B$$

and —by 7.3.1—

$$A \rightarrow B, B \rightarrow A \vdash A \equiv B \quad (1)$$

Thus, *to prove* $\Gamma \vdash A \equiv B$ *in Hilbert style* it suffices —by (1), which is a derived rule!— to offer TWO Hilbert proofs:

$$\underline{\Gamma \vdash A \rightarrow B} \text{ AND } \underline{\Gamma \vdash B \rightarrow A}$$

This back and forth motivates the nickname “ping-pong” for this proof technique.



7.5.2 A Few Memorable Examples

7.5.6 Theorem. (Distributivity of \forall over \wedge)

$$\vdash (\forall \mathbf{x})(A \wedge B) \equiv (\forall \mathbf{x})A \wedge (\forall \mathbf{x})B$$

Proof. By Ping-Pong argument.

We will show TWO things:

$$1. \vdash (\forall \mathbf{x})(A \wedge B) \rightarrow (\forall \mathbf{x})A \wedge (\forall \mathbf{x})B$$

and

$$2. \vdash (\forall \mathbf{x})A \wedge (\forall \mathbf{x})B \rightarrow (\forall \mathbf{x})(A \wedge B)$$

(\rightarrow) (“1.” above)

By DThm, it suffices to prove $(\forall \mathbf{x})(A \wedge B) \vdash (\forall \mathbf{x})A \wedge (\forall \mathbf{x})B$.

- | | | |
|-----|--|--|
| (1) | $(\forall \mathbf{x})(A \wedge B)$ | $\langle \text{hyp} \rangle$ |
| (2) | $A \wedge B$ | $\langle 1 + \text{spec (7.5.5)} \rangle$ |
| (3) | A | $\langle 2 + \text{Post} \rangle$ |
| (4) | B | $\langle 2 + \text{Post} \rangle$ |
| (5) | $(\forall \mathbf{x})A$ | $\langle 3 + \text{gen; OK: hyp contains no free } \mathbf{x} \rangle$ |
| (6) | $(\forall \mathbf{x})B$ | $\langle 4 + \text{gen; OK: hyp contains no free } \mathbf{x} \rangle$ |
| (7) | $(\forall \mathbf{x})A \wedge (\forall \mathbf{x})B$ | $\langle (5,6) + \text{Post} \rangle$ |

NOTE. We *ABSOLUTELY MUST* acknowledge for each application of “gen” that the restriction is met.

(\leftarrow) (“2.” above)

By DThm, it suffices to prove $(\forall \mathbf{x})A \wedge (\forall \mathbf{x})B \vdash (\forall \mathbf{x})(A \wedge B)$.

- | | | |
|-----|--|--|
| (1) | $(\forall \mathbf{x})A \wedge (\forall \mathbf{x})B$ | $\langle \text{hyp} \rangle$ |
| (2) | $(\forall \mathbf{x})A$ | $\langle 1 + \text{Post} \rangle$ |
| (3) | $(\forall \mathbf{x})B$ | $\langle 1 + \text{Post} \rangle$ |
| (4) | A | $\langle 2 + \text{spec} \rangle$ |
| (5) | B | $\langle 3 + \text{spec} \rangle$ |
| (6) | $A \wedge B$ | $\langle (4,5) + \text{Post} \rangle$ |
| (7) | $(\forall \mathbf{x})(A \wedge B)$ | $\langle 6 + \text{gen}; \text{OK: hyp has no free } \mathbf{x} \rangle \square$ |

Easy and Natural! Right?

7.5.7 Theorem. $\vdash (\forall \mathbf{x})(\forall \mathbf{y})A \equiv (\forall \mathbf{y})(\forall \mathbf{x})A$

Proof. By Ping-Pong. $\vdash (\forall \mathbf{x})(\forall \mathbf{y})A \xrightarrow{\rightarrow} (\forall \mathbf{y})(\forall \mathbf{x})A$.

(\rightarrow) direction.

By DThm it suffices to prove $(\forall \mathbf{x})(\forall \mathbf{y})A \vdash (\forall \mathbf{y})(\forall \mathbf{x})A$

- (1) $(\forall \mathbf{x})(\forall \mathbf{y})A$ $\langle \text{hyp} \rangle$
- (2) $(\forall \mathbf{y})A$ $\langle 1 + \text{spec} \rangle$
- (3) A $\langle 2 + \text{spec} \rangle$
- (4) $(\forall \mathbf{x})A$ $\langle 3 + \text{gen; OK hyp has no free } \mathbf{x} \rangle$
- (5) $(\forall \mathbf{y})(\forall \mathbf{x})A$ $\langle 4 + \text{gen; OK hyp has no free } \mathbf{y} \rangle$

(\leftarrow)

Exercise! Justify that you can write the above proof backwards! □



7.5.8 Example. Say A has no free x . Then $\vdash (\forall x)A \equiv A$. Indeed, $\vdash (\forall x)A \rightarrow A$ by ax. 2 and $\vdash A \rightarrow (\forall x)A$ by ax. 3. □



Nov. 11, 2024

7.5.9 Metatheorem. (Monotonicity of \forall) *If $\Gamma \vdash A \rightarrow B$, then $\Gamma \vdash (\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})B$, as long as no wff in Γ has a free \mathbf{x} .*

Proof.

- | | | | |
|-----|---|---|---|
| (1) | $A \rightarrow B$ | $\langle \text{invoking a } \Gamma\text{-thm} \rangle$ | |
| (2) | $(\forall \mathbf{x})(A \rightarrow B)$ | $\langle 1 + \text{gen; OK no free } \mathbf{x} \text{ in } \Gamma \rangle$ | |
| (3) | $(\forall \mathbf{x})(A \rightarrow B) \rightarrow (\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})B$ | $\langle \text{axiom 4} \rangle$ | |
| (4) | $(\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})B$ | $\langle (2, 3) + MP \rangle$ | □ |

We annotate an application of “Monotonicity of \forall ” by either of “A-MON” or “ \forall -MON”.

7.5.10 Corollary. *If $\vdash A \rightarrow B$, then $\vdash (\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})B$.*

Proof. Case of $\Gamma = \emptyset$. The restriction is vacuously satisfied. □

7.5.11 Corollary. *If $\Gamma \vdash A \equiv B$, then also $\Gamma \vdash (\forall \mathbf{x})A \equiv (\forall \mathbf{x})B$, as long as Γ does not contain wff with \mathbf{x} free.*

Proof.

- | | | | |
|-----|---|--|-----------|
| (1) | $A \equiv B$ | $\langle \Gamma\text{-theorem} \rangle$ | |
| (2) | $A \rightarrow B$ | $\langle 1 + \text{Post} \rangle$ | |
| (3) | $B \rightarrow A$ | $\langle 1 + \text{Post} \rangle$ | |
| (4) | $(\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})B$ | $\langle 2 + \forall\text{-MON (7.5.9)} \rangle$ | |
| (5) | $(\forall \mathbf{x})B \rightarrow (\forall \mathbf{x})A$ | $\langle 3 + \forall\text{-MON (7.5.9)} \rangle$ | |
| (6) | $(\forall \mathbf{x})A \equiv (\forall \mathbf{x})B$ | $\langle (4,5) + \text{Post} \rangle$ | \square |

7.5.12 Corollary. *If $\vdash A \equiv B$, then also $\vdash (\forall \mathbf{x})A \equiv (\forall \mathbf{x})B$.*

Proof. Take $\Gamma = \emptyset$.

\square

7.6 Weak Leibniz for 1st-Order Logic

Note that since Post's theorem holds in first-order logic, we have that the *Boolean* primary rules (and all Boolean derived rules; *WHY?*) hold in predicate logic.

For example, the Boolean Leibniz rule

$$A \equiv B \vdash C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$$

holds since we have

$$A \equiv B \models_{\text{taut}} C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$$

What makes the rule “Boolean” is that we look at all of A, B, C and \mathbf{p} from the *Boolean “citizen’s” point of view* (*Boolean abstractions*). In particular, \mathbf{p} is *NOT* in the scope of any quantifier! Because if IT IS, then the Boolean practitioner CANNOT SEE IT, thus *rendering the rule trivial*:

$$\frac{A \equiv B}{C \equiv C}$$



Hmmm. Can I do Leibniz with a \mathbf{p} that is IN the scope of a quantifier? You bet!!



7.6.1 Metatheorem. (Weak (1st-order) Leibniz “WL”)

If $\vdash A \equiv B$, then also $\vdash C[\mathbf{p} \setminus A] \equiv C[\mathbf{p} \setminus B]$.

Proof. This generalises 7.5.12.

The metatheorem is proved by *Induction on the (formation of) wff C* .

Basis. Atomic case for C :

(1) C is \mathbf{p} . The metatheorem boils down to “if $\vdash A \equiv B$, then $\vdash A \equiv B$ ”, which trivially holds!

(2) C is *NOT* \mathbf{p} —that is, it is \mathbf{q} (other than \mathbf{p}), or is \perp or \top , or is $t = s$, or it is $\phi(t_1, \dots, t_n)$. That is, C does not contain the “text” \mathbf{p} .

Then our Metatheorem statement becomes “*if $\vdash A \equiv B$, then $\vdash C \equiv C$* ”.

Given that $\vdash C \equiv C$ is indeed the case by *Axiom 1*, the “if” part is irrelevant. Done.

The complex cases.

- (i) C is $\neg D$. From the I.H. we have $\vdash D[\mathbf{p} \setminus A] \equiv D[\mathbf{p} \setminus B]$,

hence $\vdash \neg D[\mathbf{p} \setminus A] \equiv \neg D[\mathbf{p} \setminus B]$ by Post.

But

$$\text{I want and DO have } \vdash \overbrace{(\neg D)}^C[\mathbf{p} \setminus A] \equiv \overbrace{(\neg D)}^C[\mathbf{p} \setminus B]$$

because

$$(\neg D)[\mathbf{p} \setminus A] \text{ is the same wff as } \neg D[\mathbf{p} \setminus A] \quad \text{WHY?}$$

- (ii) C is $D \circ E$, where $\circ \in \{\wedge, \vee, \rightarrow, \equiv\}$.

The I.H. yields $\vdash D[\mathbf{p} \setminus A] \equiv D[\mathbf{p} \setminus B]$ and $\vdash E[\mathbf{p} \setminus A] \equiv E[\mathbf{p} \setminus B]$ hence

$$\vdash \overbrace{D[\mathbf{p} \setminus A]}^X \circ \overbrace{E[\mathbf{p} \setminus A]}^Y \equiv \overbrace{D[\mathbf{p} \setminus B]}^{X'} \circ \overbrace{E[\mathbf{p} \setminus B]}^{Y'} \text{ by Post.}$$

To see Post relevance, the above follows from this:

Rename:

- $D[\mathbf{p} \setminus A]: X$
- $E[\mathbf{p} \setminus A]: Y$
- $D[\mathbf{p} \setminus B]: X'$
- $E[\mathbf{p} \setminus B]: Y'$

Then $X \equiv X', Y \equiv Y' \models_{\text{taut}} X \circ Y \equiv X' \circ Y'$.

I want **and have**

$$\vdash \overbrace{(D \circ E)}^C[\mathbf{p} \setminus A] \equiv \overbrace{(D \circ E)}^C[\mathbf{p} \setminus B]$$

due to the way substitution works, namely,

$(D \circ E)[\mathbf{p} \setminus A]$ is **the same wff as** $D[\mathbf{p} \setminus A] \circ E[\mathbf{p} \setminus A]$ **WHY?**

(iii) C is $(\forall \mathbf{x})D$. This is the “*interesting case*”.

From the I.H. follows $\vdash D[\mathbf{p} \setminus A] \equiv D[\mathbf{p} \setminus B]$.

From 7.5.12 we get

$$\vdash (\forall \mathbf{x})\left(D[\mathbf{p} \setminus A]\right) \equiv (\forall \mathbf{x})\left(D[\mathbf{p} \setminus B]\right)$$

also written as

$$\vdash \overbrace{((\forall \mathbf{x})D)}^C[\mathbf{p} \setminus A] \equiv \overbrace{((\forall \mathbf{x})D)}^C[\mathbf{p} \setminus B]$$

because

$((\forall \mathbf{x})D)[\mathbf{p} \setminus A]$ is **the same wff as** $(\forall \mathbf{x})\left(D[\mathbf{p} \setminus A]\right)$

□



*WL is the only “Leibniz” we will ever need (practically)
in our use of 1st-order logic in these lectures.*

Why “weak”? Because of the restriction on the Rule’s Hypothesis: $A \equiv B$ must be an absolute theorem. (Recall that the Boolean Leibniz was not so restricted).

SKIP to “END SKIP”, p.263.

Why not IGNORE the restriction and “adopt” the strong rule (i) below?

Well, in logic you do *NOT* arbitrarily “*adopt*” derived rules; you prove them.

BUT, CAN I prove (i) below then?

NO, our logic does not allow it; here is why. If I can prove (i) then I can also prove STRONG generalisation (ii) from (i).

$$A \equiv B \vdash C[\mathbf{p} \setminus A] \equiv C[\mathbf{p} \setminus B] \quad (i)$$

$$\text{strong generalisation: } A \vdash (\forall \mathbf{x})A \quad (ii)$$

But I can PROVE strong generalisation. No?

NO!



Here is why (this is premature, but **important**; hence the  symbol.

We will revisit after we have done 1st-Order Soundness:

1. Let us accept that *all* first-order theorems are “true” everywhere.

2. If I can prove (ii), p.258, then by DThm I can also prove $A \rightarrow (\forall \mathbf{x})A$.

So, this theorem is “true” everywhere, no matter what A and \mathbf{x} are. In particular, if I take z to be \mathbf{x} and A to be $z = 0$ over the natural numbers \mathbb{N} , I must conclude that

$$z = 0 \rightarrow (\forall z)z = 0$$

is true.

Then the following special cases is true

$$\underbrace{\overbrace{0 = 0}^{\text{t}} \rightarrow \overbrace{(\forall z)z = 0}^{\text{f}}}_{\text{f}}$$

is true. But it is NOT!

Tracing this if-then-chain argument backwards I see that I contradicted that I can prove (ii).



That settled, I can now prove that my logic CANNOT prove (i) EITHER.

Why? If it can, then it can prove (ii) of p.258. *Forget it then!*

So, assume I have “Rule” (i). THEN (towards proving (ii))

- | | | |
|-----|---|--|
| (1) | A | $\langle \text{hyp} \rangle$ |
| (2) | $A \equiv \top$ | $\langle (1) + \text{Post} \rangle$ |
| (3) | $(\forall \mathbf{x})A \equiv (\forall \mathbf{x})\top$ | $\langle (2) + (i), \text{p.258}; \text{“Denom.” } (\forall \mathbf{x})\mathbf{p} \rangle$ |
| (4) | $(\forall \mathbf{x})\top \equiv \top$ | $\langle \text{Ax2} + \text{Ax3} + \text{ping-pong}; \text{cf. Example 7.5.8} \rangle$ |
| (5) | $(\forall \mathbf{x})A \equiv \top$ | $\langle (3, 4) + \text{Post} \rangle$ |
| (6) | $(\forall \mathbf{x})A$ | $\langle (5) + \text{Post} \rangle$ |

So if I have (i) I have (ii) too. THUS, I CANNOT have (i), p.258.

► **SKIP** Strong Leibniz in Predicate Logic, but here it is for the curious!

We **CAN** have a MODIFIED (i) where the substitution into **p** is restricted.

7.6.2 Metatheorem. (Strong Leibniz — Acronym “SL”)

$A \equiv B \vdash C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$



Goes without saying that if the rhs of \vdash is *NOT* defined, then there is nothing to prove since the expression “ $C[\mathbf{p} := A] \equiv C[\mathbf{p} := B]$ ” represents no wff.

Remember this comment during the proof!



Proof. As we did for WL, the proof is an induction on the definition/formation of C .

Basis. C is atomic:

subcases

- C **is** \mathbf{p} . We need to prove $A \equiv B \vdash A \equiv B$, which is the familiar $X \vdash X$.
- C **is not** \mathbf{p} . The metatheorem now claims $A \equiv B \vdash C \equiv C$ which is correct since $C \equiv C$ is an axiom.

The complex cases.

- (i) C is $\neg D$. By the I.H. we have $A \equiv B \vdash D[\mathbf{p} := A] \equiv D[\mathbf{p} := B]$, thus, $A \equiv B \vdash \neg D[\mathbf{p} := A] \equiv \neg D[\mathbf{p} := B]$ by Post.

We can rewrite the above as $A \equiv B \vdash (\neg D)[\mathbf{p} := A] \equiv (\neg D)[\mathbf{p} := B]$ since when substitution is allowed

$\overbrace{(\neg D)}^C[\mathbf{p} := A]$ is the **same as** $\neg D[\mathbf{p} := A]$, etc.

- (ii) C is $D \circ E$. By the I.H. we get $A \equiv B \vdash D[\mathbf{p} := A] \equiv D[\mathbf{p} := B]$

and

$$A \equiv B \vdash E[\mathbf{p} := A] \equiv E[\mathbf{p} := B].$$

Thus, by Post,

$$A \equiv B \vdash D[\mathbf{p} := A] \circ E[\mathbf{p} := A] \equiv D[\mathbf{p} := B] \circ E[\mathbf{p} := B]$$

The way substitution works (when defined), the above says

$$A \equiv B \vdash \overbrace{(D \circ E)}^C[\mathbf{p} := A] \equiv \overbrace{(D \circ E)}^C[\mathbf{p} := B]$$

- (iii) C is $(\forall \mathbf{x})D$. This is the “interesting case”.

From the I.H. we get

$$A \equiv B \vdash D[\mathbf{p} := A] \equiv D[\mathbf{p} := B]$$

Now, since the expressions $C[\mathbf{p} := A]$ and $C[\mathbf{p} := B]$ **ARE** defined —**else we wouldn’t be doing all this**— the definition of **conditional** (restricted) substitution implies that neither A nor B have any free

occurrences of \mathbf{x} .

Then \mathbf{x} does not occur free in $A \equiv B$ either.

From 7.5.11 we get

$$A \equiv B \vdash (\forall \mathbf{x})D[\mathbf{p} := A] \equiv (\forall \mathbf{x})D[\mathbf{p} := B]$$

which —the way substitution works— *is the same as*

$$A \equiv B \vdash \overbrace{((\forall \mathbf{x})D)}^C[\mathbf{p} := A] \equiv \overbrace{((\forall \mathbf{x})D)}^C[\mathbf{p} := B]$$

□

END SKIP.

More Memorable Examples and “Techniques”.

7.6.3 Theorem. $\vdash (\forall \mathbf{x})(A \rightarrow B) \equiv (A \rightarrow (\forall \mathbf{x})B)$, as long as \mathbf{x} has no free occurrences in A .

Proof.

Ping-Pong using DThm.

(\rightarrow) I want

$$\vdash (\forall \mathbf{x})(A \rightarrow B) \rightarrow (A \rightarrow (\forall \mathbf{x})B)$$

Better still, let me do (DThm)

$$(\forall \mathbf{x})(A \rightarrow B) \vdash A \rightarrow (\forall \mathbf{x})B$$

and, even better, (DThm!) I will do

$$(\forall \mathbf{x})(A \rightarrow B), A \vdash (\forall \mathbf{x})B$$

- | | | |
|-----|---|--|
| (1) | $(\forall \mathbf{x})(A \rightarrow B)$ | $\langle \text{hyp} \rangle$ |
| (2) | A | $\langle \text{hyp} \rangle$ |
| (3) | $A \rightarrow B$ | $\langle (1) + \text{spec} \rangle$ |
| (4) | B | $\langle (2, 3) + \text{MP} \rangle$ |
| (5) | $(\forall \mathbf{x})B$ | $\langle (4) + \text{gen}; \text{OK: no free } \mathbf{x} \text{ in } (1) \text{ or } (2) \rangle$ |

(\leftarrow) I want

$$\vdash (A \rightarrow (\forall \mathbf{x})B) \rightarrow (\forall \mathbf{x})(A \rightarrow B)$$

or better still (DThm)

$$A \rightarrow (\forall \mathbf{x})B \vdash (\forall \mathbf{x})(A \rightarrow B) \quad (1)$$

Seeing that $A \rightarrow (\forall \mathbf{x})B$ has no free \mathbf{x} , I can prove the even easier

$$A \rightarrow (\forall \mathbf{x})B \vdash A \rightarrow B \quad (2)$$

and *after* this proof is done, then I can apply gen to $A \rightarrow B$ to get $(\forall \mathbf{x})(A \rightarrow B)$.

OK! By DThm I can prove the even simpler than (2)

$$A \rightarrow (\forall \mathbf{x})B, A \vdash B \quad (3)$$

Here it is:

(1)	$A \rightarrow (\forall \mathbf{x})B$	$\langle \text{hyp} \rangle$	
(2)	A	$\langle \text{hyp} \rangle$	
(3)	$(\forall \mathbf{x})B$	$\langle (1, 2) + \text{MP} \rangle$	
(4)	B	$\langle (3) + \text{spec} \rangle$	\square

Nov. 13, 2024

7.6.4 Corollary. *If $\Gamma \vdash A \rightarrow B$ and \mathbf{x} is not free in either Γ or A , then we have also $\Gamma \vdash A \rightarrow (\forall \mathbf{x})B$.*



The operation expressed in the corollary is called “ \forall -Introduction”, in short, “*A-intro*”.



Proof. We have $\Gamma \vdash (\forall \mathbf{x})(A \rightarrow B)$ by Gen (restriction on Γ makes it OK!). Then viewing 7.6.3 as an Equational proof

$$\begin{aligned} & (\forall \mathbf{x})(A \rightarrow B) \\ \Leftrightarrow \langle 7.6.3 \rangle & \\ & A \rightarrow (\forall \mathbf{x})B \end{aligned}$$

we have $\Gamma \vdash A \rightarrow (\forall \mathbf{x})B$. □

7.6.5 Corollary. $\vdash (\forall \mathbf{x})(A \vee B) \equiv A \vee (\forall \mathbf{x})B$, as long as \mathbf{x} does not occur free in A .

Proof.

$$\begin{aligned}
 & (\forall \mathbf{x})(A \vee B) \\
 \Leftrightarrow & \langle \text{WL} + \neg\forall \text{ (axiom, so abs. thm!); "Denom.:" } (\forall \mathbf{x})\mathbf{p} \rangle \\
 & (\forall \mathbf{x})(\neg A \rightarrow B) \\
 \Leftrightarrow & \langle \text{"}\forall \rightarrow\text{" (7.6.3)} \rangle \\
 & \neg A \rightarrow (\forall \mathbf{x})B \\
 \Leftrightarrow & \langle \text{tautology, hence axiom} \rangle \\
 & A \vee (\forall \mathbf{x})B
 \end{aligned}
 \quad \square$$



Most of the statements we prove in what follows have *Dual* counterparts obtained by swapping \forall and \exists and \vee and \wedge .

Let us give a theorem version of the definition of \exists . This is useful in *Equational* proofs in Predicate Logic.

Definition (Recall):

$$(\exists \mathbf{x})A \text{ is short name for } \neg(\forall \mathbf{x})\neg A \quad (1)$$

Next consider the axiom

$$\neg(\forall \mathbf{x})\neg A \equiv \neg(\forall \mathbf{x})\neg A \quad (2)$$

Let me use the *ABBREVIATION* (1) ONLY on *ONE* side of “ \equiv ” in (2). I get the theorem

$$(\exists \mathbf{x})A \equiv \neg(\forall \mathbf{x})\neg A$$

So I can write the theorem without words like this:

$$\vdash (\exists \mathbf{x})A \equiv \neg(\forall \mathbf{x})\neg A \quad (3)$$

HEY! I can apply (3) in Equational proofs —via WL— easily!

I will still refer to (3) in proofs as “Def of E”.



Here's something useful AND good practise too! It is the *Dual* of 7.6.5.

7.6.6 Corollary. $\vdash (\exists \mathbf{x})(A \wedge B) \equiv A \wedge (\exists \mathbf{x})B$, as long as \mathbf{x} does not occur free in A .



In annotation we may call the above the “ $\exists\wedge$ theorem”.



Proof.

$$\begin{aligned}
 & (\exists \mathbf{x})(A \wedge B) \\
 \Leftrightarrow & \langle \text{Def of E} \rangle \\
 & \neg(\forall \mathbf{x})\neg(A \wedge B) \\
 \Leftrightarrow & \langle \text{WL} + \text{axiom 1 (deM)}; \text{“Denom:” } \neg(\forall \mathbf{x})\mathbf{p} \rangle \\
 & \neg(\forall \mathbf{x})(\neg A \vee \neg B) \\
 \Leftrightarrow & \langle \text{WL} + \forall \text{ over } \vee \text{ (7.6.5) —no free } \mathbf{x} \text{ in } \neg A; \text{“Denom:” } \neg\mathbf{p} \rangle \\
 & \neg(\neg A \vee (\forall \mathbf{x})\neg B) \\
 \Leftrightarrow & \langle \mathbf{Ax1 (deM)} \rangle \\
 & A \wedge \neg(\forall \mathbf{x})\neg B \\
 \Leftrightarrow & \langle \text{WL} + \text{Def of E}; \text{“Denom:” } A \wedge \mathbf{p} \rangle \\
 & A \wedge (\exists \mathbf{x})B
 \end{aligned}$$

□

7.7 Ad hoc Memorable Examples

1. While the following theorem —nicknamed “*One-point rule*” — will not play a big role in our lectures, still, on one hand it gives us an example of how we *use the axioms of equality* (Axioms 5 and 6) and on the other hand every mathematician uses it without even thinking about it, in the form, for example,

“ $A(3)$ is the same as $(\exists x)(x = 3 \wedge A(x))$ ”

7.7.1 Theorem. (One point rule — \forall version) *On the condition that \mathbf{x} does not occur in t ,[†] we have*
 $\vdash (\forall \mathbf{x})(\mathbf{x} = t \rightarrow A) \equiv A[\mathbf{x} := t].^{\ddagger}$

Proof. By Ping-Pong.

(\rightarrow) Note that since \mathbf{x} does not occur in t , we have

$$(\mathbf{x} = t \rightarrow A)[\mathbf{x} := t] \quad \text{means the same thing as} \quad t = t \rightarrow A[\mathbf{x} := t]$$

Thus,

$$\begin{array}{ll}
 (1) & (\forall \mathbf{x}) \overbrace{(\mathbf{x} = t \rightarrow A)}^B \rightarrow \overbrace{t = t \rightarrow A[\mathbf{x} := t]}^{B[\mathbf{x} := t]} \quad \langle \mathbf{Ax2} \rangle \\
 (2) & (\forall \mathbf{x}) \mathbf{x} = \mathbf{x} \quad \langle \text{partial gen. of Ax5} \rangle \\
 (3) & t = t \quad \langle (2) + \text{spec} \rangle \\
 (4) & (\forall \mathbf{x})(\mathbf{x} = t \rightarrow A) \rightarrow A[\mathbf{x} := t] \quad \langle (1, 3) + \text{Post} \rangle
 \end{array}$$

[†]We can also say “does not occur free in t ”, but that is an overkill: A term t has NO bound variables.

[‡]Of course, if $A[\mathbf{x} := t]$ is *undefined*, then there is nothing to prove!

(\leftarrow) Recall the **General form of Axiom 6**:

$$\boxed{s = t \rightarrow (A[\mathbf{x} := s] \equiv A[\mathbf{x} := t])}$$

- (1) $\mathbf{x} = t \rightarrow (A \equiv A[\mathbf{x} := t])$ $\langle \mathbf{Ax6} \rangle$
 (2) $A[\mathbf{x} := t] \rightarrow \mathbf{x} = t \rightarrow A$ $\langle (1) + \text{Post} \rangle$

Re above step: $p \rightarrow (q \equiv r) \models_{\text{taut}} r \rightarrow p \rightarrow q$

- (3) $A[\mathbf{x} := t] \rightarrow (\forall \mathbf{x})(\mathbf{x} = t \rightarrow A)$ $\langle (2) + \text{A-Intro 7.6.4} \rangle$

2.

7.7.2 Corollary. (One point rule $\text{---}\exists$ version) *On the condition that \mathbf{x} does not occur in t , we have*
 $\vdash (\exists \mathbf{x})(\mathbf{x} = t \wedge A) \equiv A[\mathbf{x} := t]$.

Proof. Exercise! (*Hint.* Use the \forall version and an Equational proof to prove the \exists version (use the “Def of E” Theorem).) \square

7.7.3 Theorem. (Bound variable renaming (\forall)) IF
 \mathbf{z} is fresh for A —that is, does not occur as either free
or bound in A — then
 $\vdash (\forall \mathbf{x})A \equiv (\forall \mathbf{z})A[\mathbf{x} := \mathbf{z}]. \leftarrow$ Read this right: “ $(\forall \mathbf{z})A(\mathbf{z})$ ”

But NOT our notation!

$$\begin{array}{ll}
(1) & (\forall \mathbf{x})A \rightarrow A[\mathbf{x} := \mathbf{z}] \quad \langle \mathbf{Ax2} \text{ —fresh } \mathbf{z}; \text{ no capture: NO} \\
& \quad \text{“}(\forall \mathbf{z})(\dots, \mathbf{x}, \dots)\text{” in } A \rangle \\
(2) & (\forall \mathbf{x})A \rightarrow (\forall \mathbf{z})A[\mathbf{x} := \mathbf{z}] \quad \langle 1 + 7.6.4; \Gamma = \emptyset \rangle
\end{array}$$

7.7.4 Lemma. *Under the assumptions about \mathbf{z} (freshness), we have that $A[\mathbf{x} := \mathbf{z}][\mathbf{z} := \mathbf{x}]$ is just the *original* A .*

- *Bound* in A . That is, there is NO “ $(\forall \mathbf{z}).(\dots)$ ” in A . So the substitution $A[\mathbf{x} := \mathbf{z}]$ *GOES THROUGH, AND* “flags” (and replaces) ALL FREE \mathbf{x} in A as \mathbf{z} .

Notes on Logic© *G. Tzourlakis*

- *Free* in A . So NO FREE z *pre-existed* in A before doing $A[x := z]$. That is, ALL FREE z in $A[x := z]$ are EXACTLY the FORMER x that BECAME z .
These z are PLACEHOLDERS for THE ORIGINAL FREE x in A .

BUT then! Doing now $[z := x]$ changes ALL z in $A[x := z]$ back to x .

We are back to the original A !

□

- $$\begin{array}{ll}
 (1) & (\forall z) \overbrace{A[x := z]}^B \rightarrow \overbrace{A[x := z][z := x]}^B \quad \langle \mathbf{Ax2} - A[x := z][z := x] \\
 & \text{OK by lemma} \rangle \\
 (1') & (\forall z) A[x := z] \rightarrow A \quad \langle \text{same as (1) —see lemma} \rangle \\
 (2) & (\forall z) A[x := z] \rightarrow (\forall x) A \quad \langle (1') + 7.6.4; \Gamma = \emptyset \rangle \quad \square
 \end{array}$$

7.8 Adding and Removing the Quantifier “ $(\exists x)$ ”

First, introducing (adding) \exists is easy via the following tools:

7.8.1 Theorem. (Dual of Ax2) $\vdash A[\mathbf{x} := t] \rightarrow (\exists \mathbf{x})A$

Proof.

$$\begin{aligned}
 & A[\mathbf{x} := t] \rightarrow \overbrace{\neg(\forall \mathbf{x})\neg A}^{\text{says “}(\exists \mathbf{x})A\text{”}} \\
 \Leftrightarrow & \langle \text{tautology} \rangle \\
 & (\forall \mathbf{x})\neg A \rightarrow \neg A[\mathbf{x} := t] \quad \text{Bingo! } \square
 \end{aligned}$$

7.8.2 Corollary. (The Dual of Specialisation)

$$A[\mathbf{x} := t] \vdash (\exists \mathbf{x})A$$

Proof. $A[\mathbf{x} := t]$ plus 7.8.1 and MP. \square

7.8.3 Corollary. $A \vdash (\exists \mathbf{x})A$

Proof. 7.8.2, taking \mathbf{x} as t . \square



Either corollaries above we call “*Dual Spec*” in annotating proofs.



But how can I remove a leading (the entire formula)
 \exists ?

We need one preliminary result to answer this.

7.8.4 Corollary. (\exists Introduction or “E-Intro”)

IF \mathbf{x} does not occur free in Γ nor in B , then $\Gamma \vdash A \rightarrow B$ IMPLIES $\Gamma \vdash (\exists \mathbf{x})A \rightarrow B$.

Proof. (Hilbert-style)

- 1) $A \rightarrow B$ $\langle \Gamma\text{-thm} \rangle$
- 2) $\neg B \rightarrow \neg A$ $\langle 1 + \text{Post} \rangle$
- 3) $\neg B \rightarrow (\forall \mathbf{x})\neg A$ $\langle 2 + \text{A-intro (7.6.4); no free } \mathbf{x} \text{ in } \Gamma \text{ and } B \rangle$
- 4) $\neg(\forall \mathbf{x})\neg A \rightarrow B$ $\langle 3 + \text{Post} \rangle$

Line (4) says that $(\exists \mathbf{x})A \rightarrow B$ is a Γ -theorem. \square

7.8.5 Metatheorem. (Aux. **Hypothesis** Metatheorem)

Suppose that $\Gamma \vdash (\exists \mathbf{x})A$.

Moreover, suppose that we know that

$$\Gamma + \overbrace{A[\mathbf{x} := \mathbf{z}]}^{\text{Aux. Hyp.}} \vdash B$$

where \mathbf{z} is fresh for ALL of: $\left\{ \begin{array}{l} \Gamma \\ (\exists \mathbf{x})A \\ \text{and } B \end{array} \right.$

Then we have $\Gamma \vdash B$.



In our annotation we call $A[\mathbf{x} := \mathbf{z}]$ an “**auxiliary HYPOTHESIS associated with —or “for”— $(\exists \mathbf{x})A$** ”. \mathbf{z} is called the auxiliary variable that we chose.

Essentially the fact that we proved $(\exists \mathbf{x})A$ allows us to adopt $A[\mathbf{x} := \mathbf{z}]$ as a NEW AUXILIARY HYPOTHESIS to help in the proof of B .

► How does it help? (1) I have a new hypothesis to work with; (2) $A[\mathbf{x} := \mathbf{z}]$ has **NO LEADING QUANTIFIER**.

(2), in general, results in uncovering the Boolean structure of $A[\mathbf{x} := \mathbf{z}]$ to enable proof by “Post”!

Stop-and-Take-Notice: Important! $A[\mathbf{x} := \mathbf{z}]$ is an **ADDED HYPOTHESIS!**

► It is **NOT TRUE** that either $(\exists \mathbf{x})A \vdash A[\mathbf{x} := \mathbf{z}]$ or that $\Gamma \vdash A[\mathbf{x} := \mathbf{z}]$. ◀

WE WILL PROVE LATER IN THE COURSE THAT SUCH A THING IS NOT TRUE!



Proof. of the Metatheorem.

By the DThm, the metatheorem assumption yields

$$\Gamma \vdash A[\mathbf{x} := \mathbf{z}] \rightarrow B$$

Thus, by \exists -Intro (7.8.4) we get

$$\Gamma \vdash (\exists \mathbf{z}) A[\mathbf{x} := \mathbf{z}] \rightarrow B \quad (1)$$

A two-line Equational proof

$$\begin{aligned} & (\exists \mathbf{z}) A[\mathbf{x} := \mathbf{z}] \rightarrow B \\ \Leftrightarrow & \langle \text{WL} + 7.7.3; \text{Denom: } \mathbf{p} \rightarrow B \rangle \\ & (\exists \mathbf{x}) A \rightarrow B \end{aligned}$$

now yields

$\Gamma \vdash (\exists \mathbf{x}) A \rightarrow B$ <p>hence (see our main assumptions)</p> $\Gamma \vdash B$
--

□

The most frequent form encountered in using Metatheorem 7.8.5 is the following corollary.

7.8.6 Corollary. *To prove $(\exists \mathbf{x})A \vdash B$ IT SUFFICES to*

- *pick a \mathbf{z} that is FRESH for $(\exists \mathbf{x})A$ and B and*
- **PROVE INSTEAD $(\exists \mathbf{x})A, A[\mathbf{x} := \mathbf{z}] \vdash B$.**

Proof. Take $\Gamma = \{(\exists x)A\}$ and invoke Metatheorem 7.8.5.

□

7.8.7 Example. $\vdash (\exists \mathbf{x})(\forall \mathbf{y})A \rightarrow (\forall \mathbf{y})(\exists \mathbf{x})A$.

Proof. By DThm it suffices to prove instead:

$$(\exists \mathbf{x})(\forall \mathbf{y})A \vdash (\forall \mathbf{y})(\exists \mathbf{x})A$$

- (1) $(\exists \mathbf{x})(\forall \mathbf{y})A$ $\langle \text{hyp} \rangle$
- (2) $(\forall \mathbf{y})A[\mathbf{x} := \mathbf{z}]$ $\langle \text{aux. hyp for (1); } \mathbf{z} \text{ fresh} \rangle$
- (3) $A[\mathbf{x} := \mathbf{z}]$ $\langle (2) + \text{spec} \rangle$
- (4) $(\exists \mathbf{x})A$ $\langle (3) + \text{Dual spec: } B[\mathbf{x} := t] \vdash (\exists \mathbf{x})B \rangle$
- (5) $(\forall \mathbf{y})(\exists \mathbf{x})A$ $\langle (4) + \text{gen; OK, all hyp lines, (1,2), have no free } \mathbf{y} \rangle$

We used the Corollary 7.8.6 of Metatheorem 7.8.5. \square

Nov. 20, 2024

7.8.8 Example. We prove $(\exists \mathbf{x})(A \rightarrow B), (\forall \mathbf{x})A \vdash (\exists \mathbf{x})B$.

- | | | |
|-----|---|--|
| (1) | $(\exists \mathbf{x})(A \rightarrow B)$ | $\langle \text{hyp} \rangle$ |
| (2) | $(\forall \mathbf{x})A$ | $\langle \text{hyp} \rangle$ |
| (3) | $A[\mathbf{x} := \mathbf{z}] \rightarrow B[\mathbf{x} := \mathbf{z}]$ | $\langle \text{aux. } \underline{\text{hyp}} \text{ for (1); } \mathbf{z} \text{ fresh} \rangle$ |
| (4) | $A[\mathbf{x} := \mathbf{z}]$ | $\langle (2) + \text{spec} \rangle$ |
| (5) | $B[\mathbf{x} := \mathbf{z}]$ | $\langle (3, 4) + MP \rangle$ |
| (6) | $(\exists \mathbf{x})B$ | $\langle (5) + \text{Dual spec} \rangle$ |

□

7.8.9 Example. We prove $(\forall \mathbf{x})(A \rightarrow B), (\exists \mathbf{x})A \vdash (\exists \mathbf{x})B$.

- | | | | |
|-----|---|--|-----------|
| (1) | $(\forall \mathbf{x})(A \rightarrow B)$ | $\langle \text{hyp} \rangle$ | |
| (2) | $(\exists \mathbf{x})A$ | $\langle \text{hyp} \rangle$ | |
| (3) | $A[\mathbf{x} := \mathbf{z}]$ | $\langle \text{aux. hyp for (2); } \mathbf{z} \text{ fresh} \rangle$ | |
| (4) | $A[\mathbf{x} := \mathbf{z}] \rightarrow B[\mathbf{x} := \mathbf{z}]$ | $\langle (1) + \text{spec} \rangle$ | |
| (5) | $B[\mathbf{x} := \mathbf{z}]$ | $\langle (3, 4) + MP \rangle$ | |
| (6) | $(\exists \mathbf{x})B$ | $\langle (5) + \text{Dual spec} \rangle$ | \square |



7.8.10 Example. Here is a common mistake people make when arguing informally.

Let us prove the following informally.

$$\vdash (\exists \mathbf{x})A \wedge (\exists \mathbf{x})B \rightarrow (\exists \mathbf{x})(A \wedge B).$$

So let $(\exists \mathbf{x})A(\mathbf{x})$ and $(\exists \mathbf{x})B(\mathbf{x})$ be true.[†]

Thus, for some value c of \mathbf{x} we have that $A(c)$ and $B(c)$ are true.

But then so is $A(c) \wedge B(c)$.

The latter implies the truth of $(\exists \mathbf{x})(A(\mathbf{x}) \wedge B(\mathbf{x}))$.

Nice, crisp and short.

And very, very wrong as we will see once we have **1st-order Soundness** in hand. Namely, we will show in the near future that $(\exists \mathbf{x})A \wedge (\exists \mathbf{x})B \rightarrow (\exists \mathbf{x})(A \wedge B)$ *is NOT* a theorem schema. It is NOT provable.

[†]The experienced mathematician considers self-evident and unworthy of mention at least two things:

- (1) The deduction theorem, and
- (2) The Split Hypothesis metatheorem.

What went wrong above?

We said

“Thus, for some value c of \mathbf{x} we have that $A(c)$ and $B(c)$ are true”.

The blunder was to assume that **THE SAME c verified BOTH $A(x)$ and $B(x)$.**

Let us see that formalism protects even the inexperienced from such blunders.

Here are the first few steps of a(n attempted) FORMAL proof via the Deduction theorem:

- (1) $(\exists \mathbf{x})A \wedge (\exists \mathbf{x})B$ $\langle \text{hyp} \rangle$
- (2) $(\exists \mathbf{x})A$ $\langle (1) + \text{Post} \rangle$
- (3) $(\exists \mathbf{x})B$ $\langle (1) + \text{Post} \rangle$
- (4) $A[\mathbf{x} := \mathbf{z}]$ $\langle \text{aux. hyp for (2); } \mathbf{z} \text{ fresh} \rangle$
- (5) $B[\mathbf{x} := \mathbf{w}]$ $\langle \text{aux. hyp for (3); } \mathbf{w} \text{ fresh} \rangle$

The requirement of freshness makes \mathbf{w} DIFFERENT from \mathbf{z} . These variables play the role of two distinct c and c' . Thus the proof **cannot be continued**. Saved by freshness!

□



7.8.11 Example. The last Example in this section makes clear that the Russell Paradox was the result of applying *bad Logic*, not just *bad Set Theory*!

I will prove that for any binary predicate ϕ we have

$$\vdash \neg(\exists \mathbf{y})(\forall \mathbf{x})(\phi(\mathbf{x}, \mathbf{y}) \equiv \neg\phi(\mathbf{x}, \mathbf{x})) \quad (R)$$

By the Metatheorem “Proof by Contradiction” I can show

$$(\exists \mathbf{y})(\forall \mathbf{x})(\phi(\mathbf{x}, \mathbf{y}) \equiv \neg\phi(\mathbf{x}, \mathbf{x})) \vdash \perp$$

instead. Here it is

- | | | |
|-----|--|--|
| (1) | $(\exists \mathbf{y})(\forall \mathbf{x})(\phi(\mathbf{x}, \mathbf{y}) \equiv \neg\phi(\mathbf{x}, \mathbf{x}))$ | $\langle \text{hyp} \rangle$ |
| (2) | $(\forall \mathbf{x})(\phi(\mathbf{x}, \mathbf{z}) \equiv \neg\phi(\mathbf{x}, \mathbf{x}))$ | $\langle \text{aux. hyp for (1); } \mathbf{z} \text{ fresh} \rangle$ |
| (3) | $\phi(\mathbf{z}, \mathbf{z}) \equiv \neg\phi(\mathbf{z}, \mathbf{z})$ | $\langle (2) + \text{spec} \rangle$ |
| (4) | \perp | $\langle (3) + \text{Post} \rangle$ |

If we let the atomic formula $\phi(\mathbf{x}, \mathbf{y})$ be Set Theory’s “ $\mathbf{x} \in \mathbf{y}$ ” then (R) that we just proved (in fact *for ANY* binary predicate ϕ not just \in) morphs into

$$\vdash \neg(\exists \mathbf{y})(\forall \mathbf{x})(\mathbf{x} \in \mathbf{y} \equiv \mathbf{x} \notin \mathbf{x}) \quad (R')$$

In plain English (R') says that *there is NO set \mathbf{y} that contains ALL x satisfying $x \notin x$.*

This theorem was proved without using even a single axiom of set theory, indeed not even using “ $\{\dots\}$ -notation” for sets, or any other symbols from set theory.

After all we proved (R') generally and abstractly in the form (R) and that expression and its proof has NO SYMBOLS from set theory!

In short, Russell’s Paradox can be expressed AND demonstrated in PURE LOGIC.

It is remarkable that Pure Logic can tell us that NOT ALL COLLECTIONS are SETS, a fact that escaped Cantor! \square

Semantics of First-Order Languages —Simplified

7.9 Interpretations

Systematically translate an *abstract* formula —*symbol by symbol*— until it becomes a *concrete* mathematical formula in an area of MATH familiar to you.

In this translation ensure that there are no free variables, so the mathematical formula evaluates *exactly as ONE* of true or false.

HOW do I translate? (Read On!)

An *interpretation* of *ONE* wff —and of *THE ENTIRE language*, that is, the set of *ALL* Terms and wff— is INHERITED from an **interpretation of all symbols of the Alphabet**.

This tool —*the Interpretation*— Translates *each wff* to some formula of a familiar branch of mathematics that

we choose, and thus questions such as “*is the translated formula true?*” can in principle be dealt with (see 7.9.2 below for details).

An interpretation is totally up to us, just as *states* were in Boolean logic but the process is a bit more complex.

Here we need to interpret not only wff but *also terms as well*.

The latter requires that we *choose* a *NONEMPTY* set of *objects* to begin with. We call this set the *Domain* of our Interpretation and *generically* call it “*D*” but in specific cases it could be $D = \mathbb{N}$ or $D = \mathbb{R}$ (the *reals*) or even something “small” like $D = \{0, 5\}$.



An *Interpretation* of a 1st-order language consists of a *PAIR* of *two things*:

The aforementioned domain D and a translation mapping M —the latter translates the abstract symbols of the Alphabet of logic to concrete mathematical symbols.

► This translation of the ALPHABET INDUCES a translation for each *term* and *wff* of the language; thus of ALL THE LANGUAGE. ◄

We denote the interpretation “package” as $\mathfrak{D} = (D, M)$ displaying the two ingredients D and M in round brackets.

The unusual calligraphy here is German capital letter calligraphy that is usual in the printed literature *to name an interpretation package*.

On the chalk board I would use ordinary calligraphy, like “ \mathcal{D} ”.

The package name chosen is usually the same as that of the Domain.



Let me *stress* that both D and M are OUR choice.



7.9.1 Definition. (Translating the Alphabet \mathcal{V}_1)

An *Interpretation* $\mathfrak{D} = (D, M)$ gives concrete *counterparts* (translations) to ALL elements of the Alphabet as follows:

In the listed cases below we may use notation $M(X)$ to indicate the concrete translation (mapping) of an abstract *linguistic symbol* X .

We also may use $X^{\mathfrak{D}}$ as an alternative notation for $M(X)$.



The literature favours $X^{\mathfrak{D}}$ and so will we.



Here are the actual translation RULES:

- (1) For each *FREE* variable (of a wff) \mathbf{x} , $\mathbf{x}^{\mathfrak{D}}$ —that is, the translation $M(\mathbf{x})$ — is some *chosen* (BY US!) *FIXED* member of D .

 *BOUND variables are NOT translated! They stay AS IS.* 

- (2) For each Boolean variable \mathbf{p} , $\mathbf{p}^{\mathfrak{D}}$ is a member of $\{\mathbf{t}, \mathbf{f}\}$ that *WE CHOOSE!*

- (3) $\top^{\mathfrak{D}} = \mathbf{t}$ and $\perp^{\mathfrak{D}} = \mathbf{f}$.

This is just as we did —via states— in the Boolean case. As was the case there we choose the value $\mathbf{p}^{\mathfrak{D}}$ *any-way we please*, but for \top and \perp we follow the fixed (Boolean) rule.

- (4) For any (object) *constant* of the alphabet, say, c , we choose a *FIXED* $c^{\mathfrak{D}}$, as we wish, in D .

- (5) For every *function* symbol f of the alphabet, the translation $f^{\mathfrak{D}}$ is a *mathematical function* of the “real” or “concrete” MATH. It has the same arity as f . $f^{\mathfrak{D}}$ —*which WE choose!*— *takes inputs* from D and *gives outputs* in D .

- (6) For every predicate ϕ of the alphabet OTHER THAN “=”, our CHOSEN translation $\phi^{\mathfrak{D}}$ is a mathematical RELATION of the metatheory with the same arity as ϕ . It takes its inputs from D while its outputs are one or the other of the truth values \mathbf{t} or \mathbf{f} .

► NOTE THAT ALL the Boolean glue as well as the equality symbol translate exactly as THEMSELVES: “=” for “equals”, \vee for “OR”, etc.

Finally, brackets translate as the SAME TYPE of bracket (left or right).



We have all we need now to translate wff, terms and thus the entire Language:

7.9.2 Definition. (The Translation of wff)

Consider a wff A in \mathbf{a}^\dagger first-order language.

Suppose we have chosen an interpretation $\mathfrak{D} = (D, M)$ of the alphabet.

The interpretation or translation of A via \mathfrak{D} *is a mathematical (“concrete”) formula of the metatheory* or a concrete object of the metatheory that we will denote by

$$A^{\mathfrak{D}}$$

It is constructed as follows one symbol at a time, scanning A from left to right until no symbol is left:

[†]A, not THE. For every choice of constant, predicate and function symbols we get a different alphabet, as we know, hence a different first-order language. Remember the examples of Set Theory vs. Peano Arithmetic!

- (i) We replace every occurrence of \perp, \top in A by $\perp^{\mathfrak{D}}, \top^{\mathfrak{D}}$ —that is, by \mathbf{f}, \mathbf{t} — respectively.
- (ii) We replace every occurrence of \mathbf{p} in A by $\mathbf{p}^{\mathfrak{D}}$ —this is an *assigned by US TRUTH VALUE*; we assigned it *when we translated the alphabet*.
- (iii) We replace each *FREE* occurrence of an object variable \mathbf{x} of A by the value $\mathbf{x}^{\mathfrak{D}}$ from D that we chose to assign *when we translated the alphabet*.
- (iv) We replace every occurrence of $(\forall \mathbf{x})$ in A by $(\forall \mathbf{x} \in D)$, which means *AND is read* “for all values of \mathbf{x} in D ”.
- (iv') We replace every occurrence of $(\exists \mathbf{x})$ in A by $(\exists \mathbf{x} \in D)$, which means *AND is read* “for some value of \mathbf{x} in D ”.
- (v) We emphasise again that Boolean connectives (glue) *translate as themselves*, and so do “=” and the brackets “(” and “)”.

Theory-specific symbols in A :

- (vi) We replace every occurrence of a(n object) constant c in A by the specific fixed $c^{\mathfrak{D}}$ from D —which we chose when translating the alphabet.
- (vii) We replace every occurrence of a function f in A by the specific fixed $f^{\mathfrak{D}}$ —which we chose when translating the alphabet.
- (viii) We replace every occurrence of a predicate ϕ in A by the specific fixed $\phi^{\mathfrak{D}}$ —which we chose when translating the alphabet. \square

7.9.3 Definition. (Partial Translation of a wff) Given a wff A in a first-order language and an interpretation \mathfrak{D} of the alphabet.

Sometimes *we do NOT wish to translate a FREE variable* \mathbf{x} of A . Then the result of the translation that *leaves \mathbf{x} as is* is denoted by $A_{\mathbf{x}}^{\mathfrak{D}}$.

Similarly, if we *choose NOT* to translate *ANY* of

$$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, \dots$$

that (*may*) occur FREE in A , then we show the result of such “*partial*” translation as

$$A_{\mathbf{x}_1, \dots, \mathbf{x}_n, \dots}^{\mathfrak{D}}$$

Thus $A^{\mathfrak{D}}$ has no free variables, but $A_{\mathbf{x}}^{\mathfrak{D}}$ will have \mathbf{x} free IF \mathbf{x} actually DID occur free in A —the notation guarantees that if \mathbf{x} so occurred, then we left it alone.

□

7.9.4 Remark. $A^{\mathfrak{D}}$ has no free variables. $A_{\mathbf{x}}^{\mathfrak{D}}$ has exactly one, \mathbf{x} . We can write—as in algebra— $A_{\mathbf{x}}^{\mathfrak{D}}(\mathbf{x})$ to depict the input variable \mathbf{x} .

Correspondingly, $A_{\mathbf{x}}^{\mathfrak{D}}(k)$ depicts the result of assigning (inputing!) the value $i \in D$ into \mathbf{x} .

We do *NOT* assign values to the subscript $A_{\mathbf{x}}^{\mathfrak{D}}(\mathbf{x})$ since its symbolic meaning is “I am not translated”; *NOT* “give me inputs!”

What is the need for the concept and notation “ $A_{\mathbf{x}}^{\mathfrak{D}}$ ”?

Well, for one, note that when we translate $(\forall \mathbf{x})A$ *FROM LEFT TO RIGHT*, we get “ $(\forall \mathbf{x} \in D)$ ” *followed by the translation of A* .

However, ANY \mathbf{x} symbols that occur *free IN A BELONG* to $(\forall \mathbf{x})$ *in the wff $(\forall \mathbf{x})A$* thus are *NOT FREE* in the latter and hence are NOT translated!

Therefore, “ $(\forall \mathbf{x} \in D)$ ” concatenated with “ $A_{\mathbf{x}}^{\mathfrak{D}}$ ” is what we get: “ $(\forall \mathbf{x} \in D)A_{\mathbf{x}}^{\mathfrak{D}}$ ”. \square

Study ALL Examples! But I will skip trivial ones.

7.9.5 Example. Consider the AF $\phi(x, x)$, ϕ is a binary predicate.

Here are some possible interpretations:

(a) $D = \mathbb{N}$, $\phi^{\mathfrak{D}} = <$.


Here “ $<$ ” is the “less than” relation on natural numbers.

So $(\phi(x, x))^{\mathfrak{D}}$, which is the same as $\phi^{\mathfrak{D}}(x^{\mathfrak{D}}, x^{\mathfrak{D}})$ —in familiar notation is the formula over \mathbb{N} :

$$x^{\mathfrak{D}} < x^{\mathfrak{D}}$$

More specifically, if we took $x^{\mathfrak{D}} = 42$, then $(\phi(x, x))^{\mathfrak{D}}$ is specifically “ $42 < 42$ ”.

Incidentally, $(\phi(x, x))^{\mathfrak{D}}$ is false for ANY choice of $x^{\mathfrak{D}}$.

 We will write $(\phi(x, x))^{\mathfrak{D}} = \mathbf{f}$ to denote the above sentence symbolically.

This practice in “real” (informal) mathematics is not unlike the expression $t = s$ in formal logic. The latter states an equality, but each of t and s may be syntactically complex terms, although s could be *just a constant*.

Example, $\overbrace{\cos(2\pi)}^t = \overbrace{1}^s$.



For the sake of practice, here are two partial *interpretations*.

In the first we exempt the variables y, z . In the second we exempt x :

(i) $\left(\phi(x, x)\right)_{y,z}^{\mathfrak{D}}$ is $x^{\mathfrak{D}} < x^{\mathfrak{D}}$. WHY?

(ii) $\left(\phi(x, x)\right)_x^{\mathfrak{D}}$ is $x < x$.

(b) $D = \mathbb{N}$, $\phi^{\mathfrak{D}} = \leq$ (the “less than or equal” relation on \mathbb{N}).

So, $\left(\phi(x, x)\right)^{\mathfrak{D}}$ is the concrete $x^{\mathfrak{D}} \leq x^{\mathfrak{D}}$ on \mathbb{N} .

Clearly, independently of the choice of $x^{\mathfrak{D}}$, we have

$$\left(\phi(x, x)\right)^{\mathfrak{D}} = \mathbf{t}$$

□

7.9.6 Example. Consider next the wff

$$f(x) = f(y) \rightarrow x = y \quad (1)$$

where f is a unary function.

Here are some interpretations:

1. $D = \mathbb{N}$ and $f^{\mathfrak{D}}$ is chosen to be $f^{\mathfrak{D}}(x) = x + 1$, for all values of x in D .

Thus $(f(x) = f(y) \rightarrow x = y)^{\mathfrak{D}}$ translates as this formula over \mathbb{N} :

$$\begin{aligned} f^{\mathfrak{D}}(x^{\mathfrak{D}}) &= f^{\mathfrak{D}}(y^{\mathfrak{D}}) \rightarrow x^{\mathfrak{D}} = y^{\mathfrak{D}} \\ x^{\mathfrak{D}} + 1 &= y^{\mathfrak{D}} + 1 \rightarrow x^{\mathfrak{D}} = y^{\mathfrak{D}} \end{aligned}$$

Note that *every* choice of $x^{\mathfrak{D}}$ and $y^{\mathfrak{D}}$ makes the above true.

2. $D = \mathbb{Z}$, where \mathbb{Z} is the set of all integers,
 $\{\dots, -2, -1, 0, 1, 2, \dots\}$

Take $f^{\mathfrak{D}}(x) = x^2$, for all x in \mathbb{Z} .

Then, $(f(x) = f(y) \rightarrow x = y)^{\mathfrak{D}}$ is, more concretely, the following formula over \mathbb{Z} :

$$(x^{\mathfrak{D}})^2 = (y^{\mathfrak{D}})^2 \rightarrow x^{\mathfrak{D}} = y^{\mathfrak{D}}$$

The above is true for some choices of $x^{\mathfrak{D}}$ and $y^{\mathfrak{D}}$ but not for others:

E.g., it is false if we took $x^{\mathfrak{D}} = -2$ and $y^{\mathfrak{D}} = 2$.

Finally here are two *partial interpretations* of (1) at the beginning of this example:

$$(i) \left(f(x) = f(y) \rightarrow x = y \right)_x^{\mathfrak{D}} \text{ is } x^2 = (y^{\mathfrak{D}})^2 \rightarrow x = y^{\mathfrak{D}}.$$

$$(ii) \left(f(x) = f(y) \rightarrow x = y \right)_{x,y}^{\mathfrak{D}} \text{ is } x^2 = y^2 \rightarrow x = y. \quad \square$$

7.9.7 Example. The following is the *equality axiom* (known as **Axiom of Extensionality**) of set theory.

$$(\forall y)(\forall z) \left((\forall x)(x \in y \equiv x \in z) \rightarrow y = z \right)$$

We can interpret this over the natural number interpretation $\mathfrak{N} = (\mathbb{N}, M)$ given by $\in^{\mathfrak{N}} = <$. I.e., interpret \in as $<$ on \mathbb{N} . **Can I do this?** You bet!

I get the translation (interpretation):

$$(\forall y \in \mathbb{N})(\forall z \in \mathbb{N}) \left((\forall x \in \mathbb{N})(x < y \equiv x < z) \rightarrow y = z \right) \quad (1)$$

Is (1) true? **Yes!** Indeed if for each of the natural numbers y and z we obtain the sets of their predecessors, $\{x : x < y\}$ and $\{x : x < z\}$, and find them equal—as (1) claims to the left of “ \rightarrow ”—then $y = z$.

Hint: We can have *neither* of $y < z$ *nor* $z < y$. Can you argue that to the end? **Exercise!** \square

Nov. 25, 2024



7.9.8 Example. (Important!) Consider the wff

$$x = y \rightarrow (\forall x)x = y \quad (1)$$

Here are a few interpretations:

1. $D = \{3\}$, $x^{\mathfrak{D}} = 3$, $y^{\mathfrak{D}} = 3$.

Since D contains one element only the above “choice”
is ALL we HAVE, being unique.

Thus (1) translates as

$$3 = 3 \rightarrow (\forall x \in D)x = 3 \quad (2)$$

Incidentally, (2) is TRUE.

2. This time I take

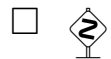
$D = \{3, 5\}$, and again $x^{\mathfrak{D}} = 3$ and $y^{\mathfrak{D}} = 3$.

Thus (1) translates as:

$$3 = 3 \rightarrow (\forall x \in D)x = 3 \quad (3)$$

This time (3) is FALSE since “ $3 = 3$ ” is TRUE as before, BUT

“($\forall x \in D$) $x = 3$ ” is FALSE.



7.9.9 Example. (7.9.7 Revisited) Let's interpret in $\mathfrak{I} = (D, M)$ the following in a two different ways:

$$(\forall x)(x \in y \equiv x \in z) \rightarrow y = z \quad (1)$$

1. Let $D = \mathbb{N}$ and $\in^{\mathfrak{D}} = |$, where “|” indicates the relation “*divides*” (with remainder zero).

E.g., $2 | 3$ and $2 | 1$ are FALSE but $2 | 4$ and $2 | 0$ are TRUE.

Then (1) translates as:

$$(\forall x \in \mathbb{N})(x | y^{\mathfrak{D}} \equiv x | z^{\mathfrak{D}}) \rightarrow y^{\mathfrak{D}} = z^{\mathfrak{D}}$$

which is also TRUE for all choices of $y^{\mathfrak{D}}, z^{\mathfrak{D}}$.

It says: “*Two natural numbers, $y^{\mathfrak{D}}$ and $z^{\mathfrak{D}}$, are EQUAL if they have exactly the same divisors*”.

2. But consider something slightly different now: Take $D = \mathbb{Z}$ —the set of all *integers*— and $\in^{\mathfrak{D}} = |$. Take also $y^{\mathfrak{D}} = 2$ and $z^{\mathfrak{D}} = -2$.

Then (1) translates as

$$(\forall x \in \mathbb{Z})(x | 2 \equiv x | -2) \rightarrow 2 = -2$$

This is FALSE, for 2 and -2 have the same divisors, but $2 \neq -2$.

So (1) is NOT TRUE IN ALL INTERPRETATIONS.

□

7.10 Soundness in Predicate Logic

7.10.1 Definition. (Universally Valid wff)

Suppose that $A^{\mathfrak{D}} = \mathbf{t}$ for some A and \mathfrak{D} .

We say that A *is true in the interpretation \mathfrak{D}* or that \mathfrak{D} *is a model of A* .

We write this thus:

$$\models_{\mathfrak{D}} A \quad (1)$$

A 1st-order wff, A , is *universally valid*—or just “*valid*”—iff *EVERY interpretation of the wff is a model of it*, that is, we have that (1) *holds for every interpretation \mathfrak{D} of the language of A* .

In symbols,

$$A \text{ is valid iff, for all } \mathfrak{D}, \text{ we have } \models_{\mathfrak{D}} A \quad (2)$$

(2) has the short expression (3) below (i.e., drop the subscript from \models):

$$\models A \quad (3)$$

A formula A that satisfies (3) is sometimes also called Logically or Absolutely valid. □

SKIP the Soundness Proof.

7.10.2 Metatheorem. (Soundness of Predicate Logic)

If $\vdash A$, then $\models A$.

We omit the trivial proof by induction on proof length.

For length one we NOTE that the ONLY formula that appears in the proof is an axiom. But that is valid!

The induction step notes that our *ONLY PRIMARY[†] rule*, MP, preserves truth.

[†]Given up in front.



7.10.3 Example. (Strong Gen; Again!) Can our logic prove *strong generalisation* as a “derived rule”?

Namely, can we have If $\Gamma \vdash A$, then $\Gamma \vdash (\forall \mathbf{x})A$, **with NO restriction on** Γ relating to \mathbf{x} ? If yes, take $\Gamma = \{A\}$.[†] We get

$$A \vdash (\forall \mathbf{x})A \quad (1)$$

By the DThm, (1) allows this:

$$\vdash A \rightarrow (\forall \mathbf{x})A \quad (2)$$

Soundness **OBJECTS** to (2):

If we got (2) then, by Soundness, we get

$$\models A \rightarrow (\forall \mathbf{x})A \quad (3)$$

I will **contradict** (3) by showing

$$\not\models A \rightarrow (\forall \mathbf{x})A \quad (4)$$

The Definition of “ \models ” (7.10.1) **(4) dictates** that I find **just ONE** \mathfrak{D} such that

$$(A \rightarrow (\forall \mathbf{x})A)^{\mathfrak{D}} = \mathbf{f} \quad (5)$$



This \mathfrak{D} is called a **countermodel** of (2).



[†]Then $A \vdash A$, hence $A \vdash (\forall \mathbf{x})A$.

PRACTICAL ADVISE: It is hopeless to search for a countermodel \mathfrak{D} *FOR A GENERAL A*:

Some instances of A do have one and some don't, in general.

For a *countermodel* I ONLY need a SPECIFIC A (a countermodel is a counterexample!)

► Always work with an *atomic* formula in place of A.

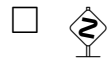
Now then! Take A to be atomic, for example, take A to be “ $x = y$ ”
 If (3) works, it should work with *this* special case of A!

DOES IT?

NO. We saw in Example 7.9.8(2.) (cf. Definition 7.10.1)

$$\not\models x = y \rightarrow (\forall x)x = y$$

So (2) is wrong and so is (1).



7.10.4 Example. We have proved in class/NOTES/Text

$$\vdash (\exists \mathbf{y})(\forall \mathbf{x})A \rightarrow (\forall \mathbf{x})(\exists \mathbf{y})A$$

We hinted in class that we cannot also prove

$$\vdash (\forall \mathbf{x})(\exists \mathbf{y})A \rightarrow (\exists \mathbf{y})(\forall \mathbf{x})A \quad (1)$$

To show that (1) is unprovable I pick a countermodel
(=an interpretation that makes the wff in it false).

Pick A to be something simple. Atomic is best!

I take $D = \mathbb{N}$ and $\mathbf{x} = \mathbf{y}$ for A . Translating the wff in (1)
I note

$$\overbrace{(\forall \mathbf{x} \in \mathbb{N})(\exists \mathbf{y} \in \mathbb{N})\mathbf{x} = \mathbf{y}}^{\text{t}} \rightarrow \overbrace{(\exists \mathbf{y} \in \mathbb{N})(\forall \mathbf{x} \in \mathbb{N})\mathbf{x} = \mathbf{y}}^{\text{f}}$$

Since the interpretation falsifies a special case of (1)
the latter is not provable (by soundness). \square

7.10.5 Example. We we cannot prove

$$\vdash (\forall \mathbf{x})(A \vee B) \rightarrow (\forall \mathbf{x})A \vee (\forall \mathbf{x})B \quad (1)$$

To demonstrate this fact we use Soundness, and construct a countermodel \mathfrak{D} so that we get

$$\not\models_{\mathfrak{D}} (\forall \mathbf{x})(A \vee B) \rightarrow (\forall \mathbf{x})A \vee (\forall \mathbf{x})B \quad (2)$$

Pick A and B to be something simple. Atomic is best!

I take $D = \mathbb{N}$ and “ $\mathbf{x} < 42$ ” for A while I take “ $\mathbf{x} \geq 42$ ” for B . Translating the wff in (1) I note

$$\overbrace{(\forall \mathbf{x} \in \mathbb{N})(\mathbf{x} < 42 \vee \mathbf{x} \geq 42)}^{\text{t}} \rightarrow \overbrace{\underbrace{(\forall \mathbf{x} \in \mathbb{N})\mathbf{x} < 42}_{\text{f}} \vee \underbrace{(\forall \mathbf{x} \in \mathbb{N})\mathbf{x} \geq 42}_{\text{f}}}_{\text{f}}$$

I established (2) via the chosen \mathfrak{D} . Thus (1) is not a theorem (by soundness). \square



7.10.6 Example. The statement $\vdash (\exists x)A \rightarrow A[x := c]$ where c is a constant is *invalid!*

We need a countermodel \mathfrak{D} for a special instance of A .

Well, take $\mathfrak{D} = (\mathbb{N}, M)$ and let “ A ” be $x = y$.

Take


$$c^{\mathfrak{D}} = 1$$

$$y^{\mathfrak{D}} = 2. \text{ We then have}$$

$$\overbrace{(\exists x \in \mathbb{N})x = 2}^{\text{t}} \rightarrow \overbrace{1 = 2}^{\text{f}}$$

$$\begin{array}{ccc} & \uparrow & \uparrow \\ & c^{\mathfrak{D}} & y^{\mathfrak{D}} \end{array}$$

By the DThm, $(\exists x)A \vdash A[x := c]$ is invalid too.

EXERCISE: Prove that $(\exists x)A \rightarrow A[x := z]$ (z fresh) is NOT a theorem either. \square 



Wait! “I thought that ‘ $(\exists x)A(x)$ ’ says, ‘for some value c of x , $A(c)$ is true’. How come we contradicted this?”

Because the singly-quoted part is ‘broken English’. We should have said instead:

“Say $(\exists x)A(x)$ be true. **LET then c be a value that makes $A(c)$ true**”. Or, “some c ” verifies $A(x)$ ”. In symbols (that are not allowed!!!) this is *the same as* “ $(\exists x)A(x)$ ”; says “ $(\exists c)A(c)$ ”, which is NOT NEW!!





7.10.7 Example. (Important!) Why is $D \neq \emptyset$ important?

Well let us start by proving

$$\vdash (\forall \mathbf{x})A \rightarrow (\exists \mathbf{x})A \quad (1)$$

Use DThm to prove instead

$$(\forall \mathbf{x})A \vdash (\exists \mathbf{x})A$$

- 1) $(\forall \mathbf{x})A$ $\langle \text{hyp} \rangle$
- 2) A $\langle 1 + \text{spec} \rangle$
- 3) $(\exists \mathbf{x})A$ $\langle 2 + \text{Dual spec} \rangle$

However, if I took $\mathfrak{D} = (D, M)$ with $D = \emptyset$ then look at the transaction of the formula in (1):

$$\underbrace{(\forall x)(x \in D \rightarrow A_x^{\mathfrak{D}})}_{\text{t vacuously}} \stackrel{(\dagger)}{\rightarrow} \underbrace{(\exists x)(x \in D \wedge A_x^{\mathfrak{D}})}_{\text{f}} \quad (2)$$

So soundness fails for the formula in (1). *We DON'T like this! So we NEVER allow $D = \emptyset$.*

□



[†]Do not forget that “ $(\forall \mathbf{x} \in D)A_{\mathbf{x}}^{\mathfrak{D}}$ ” means “ $(\forall \mathbf{x})(\mathbf{x} \in D \rightarrow A_{\mathbf{x}}^{\mathfrak{D}})$ ”, while “ $(\exists \mathbf{x} \in D)A_{\mathbf{x}}^{\mathfrak{D}}$ ” means “ $(\exists \mathbf{x})(\mathbf{x} \in D \wedge A_{\mathbf{x}}^{\mathfrak{D}})$ ”.

Bibliography

- [Bar78] Jon Barwise, *An introduction to first-order logic*, Handbook of Mathematical Logic (Jon Barwise, ed.), North-Holland, Amsterdam, 1978, pp. 5–46.
- [DS90] Edsger W. Dijkstra and Carel S. Scholten, *Predicate Calculus and Program Semantics*, Springer-Verlag, New York, 1990.
- [End72] Herbert B. Enderton, *A Mathematical Introduction to Logic*, Academic Press, New York, 1972.
- [GS94] David Gries and Fred B. Schneider, *A Logical Approach to Discrete Math*, Springer-Verlag, New York, 1994.
- [Jec78] T. J. Jech, *Set theory*, Academic Press, New York, 1978.
- [Lev79] A. Levy, *Basic Set Theory*, Springer-Verlag, New York, 1979.
- [Man77] Yu. I. Manin, *A Course in Mathematical Logic*, Springer-Verlag, New York, 1977.
- [Sho67] Joseph R. Shoenfield, *Mathematical Logic*, Addison-Wesley, Reading, MA, 1967.

- [Tou03a] G. Turlakis, *Lectures in Logic and Set Theory, Volume 1: Mathematical Logic*, Cambridge University Press, Cambridge, 2003.
- [Tou03b] ———, *Lectures in Logic and Set Theory, Volume 2: Set Theory*, Cambridge University Press, Cambridge, 2003.
- [Tou08] ———, *Mathematical Logic*, John Wiley & Sons, Hoboken, NJ, 2008.