Mathematical Logic

Part I - 2014-2015

Roberta Prendin, 819575
robertaprendin@gmail.com

Introduction

In Mathematics sentences – theorems, propositions or lemmas – express properties of mathematical objects (integers, reals, matrices, functions, etc); these sentences are expressed in the natural language, which is redundant at best and vague at worst. Formal languages and natural languages are in fact worlds apart: while natural languages are the product of their time and thus extremely dynamic, formal languages are **less prone to modifications** and their interpretation is not up to discussion. A formal language is also **less redundant,** as information is represented using compact strings with a simpler structure than that of a normal language: for example, the meaning of the word "this" depends on the context in which it is used (technically speaking, "this" has a deictic value that is completely absent in any formal language). That does not mean that mathematical languages didn't change in the past, as technical books from the 15th or 16th century demonstrate; still, archaic formal languages are understandable to this day: we must thus recognize that formal languages are not only convenient but also powerful[1]. It comes as no surprise then that during the last century the focus was indeed on *formal languages*, which can be of **two kinds**:

a. *Formal languages* interpreting mathematics and its lemmas, axioms and propositions as **strings with an associated truth value**;
b. *Programming languages* aiming to provide an output given a singular input (or a certain set of inputs) and a possible interaction with the environment.

Given that even mathematical sentences can be expressed in the natural language, there's the risk of misunderstanding their meaning: *Mathematical Logic* is thus the subfield of mathematics exploring the **applications of formal logic to mathematics**; it interprets mathematical sentences by using formal languages and it studies the way information (lemmas, propositions, theories, theorems and so on) is represented. This way, sentences and proofs are formalized and **become mathematical objects just like integers or functions**. In this course we'll also analyze the **semantic relationship** between the strings produced by formal languages and any mathematical object of interest (from natural numbers to complex theorems), focusing on the reasons why the semantic of a certain formula (for example a theorem with a truth value) and its formal representation are related. Before going further, though, a *caveat*: given a certain formula as an input we can't possibly construct a function or a program capable of returning the

[1] The power of formal languages, however, is not absolute as Gödel stipulated in 1931 with the *Incompleteness theory*, which can be explained by using the following example. Suppose we want to know everything about the Universe: unfortunately this is an impossible task, for we are part of what we want to study. The same applies to formal languages: given a certain formal language with a defined syntax, using that language we can't tell if that language is consistent (if a certain string is part of the language or not). A higher, more powerful language is needed: we never know if our systems are consistent by staying *inside* those systems.

truth value of the initial formula: as we'll see in the following pages, there's a huge difference between the truth value of a formula and what we're capable of computing.

1 - The Language of Mathematics

Classical logic is composed by:

- A **formal language** to express properties and sentences. It's composed by **constants**, **operations** (unary, binary, ternary, etc) and **relations**.
- **Semantics** to give meanings to the strings of the formal language (of course we have more than one formula to express the same truth; however, the *semantic* of those formulas are the same).
- **Formal deduction** to prove sentences using logical rules.

Let's focus on the inner structure of the formal language first, which is in fact a simplified version of the natural language.

Terms and expressions. The atomic structures of any formal sentence are ***terms* or *expression***, which **must be considered mathematical objects.** Terms are symbols may or may not involve **evaluation or computing**. For example these terms denote mathematical objects:

- 3 denotes (=is a name for) the natural number three;
- III also denotes the natural number three, in the roman representation.

Semantically, 3 and III represent the same information "the natural number three" despite being two different symbols: this means that **the concept is not the symbol** and thus a different symbol might be preferred depending on the situation. What we can do with certain information depends on the way the information is represented (for example, 3 is useful to write algorithms, III might be more suitable in another context entirely). Other examples are:

- $5 + 6$ is a three-character string which denotes the natural number eleven. Notice that $5 + 6$ and 11 are the same mathematical object, but $5 + 6$ gives us more information: the natural number eleven can be computed by adding 5 to 6. This means that the same mathematical object can be described with different names, and every name provides a certain quantity of information, with some names more informational than others.
- 0.3147 denotes the real number $3/10 + 1/100 + 4/1000 + 7/10000$. The first expression denotes a real number which can be obtained by computing a series of sums and divisions.
- $\{3,6,7\}$ denotes the set whose elements are $\{3,6,7\}$.
- "$the\ author\ of\ Romeo\ and\ Juliet$" is a term denoting Shakespeare. Note that what we've said before applies even to the natural language: this string is an expression providing much more information than the simple name of the author.

Terms and expressions with variables. Mathematics also has the **ability to generalize**. We know firsthand that while generalizations may work in mathematics, they're often frowned upon in the everyday

life (think of the stereotypes we might be tempted to use every day). Nevertheless, generalizations are helpful to **express properties of a large set of items** (*all* the natural numbers, *all* the squares, etc):

- x denotes a generic element of the universe;
- $5 + x$, where x ranges over the set of natural numbers, denotes a generic natural number greater than 5.
- $0.3xy7$ denotes a generic real number of the form $3/10 + x/100 + y/1000 + 7/10000$.
- $\{x, y, z\}$ denotes a generic set of at most three elements.
- "$The\ author\ of\ x$" denotes a generic writer.
- $the\ natural\ number\ x$, where x represents a generic natural number ranging from 0 to $+\infty$.

While sentences such as "$the\ author\ of\ Romeo\ and\ Juliet$" have clear, perfect meanings not depending on the context, a variable x always has a range of possible meanings: therefore, we need to **specify the set or the world in which the variable takes its values**. By combining different terms and expressions we can construct atomic sentences (without variables) and atomic formulas (with variables).

$$Atomic\ sentences \rightarrow NO\ variables.$$

$$Atomic\ \boldsymbol{formulas} \rightarrow \boldsymbol{YES}\ variables.$$

Atomic sentences. An **atomic sentence** always denotes a truth value – meaning that **we can always say if an atomic sentence is true or false**. Examples of atomic sentences are:

- "$3\ divides\ 21$" is true;
- "$3 = 5 + 6$" is false);
- "121 is a multiple of 11" is true;
- "$11\ +\ 6$ is a prime number"
- "5 is odd" is true;
- "$3\ \in \{3, 6, 7\}$" is true;
- "$\{3\} \subseteq \{3, 6, 7\}$" is true;
- $John\ loves\ Mary$; might be true or false depending on the universe.
- $John\ is\ father\ of\ Mary$; might be true or false depending on the universe.
- $P(5)$; might be true or false depending on the universe and on the meaning of P.
- $Q(dog;\ cat;\ dogcat)$; might be true or false depending on the universe and the meaning of Q.

The words "$divides$", "$is\ a\ multiple\ of$", "$=$", "$\subseteq$", "$loves$" and "$is\ a\ father\ of$" are **binary relations** (or predicates with arity 2) as they relate pairs of elements of the universe we are speaking about. The words "$is\ odd$", "$is\ a\ prime\ number$" and "P" are instead **unary predicates** or properties of the elements of the universe. The ternary predicate "Q" relates triple of elements of the universe.

Atomic formulas. Atomic formulas are exactly like atomic sentences, but their truth values depends on the interpretation of the variables: the scope of the variable (the universe in which the variable is used)

must always be analyzed before making assumptions. For example, we might be tempted to say that "$x + y = y + x$" is true when, in fact, it may not work in a different universe where the sum is defined as:

+	***a***	***b***
a	b	a
b	b	a

This way, "$a + b = b + a$" is equals to "$a = b$", which is not true. Other examples of atomic formulas are:

- $x\ divides\ 122$; its truth value depends on the meaning of x.
- $x\ is\ a\ prime\ number$; its truth value depends on the meaning of x.
- $x\ is\ odd$; its truth value depends on the meaning of x.
- $3 \in X$; its truth value depends on the meaning of X.
- $John\ loves\ X$; its truth value depends on the meaning of X.
- $P(x)$; its truth value depends on the meaning of x.
- $Q(dog, x, y)$; its truth value depends on the meaning of x, y.

Propositional formulas. Propositional formulas are obtained by combining atomic sentences and propositional connectives:

$$and\ (\wedge),\ or\ (\vee),\ not\ (\neg),\ if - then\ (\rightarrow),\ if\ and\ only\ if\ (\leftrightarrow)$$

Propositional connectives create **relations between sentences**: for example "$2\ divides\ 122 \wedge 3 + 5 = 5 + 3$" is the connection between two atomic sentences, "$2\ divides\ 122$" and "$3 + 5 = 5 + 3$". Note that if A, B are formulas, $A \wedge B$, $A \vee B$, $\neg A$, $A \rightarrow B$ are formulas.

The semantic of propositional formulas is given by the truth tables of the connectives used. The general rules are:

- $A \wedge B$ is true iff A is true and B is true;
- $A \vee B$ is false iff A is false and B is false and therefore it's true if at least one sentence is true (either A or B; it could be both, of course);
- $\neg A$ is true iff A is false ($\neg$ negates the truth value of the sentence);
- $A \rightarrow B$ is false iff A is true and B is false. Note that any **implication with a false premise is true by default.** For example, the propositional formula "$if\ 5\ is\ odd, then\ 3\ is\ odd$" is true because the premise is true and the consequence is true. Note that the notational priority of the implication is:

$$A \rightarrow B \rightarrow C \rightarrow D \quad A \rightarrow \big(B \rightarrow (C \rightarrow)\big)$$

- $A \leftrightarrow B$ is **true** iff A, B are **both true or both false**.

To know the truth value of any propositional formula we need to know the truth values of its propositional variables; there are, however, propositional formulas that are **true for every possible interpretation** of the propositional variables: among these **tautologies** we count $A \rightarrow A$.

First-order formulas: *quantifiers*. More complex formulas can be obtained by using atomic formulas and **universal or existential quantifiers**:

$$\forall\ (For\ all) \quad \exists\ (There\ exists)$$

Their role is to specify the **scope of the variable**[2]: for example, given the set of natural numbers N, $\forall x \in N$ selects every possible element of the set N while $\exists x \in N$ considers a subset, depending on the property specified. Note that if A is a formula, $\forall xA$ and $\exists xA$ are also formulas. Some examples of sentences with universal or existential quantifiers in the natural language are:

- $Every\ man\ loves\ some\ woman.$
- $Some\ odd\ natural\ number\ divides\ 122.$
- $John\ likes\ every\ woman.$
- $Every\ triangle\ admits\ an\ acute\ angle.$
- $Every\ triangle\ admits\ an\ acute\ angle.$

Whenever we use either $\forall$ or $\exists$ in a sentence we must specify the universe in which they're used: for example, to understand the truth value of *"$John\ loves\ every\ woman$"* we must specify the universe in which "every woman" and "John" have a meaning. The sentences we've seen previously seen can also be translated in the formal language of mathematics:

- *"$Every\ man\ loves\ some\ woman$"*:

$$\forall x(Man(x) \rightarrow \exists y\big(Woman(y) \wedge loves(x,y)\big)$$

- *"$Every\ triangle\ admits\ an\ acute\ angle$"* can be translated as

$$\forall x\big(triangle(x) \rightarrow\ \exists y(acuteangle(y) \wedge\ y\ angleof\ (x)\big)$$

- *"$Every\ man\ loves\ some\ woman$"* becomes:

$$\forall x(man(x) \rightarrow \exists y(woman(y) \wedge x\ loves\ y)$$

- *"$Every\ even\ number\ is\ odd$"*:

$$\forall x\big(even(x) \rightarrow odd(x)\big)$$

- "$Some\ odd\ natural\ number\ divides\ 122$":

$$\exists x\big(odd(x) \wedge divides(x, 122)\big)$$

The **semantics of the quantifiers** is generally more complex than what've seen so far. given a model $\mathcal{M}$ with a universe $M = \{a_1, a_2, \dots, a_n\}$ and a general propriety $P(-)$, we have:

- $\forall xP(x) \leftrightarrow \big(P(\mathrm{a}_0)\ \wedge\ P(\mathrm{a}_1)\ \wedge \dots \wedge P(a_n)\big)$

  This means that that $\forall xP(x)$ is true in our universe if and only $P(a_1) \wedge P(a_2) \wedge \dots \wedge P(a_n)$ is true.

- $\exists xP(x) \leftrightarrow \big(P(\mathrm{a}_1)\ \vee\ P(\mathrm{a}_2)\ \vee \dots \vee P(a_n)\big)$

  This means that $\exists xP(x)$ is true in our universe if and only if $P(a_1) \vee P(a_2)\ \vee \dots \vee P(a_n)$ is true.

[2] Note that the quantifiers apply **only to variables** and **not to functions or properties** – and this is, in fact, the main feature of the *first-order logic*.

When ∀ is concerned we must prove that the property $P(-)$ is true for every object of the universe M:

$M = \{0,1,2,3,4, \dots\}$

$even(0) \rightarrow odd(0)$

$even(1) \rightarrow odd(1)$

…

If the universe M is infinite, we can't check if $\forall x P(x)$ is true in a finite time: we should check that each formula in the infinite sequence $P(a_1), \dots, P(a_n)$ is true:

$even(0) \rightarrow odd(0)$ premise: true; conclusion: false; implication: false.

$even(1) \rightarrow odd(1)$ premise: false; conclusion: true; implication: false.

…

Similarly, the check that $\exists x P(x)$ is false cannot be done in a finite time, because we have to check that each formula in the infinite sequence $P(a_1), \dots, P(a_n)$ is false. This is the reason why we need *proofs*: a proof of $\forall x P(x)$ gives us, in a finite time, the information whether $\forall x P(x)$ is true or false. Of course we need to have a starting point - *axioms*, for example: they're sentences which truthfulness is proved beyond doubt. If the universe is finite, instead, were able of proving sentences using **purely semantic methods** (meaning we do not need a proof).

Models and functions. Before giving a judgment of truth to a sentence we must **fix the model $\mathcal{M}$** which the sentence belongs to. The model $\mathcal{M}$ is a given by a **set M of elements** (called the universe of the model) among which there's a number of operational symbols $(f^M, g^M \dots)$ and relations $(P^m, Q^m, \dots)$. Operations and relations are therefore **subsets of the universe M**: for example, a unary predicate $P(-)$ is interpreted as a subset $P^m \subseteq M$, a binary predicate $R(-,-)$as a subset $R^m \subseteq M \times M$, etc. These functions and operations are **applied to the variables of the model**: for example in the natural language we can apply these functions…

$The\ father\ of(-)$,

$eats(-,-)$

…to the variables $John,\ Mary, Vegetables, x, y, z$… obtaining sentences such as "$The\ father\ of\ John$", "$Mary\ eats\ vegetable$", "$eats\left(father\ of\ (John),\ Father\ of(Mary)\right)$"…. If these sentences belong to the model $\mathcal{M}$ then they're true, otherwise they're false (technically speaking, if a sentence belongs to the subset P^m, then $P^m(sentence)$ is true, otherwise is false). For every function we must always specify the **arity** or number of arguments: "$The\ father\ of(-)$" has arity 1 while "$eats(-,-)$" has arity 2, as it's a binary relation. In general, if I have the function f with arity k:

$f^m: M^k \rightarrow M$

…it means that f takes k elements in input. To sum up, the universe M of a model $\mathcal{M}$ is a **set of constants (terms or expressions), operations and relations**. Every language can be organized as a model with these three components:

- $\forall\ constant\ c, c^m \in M$
- $\forall\ \boldsymbol{operation}\ symbol\ f, f^M \in M$
- $\forall\ \boldsymbol{relation}\ R, R^m \in M$

Formalization: from natural language to formal language. Every mathematical sentence we express in the natural language can also be formally translated, using the quantifiers, connectives and variables we've seen before. A general rule for translating quantifiers correctly is the following:

- "***every***" is generally translated with the **universal quantifier** $\forall$ **and an implication** $\rightarrow$, to reduce the scope of the sentence. For example "*Every professor is…*", "*Every multiple of 3 is…*" and, in general, "*Every P is…*" are translated formally as follows:

$$\forall x(P(x) \rightarrow \cdots)$$

- "***there exists***" is instead translated with the **existential quantifier** $\exists$ **and the propositional connective** $\wedge\ (and)$. For example "Some professor is…", "there is a multiple of 3 such that…" and, in general, "Some P is…" are translated as follows:

$$\exists x(P(x) \wedge \dots)$$

Also, remember that we can switch between quantifiers using these equivalences:

- $\neg\forall x P(x) = \exists x \neg P(x)$
- $\neg\exists x P(x) = \forall x \neg P(x)$

Several other useful rules are:

- $A \rightarrow B\ = \neg B \rightarrow \neg A$
- $\neg A \vee \neg B = \neg(A \wedge B)$
- $A \rightarrow B = \neg A \vee B$

2 - Proof in mathematics

Proof theory is a branch of Mathematical Logic concerned with the proving of formulas: instead of interpreting logical formulas in their model, they're actively proved using the rules for propositional connectives and quantifiers. As we'll see in the following pages, most of these rules work in *Intuitionistic logic* which is in fact **the only logical basis of Computer Science** and thus of programming.

Intuitionistic logic. Sometimes called *Constructive logic*, intuitionistic logic differs from classical logic by replacing the traditional concept of truth with the concept of ***constructive probability***; while in classical logic formulas are always assigned a truth value regardless of whether we have direct evidence for either case, formulas in intuitionistic logic are not assigned any definite truth value, and instead are considered **true only when we have a proof of them**. Otherwise, any formula has an unknown truth value until it's either proved or disproved. This means that any operation in intuitionistic logic needs **justification**. Formally, *Intuitionistic logic* has two main properties:

- **Disjunction property**: we have a proof for $A \wedge B$ if and only of we have of either A or B.

- **Witness property**: $\vdash \exists x\ A(x)$ if and only if $\vdash A(t)$ for some witness t. It means that if we prove the existence of an element t satisfying the property A, then we have a witness t for this proof of existence. Note that $\vdash$ A indicates that there's a formal proof for A.

Semantically, intuitionistic logic is a restriction of Classical Logic, as the ***law of excluded middle*** and ***double negation elimination*** are **not admitted**:

- **Law of excluded middle:** classical logic believes that the Boolean logic of truth values ("not true = false", "not false = true") also applies to proofs ("not provable = contradictory", "not contradictory = provable"), and therefore for any sentence A there's a proof of either A or $\neg A$; *intuitionistic logic* refutes this, stating that there might be sentences nor provable nor contradictory. Also, because of this, a non-contradictory formula is not automatically true in *Intuitionistic logic*: there's some "middle ground" that Classical logic fails to notice.
- ***Double negation elimination***: Classical logic believe that $\neg\neg A = A$. This is not true in intuitionistic logic, where the truth value of $\neg\neg A$ might also be undefined. We'll see some examples of this when studying *RRA* and *Peirce Law*.

Moreover, it's not not surprising that *Classical logic* is completely symmetric whereas *Proof theory* is not, as **we can only prove *true* sentences** using logical rules of deducibility.

Natural deduction

Suppose we have a certain formula A; in *Proof theory* "A" means "$there\ is\ a\ proof\ for\ A$", while "$\neg A$" means "$A\ is\ contradictory$". We can use the symbol $\vdash$ to suggest that we have a proof for a certain sentence: $A_1, \dots A_k \vdash$ B means "*we have a proof for* B *given the assumptions* $A_1, \dots, A_k$". $\vdash$ must not be confused with $|=$: $A_1, \dots, A_k \mid = B$ means "*the formula* B *is a logical consequence of the formula* $A_1, \dots, A_k$" or "*the formula* B *is true in every model of the formula* $A_1, \dots, A_k$".

Tree notation. Before introducing the rules of deduction of propositional logic, let's focus on the way these rules are presented. Imagine we have proofs for A and B: we can construct other proofs using them, for example the proof for $A \wedge B$:

$A, B \vdash$ A $\wedge$ B;

The same rule can be written using the **tree notation**:

$$\frac{A \quad B}{A \wedge B}[\wedge_i]$$

The meaning is the same: "*if we have a proof of the formula A and a proof of the formula B, then we have a proof of the formula A and B*". The upper part of the tree notation contains the what's already been proved, the lower part the consequences of the previous assumptions. We can list several rules:

- ***AND (IL)*:**

$$\frac{A \wedge B}{A}[\wedge_e] \quad (\wedge\ elimination) \qquad \frac{A \wedge B}{B}[\wedge_e] \quad (\wedge\ elimination)$$

$$\frac{A \quad B}{A \wedge B}[\wedge_i] \quad (\wedge \, introduction)$$

- ***OR (IL):***

$$\frac{A}{A \vee B}[\vee_i] \quad (\vee \, introduction) \qquad \frac{B}{A \vee B}[\vee_i] \quad (\vee \, introduction)$$

Notice that ∨ **can't be split in two parts**: given $A \vee B$ we can obtain only A or only B

$$\frac{\begin{matrix} A \vee B \end{matrix} \quad \begin{matrix} [A] \\ \dots \\ C \end{matrix} \quad \begin{matrix} [B] \\ \dots \\ C \end{matrix}}{C}[\vee_e] \qquad (\vee \, elimination)$$

If we have a proof for "A or B" and "with the assumption of A we have C" and "with the assumption of B we have C" then C is proved and ∨ is eliminated. Note that A and B are **discharged** by this operation.

- ***Implication or arrow*** $(\rightarrow)$***:***

$$\frac{A \rightarrow B \quad A}{B}[\rightarrow_e] \quad (\rightarrow \, elimination)$$

Arrow introduction: "*if we have a proof for A and a proof that $if\ A\ then\ B$, then we have a proof for B*"; it's also called ***modus ponens*** .

$$\frac{\begin{matrix} [A] \\ \dots \\ B \end{matrix}}{A \rightarrow B}[\rightarrow_i] \quad (\rightarrow \, introduction)$$

Arrow elimination: if by assuming A we obtain B, then we have a proof for $A \rightarrow B$. Note that A can be discharged.

- ***Contradiction or bottom*** $(\perp)$***.***

$$\frac{\perp}{A}[\perp_e]$$

The contradiction (or "false" or "absurd") is very useful: if we have a proof of the absurd, then **we can prove anything** we like and eliminate the contradiction. This rule will be very useful in λ-calculus.

- ***Elimination or not*** $(\neg)$***.*** The theoretical interpretation of the formula $\neg A$ is "I have a proof of $\neg A$ iff from the assumption A we can prove the absurd $\perp$ ". Therefore, not-elimination is:

$$\frac{A \quad \neg A}{\perp}[\neg_e]$$

If we have proofs of A and $\neg A$, then we have a contradiction. The not-introduction is instead:

$$\frac{\begin{matrix} [A] \\ \dots \\ \perp \end{matrix}}{\neg A}[\neg_i]$$

If, by assuming A, we reach a contradiction, then we've proved $\neg A$ and the assumption A is discharged.

- ***RRA (reduction to absurd, CL).*** This rule must not be confused with the very similar rule of not-introduction:

$$\begin{array}{l}[\neg A]\\ \dots\\ \dfrac{\perp}{A}\ [RRA]\end{array}$$

RRA marks the **distinction between classical logic, which this rule is part of, and intuitionistic logic**. This is easy to see if we consider these two properties of this rule:

- **Property #1**. RRA says that if by assuming $\neg A$ we get a contradiction $\perp$ then A is proved. Since the Intuitionistic logic does not automatically assume that "not contradictory = false", it's clear that RRA belongs to classical logic only.
- **Property #2**. In *Classical Logic*, given that by assuming $\neg A$ we get a contradiction $\perp$, it's automatically assumed that a witness of A exists. In intuitionistic logic, instead, *if* we prove the existence of an element *t* satisfying the property A, *then* it only means we have a witness t for this proof of existence.

$$\begin{array}{l}[\neg \exists x\ A(x)]\\ \dots\\ \dfrac{\perp}{\exists x\ A(x)}\ [RRA, classical\ logic\ approach]\end{array}$$

Question: proofs without $\boldsymbol{RRA}$. What happens when we are in this situation…

$$\begin{array}{l}[\neg A]\\ \dots\\ \dfrac{\perp}{?}\ [?]\end{array}$$

…and can't use RRA? We must use the not-introduction rule instead, thus obtaining a double negation:

$$\begin{array}{l}[\neg A]\\ \dots\\ \dfrac{\perp}{\neg(\neg A)}\ [\neg_i]\end{array}$$

Note that $\boldsymbol{RRA}$ **and** $\perp_{\mathbf{e}}$ **are equivalent**: we can prove $\neg\neg A$ from A and A from $\neg\neg A$:

$$\frac{\neg\neg A \qquad [\neg A]}{\dfrac{\perp}{A}[RRA]}[\neg_e] \qquad\qquad \begin{array}{l}[\neg A]\\ \dots\\ \perp\end{array}\over{\dfrac{\neg\neg A}{A}[\perp_{\mathrm{e}}]}[\neg_i]$$

- `Double negation (CL).` This rule is very intuitive and w**orks only in *Classical logic***.

$$\frac{\neg\neg A}{A}[\perp_{\mathrm{e}}]\ (double\ negation)$$

- `Peirce law (CL).` *Peirce Law* is defined in **classical logic** alone[3]:

$$\big((A \to B) \to A\big) \to A \qquad\qquad \begin{array}{l}[A \to B]\\ \dots\\ A\end{array}\over{A}[Peirce]$$

[3] Actually, with little modifications, also be defined in intuitionistic logic, if we remember that $A = \neg\neg A = (A \to\perp) \to\perp$. The rule thus becomes $\big((A \to B) \to A\big) \to \neg\neg A$, given that $\neg\neg A = (A \to\perp) \to\perp$.

Peirce Law **is always true** no matter the truth value of A and B or the model we've chosen, as we can see from this truth table:

A	B	$A \to B$	$(A \to B) \to A$	$\big((A \to B) \to A\big) \to A$
0	0	1	0	1
0	1	1	0	1
1	0	0	1	1
1	1	1	1	1

Peirce Law becomes particularly useful when $B \equiv \bot$; given that we're in the domain of *Classical logic*, then: $(A \to \bot) = \neg A$ and the law becomes:

$$\big((A \to \bot) \to A\big) \to A \qquad \to \qquad (\neg A \to A) \to A$$

The proof of this version of the law is:

$$\dfrac{\dfrac{\dfrac{[\neg A \to A] \qquad [\neg A]}{A}[\to_e] \qquad [\neg A]}{\bot}[\neg_e]}{\dfrac{A}{(\neg A \to A) \to A}[\to_i]}[RRA]$$

In fact:

- $(\neg A \to A) \to A$: this is an arrow-introduction, so we get A and assume $(\neg A \to A)$.
- A can be proved using RRA: we get $\bot$ and assume $\neg A$.
- $\bot$ can be proved using A and $\neg A$, with not-elimination. $\neg A$ is a previous assumption and can be discharged.
- A can be proved using arrow-elimination with $[\neg A]$, which is another assumption we can discharge, and $(\neg A \to A)$, which is again another dischargeable assumption.

In general, Peirce Law can be proved by using *RRA* and $\bot$. In fact:

- A: this is a RRA, so we get $\bot$ and assume A.
- $\bot$ can be proved using arrow-elimination between $\neg(A \to B)$ and $A \to B$: we get $\bot$ and assume $\neg A$.
- $\neg(A \to B)$ is a not-introduction: we get $\bot$ and assume $A \to B$.
- $\bot$ can be obtained through arrow-elimination. $A \to \bot$ is a previous assumption and can be discharged, A is proved by the assumption $A \to B$, which is also discharged.
- $A \to B$ is an arrow-introduction: we get B and assume A.
- B can be proved using $\bot$, which can also be proved using previous assumptions A and $A \to \bot$.

$$\dfrac{\dfrac{\dfrac{[A \to B]^{*}}{\vdots}}{\dfrac{A \qquad \dfrac{[A \to \bot]^{***}}{\bot}[\to_e]}{\neg(A \to B)}[\to_i;^{*}]} \qquad \dfrac{\dfrac{[A]^{**} \qquad [A \to \bot]^{***}}{\bot}[\to_e]}{\dfrac{\bot}{B}[\bot]}}{\dfrac{\bot}{A \to B}[\to_i;^{**}]}[\to_e]}{\dfrac{\bot}{A}[RRA;^{***}]}$$

The inverse is also possible, of course: given ⊥ and Pierce, we have:

$$\dfrac{\dfrac{\dfrac{[A \to \bot]^{*}}{\vdots}}{\bot}[\bot]}{\dfrac{A}{A}[\text{Peirce};^{*}]}$$

In fact:

- Using Peirce Law, we can prove A and assume $A \to \bot$.
- ⊥ can be proved by everything, including $A \to \bot$.

Note: *`call-cc`*. We'll see in future lessons that there's a strong similarity *Types* and the rules of *Natural Deduction*. Right now it's enough to say that it's been observed that the **Peirce Law is in fact the type of a construct called `call-cc`**, introduced in functions of programming languages years before these kind of studies in theoretical computer science and mathematical logic.

`Minimal logic: implication only`. If we further restrict the ìintuitionistic logic domains to the **sole implication**, we obtain the ***minimal intuitionistic logic***. This restriction is not surprising given that implication is one of the most important logic connectives, as it's related to the construction of functions: in fact $f: A \to B$ means that the function *f* implies a mapping between the type A and the type B. Consider now this following formula:

$$\big(A \to (B \to C)\big) \to (A \to B) \to A \to C$$

We can prove it just by using *minimal intuitionistic logic*, using *arrow introduction* and *arrow elimination*. Note that the **implication can simulate the ∧ connective**:

A	B	C	$A \to (B \to C)$	$(A \wedge B) \to C$
0	0	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	1	1

1	1	0	0	0
1	1	1	1	1

The result is:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[A \to B] \quad [A\,]}{B}[\to_e] \qquad \dfrac{[A \to (B \to C)] \quad [A]}{B \to C}[\to_e]}{C}[\to_e]}{A \to C}[\to_i]}{(A \to B) \to A \to C}[\to_i]}{\big(A \to (B \to C)\big) \to ((A \to B) \to A \to C)}[\to_i]$$

…where arrow-introduction and arrow-elimination are the only rules we've used. The implication can also **simulate ¬ introduction and ¬ elimination**:

$$\dfrac{\begin{array}{c}[A] \\ \dots \\ \bot \end{array}}{\neg A}[\neg_i] \qquad \dfrac{[A] \quad [\neg A]}{\bot}[\neg_e]$$

Given that $\neg A$ is logically equivalent to $A \to \bot$, we have:

$$\dfrac{[A] \quad [\neg A]}{\bot}[\neg_e] \quad becomes \quad \dfrac{[A] \quad [A \to \bot]}{\bot}[\to_e]$$

… and therefore, using the same equivalence, we have:

$$\dfrac{\begin{array}{c}[A] \\ \dots \\ \bot \end{array}}{\neg A}[\neg_i] \quad becomes \quad \dfrac{\begin{array}{c}[A] \\ \dots \\ \bot \end{array}}{A \to \bot}[\to_i]$$

Deduction rules for quantifiers.

Quantifiers can be proved using deduction rules. Remember the difference between **bound and free variables**: if we consider this example…

$$\forall x: \quad x + x = y$$

…x is bounded, as it's under the scope of a quantifier; y is free, it's not under the scope of a quantifier. We'll find these notions in the following pages: for now it's enough to note that:

$$\dfrac{P(x)}{\forall x P(x)}[\forall_i] \text{ means } \quad x\ is\ not\ free\ in\ \text{the hypotheses of the proof of } P(x)$$

$$\dfrac{\exists x P(x) \quad C}{C}[\exists_i] \text{ means } \quad x\ is\ not\ free\ in\ \text{the hypotheses of the proof of } P(x)$$

There exists intruduction, $\exists_i$. An existential statement such as $\exists x P(x)$ can be proved in two different ways:

- ***Provide a witness (or constructive method).*** To prove that there exists an element x satisfying the property P it's sufficient to **provide a witness**, a, for which the property holds. For example, to prove the formula "*there exists a prime number*" we just need to say that 3 is indeed a prime number. In tree notation:

$$\dfrac{P(a)}{\exists x\, P(x)}[\exists_i] \qquad \to \qquad \dfrac{\boldsymbol{P(3)}}{\exists x\, P(x)}[\exists_i]$$

- ***By contradiction***. This method consists of assuming that there exists a proof of $\neg\exists x\, P(x)$; then, we reason to get a contradiction, which proves that the initial assumption - $\neg\exists x\, P(x)$ - was indeed true[4].

Example #1. Suppose we have want to prove this theorem:

There exist two irrational numbers a and b such that a^b is rational[5]

To prove it we either use the constructive method and provide two witnesses – two irrational numbers such that their power is rational - or we try by the way of contradiction. Let's assume that the theorem is false; it means that $\sqrt{2}^{\sqrt{2}}$ is an irrational number. Of course $\sqrt{2}^{\sqrt{2}^{\sqrt{2}}}$ must be another irrational number, but if we compute it we discover that $\sqrt{2}^{\sqrt{2}*\sqrt{2}} = \sqrt{2}^2 = \sqrt{2} * \sqrt{2} = 2$, which is not an irrational number. We've found a contradiction and therefore the theorem is true.

For all introduction, $\forall_i$. In order to show a universal statement $\forall x P(x)$ either we give a **schema of proof** (starting as follows: "Let x be a natural number, then…. " and using general properties of the variable x), or we assume that there is no proof of $\forall x P(x)$ and we get a contradiction[6].

$$\frac{P(x)}{\forall x P(x)}[\forall_i] \quad x\ not\ free\ in\ hypotheses\ of\ P(x)$$

Note that we must assume that x **is not free in the hypotheses** of the proof of $P(x)$. To understand this, consider the following example.

Example #2 (*wrong application*). Suppose we want to prove this sentence:

$$\forall x[Prime(x) \rightarrow \forall x Prime(x)]$$

Proof. This formula is intuitively false, because we know there are natural numbers which are not prime, but we could be tempted to prove it anyway using this (wrong) proof:

$$\dfrac{\dfrac{\dfrac{[x \text{ is a prime number}]^*}{\forall x(x \text{ is a prime number})}[\forall_i; (\text{wrong application})]}{(x \text{ is a prime number}) \rightarrow \forall x(x \text{ is a prime number})}[\rightarrow_i;^*]}{\forall x[(x \text{ is a prime number}) \rightarrow \forall x(x \text{ is a prime number})]}[\forall_i; (\text{right application})]$$

---

[4] Note that classical logic it holds the law of excluded middle: for any sentence A, either A is provable or $\neg A$ is provable. Then in classical logic we can conclude that there exists a proof for $\exists \mathrm{x}\, \mathrm{P}(\mathrm{x})$, without providing a witness satisfying property P. In intuitionistic logic the fact that there's no proof of $\neg\exists \mathrm{x}\, \mathrm{P}(\mathrm{x})$ does not imply that there exists a proof for $\exists x\, P(x)$.

[5] Remember that rational numbers are $\frac{x}{y}$ where x and y are integers; irrational numbers can't be written as ratios: for example π, $\sqrt{3}$, etc.

[6] Again, this second way of reasoning only holds in classical logic.

We have two application of $\forall_i$ but only one is right: in the first case x is not free (… $\rightarrow \forall x\ (x\ is\ a\ prime\ number)$), while in the second x is free ("$x\ is\ a\ prime\ number$"). We can notice the absurd if, by assuming this demonstration is true, we choose $x = 3$…

$$Prime(3) \rightarrow \forall x\ Prime(x)$$

…and state that the assumption "$3\ is\ a\ prime\ number$" is true. This implies the conclusion $\forall x Prime(x)$ is also true: every natural number is prime! However, $Prime(3)$ is true but $\forall x\ Prime(x)$ is false: true implies false, and thus the sentence is false. Given that there exists an element such that the sentence is false, the initial sentence is false and can't be proved.

Example (*right application*). Suppose we want to prove the following statement:

$$\forall x(x^2 + 7x + 12\ is\ even)$$

Let x be a natural number. We consider two assumptions about x:

- x is even: if x is even, then we can write $x = 2k$, and therefore:

  $x^2 + 7x + 12 = 2(k^2 + 7x + 6)$, which is an even number.

- x is odd: if x is odd, then we can write it as $x = 2k + 1$ and therefore:

  $x^2 + 7x + 12 = (2k + 1)^2 + 7(2k + 1) + 12 = 4k^2 + 1 + 4k + 14k + 7 + 12 = 4k^2 + 18k + 20 = 2(2k^2 + 9k + 10)$, which is another even number.

We've thus proved correctly that the sentence is true.

For all elimination, $\forall_e$. This rule is rather basic: given a universal statement $\forall x P(x)$, then this statement is also true for a particular element a:

$$\frac{\forall x P(x)}{P(a)}\ [\forall_e]$$

Example. Let's get back to the previous general assumption:

$$\forall x(x^2 + 7x + 12\ is\ even)$$

So we get:

$$\frac{\forall x(x^2 + 7x + 12\ is\ even)}{3^2 + (7 * 3) + 12\ is\ even}\ [\forall_e]$$

42 is an even number and therefore we've correctly used $\forall_e$.

Example. In order to show a universal statement $\neg\forall x P(x)$ either we give a counter-example (an element a such that $P(a)$ is false) or we assume that there exists a proof of $\forall x P(x)$ and we get a contradiction; in this second case, we conclude that there is no proof of $\forall x P(x)$, which is a proof for $\neg\forall x P(x)$. For example, the statement $\forall n(n^2 + 3n + 4\ is\ a\ perfect\ square)$ is false because we have a counter-example, $n = 1$. Formally:

$$\dfrac{\dfrac{\dfrac{[\forall n(n^2+3n+4 \ is \ a \ perfect \ square)]^*}{1^2+(3\times 1)+4 \ is \ a \ perfect \ square}[\forall_e]}{8 \ is \ a \ perfect \ square}[evaluation] \quad \dfrac{}{8 \ is \ not \ a \ perfect \ square}[by \ arithmetics]}{\dfrac{\bot}{\neg\forall n(n^2+3n+4 \ is \ a \ perfect \ square)}[\neg_i{}^*]}[\neg_e]$$

In fact:

- $\neg\forall n(n^2+3n+4 \ is \ a \ perfect \ square)$ is a not-introduction: we get $\bot$ and assume $\forall n(n^2+3n+4 \ is \ a \ perfect \ square)$.
- $\bot$ can be obtained by two contradictorial sentences: for $x=1$, we have "$8 \ is \ a \ perfect \ square$" and "$8 \ is \ not \ a \ perfect \ square$".
- "$8 \ is \ not \ a \ perfect \ square$": in arithmetic we know that a number is a perfect square if we can decompose it in prime numbers and each prime number has an even exponent. Now $8 = 2^3$ is not a perfect square as the exponent is odd: we evaluate "$8 = 1^2+(3*1)+4 \ is \ a \ perfect \ square$" and discharge $\forall n(n^2+3n+4 \ is \ a \ perfect \ square)$.

Note that from the **point of view of *Computer Science***, $\exists x P(x)$ can be proved, sometimes, by showing a *witness*: if I am able to find an element satisfying P, then $\exists x P(x)$ is at least semidecidable; note that $\forall\, xP(x)$ is neither semidecidable nor decidable.

There exists elimination, $\exists_e$. This rule is similar to ∨-elimination, at least in structure:

$$\dfrac{\exists x P(x) \qquad \begin{matrix}[P(x)]\\ \dots \\ C\end{matrix}}{C}[\exists_e]$$

Note that $\boldsymbol{x}$ **is not free** in hypotheses of the proof of $\exists x P(x)$ and C. This rule is often use to show $\neg\exists x P(x)$:

$$\dfrac{\exists x P(x) \qquad \begin{matrix}[P(x)]\\ \dots \\ \bot\end{matrix}}{\dfrac{\bot}{\neg\exists P(x)}[\neg_i]}[\exists_e]$$

Not-introduction is also useful to prove $\neg\ \forall\, xP(x)$:

$$\dfrac{\begin{matrix}[P(x)]\\ \dots \\ \bot\end{matrix}}{\neg\ \forall\, xP(x)}[\neg_i]$$

Example. We want to prove $\neg\exists x\left(P(x)\wedge\neg P(x)\right)$. Note that we can already say this is a false formula, as there can't possibly be an element satisfying both $P(x)$ and $\neg P(x)$ at the same time. Also note that x is not free, as it's bound by the quantifier ∃. The solution is:

$$\dfrac{\dfrac{[\exists x\,(P(x)\wedge\neg P(x))] \qquad \dfrac{\dfrac{[P(x)\wedge\neg P(x)]}{c}[\wedge_e] \qquad \dfrac{[P(x)\wedge\neg P(x)]}{\neg P(x)}[\wedge_e]}{\bot}[\neg_e]}{\bot}}{\neg\exists x\,(P(x)\wedge\neg P(x))}[\neg_i]$$

In fact:

- $\neg\exists x\,(P(x)\wedge P(x))$: this is a not-introduction, and therefore we have $\bot$ and assume $\exists x\,(P(x)\wedge\neg P(x))$.
- As usual, $\bot$ can be proved using different methods: given that we have $\exists x\,(P(x)\wedge \text{neg}\,P(x))$ among our assumptions, we can use exists-elimination: we get $\bot$ and assume $P(x)\wedge\neg P(x)$. $\exists x\,(P(x)\wedge\neg P(x))$ is discharged.
- Again, $\bot$ can be proved using different methods: given that we have $P(x)\wedge\neg P(x)$, we use $P(x)$ and $\neg P(x)$, with a not-elimination.
- We can prove both $P(x)$ and $\neg P(x)$ using $P(x)\wedge\neg P(x)$, which is thus discharged.

3 - Axiomatic Systems

We do not prove only tautologies, that is, sentences which are true thanks to their logical structure; we're also interested in proving **sentences from sets of axioms** which are assumed "true" (axioms of Euclidean geometry, the axioms of set theory ,etc). In this section we provide a certain number of *axiomatic systems* but before doing this, let's review three very important theorems.

- ***Completeness theorem for first-order logic* (1939)**: given a set of axioms – Ax -, we can have a formal prove of the formula ϕ if and only the formula ϕ is true in every model $\mathcal{M}$.

$$Ax\ \vdash \phi \quad iff \quad \forall\, model\ \mathcal{M} of\ Ax,\ \mathcal{M}\ |= \phi$$

This is a very strong relation between *formal proofs* and *semantics*. To understand its meaning, let's introduce an example of axiomatic system. Imagine we have a language with two binary predicates, $\leq$ and $-$, a constant 0 and the following axioms:

$\forall x(x\leq x)$ ***reflexive property***

$\forall x,y,x(x\leq y\wedge y\leq z\rightarrow x\leq z)$ ***transitive property***

$\forall x,y(x\leq y\wedge y\leq x\rightarrow x=y)$ ***antisymmetric property***

$\forall(0\leq x)$

This is a theory of *partially ordered sets with a minimum element* - a set of sentences provable using the axioms we've mentioned above. Suppose now we have this sentence:

$$\forall x,y(x\leq y\vee y\leq x)$$

Can we prove it? The answer is **no**: we cannot prove this sentence using the *axioms of partial ordering* because this sentence formalizes a *total* order and not a *partial* one. If I find a model where the sentence is false, then by the *completeness theorem* we cannot prove this sentence from the axioms. If the sentence is true in *every* model, then we know for certain that a formal proof of it must exist. Remember that this

theorem is **true only in first order logic**, in which ***predicates do not have other predicates or functions as arguments*** (variable range over the atomic elements of the model, and not properties with inner structures like functions or properties). Both **arithmetics** and **geometry** are **not within the boundaries** of first order of logic (arithmetic, for example, because it uses the induction principle which uses a quantifier over the property P:

$$\forall P(P(0) \wedge \forall\big(P(x) \rightarrow P(x+1)\big) \rightarrow \forall x\, P(x)$$

- ***Compactness theorem for first order logic*** . A set of sentences T is consistent – meaning we **cannot derive the contradiction** from it and **has models** – if and only if every finite subset of T is consistent.
- ***Completeness theorem of First-order logic***. According to Godel's ***Completeness theorem of First-order logic*** (1939), *semantics and syntax of first-order logic are equivalent*. It means that given a set of axioms Ax, a sentence F is proved if and only if F is true in every model $\mathcal{M}$ of the axioms Ax
  - $\vdash F \quad iff \quad F\ is\ true\ in \forall\ model\ of\ the\ axiom$

It means that, to prove F is false, it's enough to find a model of the axioms Ax where F is false. It's also interesting that both **arithmetics** and **geometry** are **not within the boundaries** of first-order of logic (arithmetic, for example, because it uses the *full induction principle*).

-

Let's now consider examples of first-order and second-order logic.

First-order Peano Arithmetics. Also known simply as Peano Arithmetics, this first-order logic can be formalized as follows:

- $\boldsymbol{Constants}$: 0,1,2,3, …
- $\boldsymbol{Operations}$: $(-) = (-)$
- $\boldsymbol{Binary\ operations}$: $+,\ *$

Peano Arithmetics is composed by all sentences in the language of arithmetics provable from the following set of axioms:

- $(\boldsymbol{PM_1})$ $\boldsymbol{Succession\ operation}$ $(+1)$: *the function successor is injective and 0 is no the successor of any other number:*

  $$\forall y(0 \neq\ y+1);\ \forall x, y(x+1 = y+1 \rightarrow x = y)$$

  Also, the successor operation is injective:

  $$\forall x, y(x+1 = y+1 \rightarrow x = y)$$

- $(\boldsymbol{PM_2})+$ $\boldsymbol{operation}$. *The function "+" is defined by induction:*

  $$\forall x(x+0 = x), \qquad \forall x, y(x+(y+1) = (x+y)+1)$$

- $(\boldsymbol{PM_3})*$ $\boldsymbol{operation}$. The function " $*$" is defined by induction:

  $$\forall x(x*0 = 0),\ \ \forall x, y(x*(y+1) = (x*y)+x)$$

- ($\boldsymbol{RI}, restricted\ Induction\ principle$). Given that this is a first-order logic, the Induction principle is thus restricted:

$$\phi(0) \wedge \forall x(\forall x\big(\phi(x) \rightarrow \phi(x+1)\big) \rightarrow \forall x \phi(x)$$

Note that this is just an *approximation* of the full induction principle and thus we lose the ability to range over different languages. A language is therefore fixed and any other property is expressed only in terms of that particular language.

Second-order Peano Arithmetics. The second-order Peano Arithmetics is defined exactly as the first-order Peano Arithmetic, with a small difference: the *restricted* induction principle is substituted with the *full* induction principle; every other axiom still stands.

- $(\boldsymbol{FI})$ ***Full induction principle***:

$$\forall\, P\ \Big(P(0) \wedge \forall x\big(P(x) \rightarrow P(x+1)\big) \rightarrow \forall x P(x)\Big)$$

Note that the full induction principle is not part of the first order logic, as the quantifier is over properties or functions. With the induction principle we can **define properties and functions.** Note that if we have Second-order Peano Arithmetics, then **there exists only one model up to isomorfism for its axioms**, the language of arithmetics: of course we can have different models using the Induction principle but, in the end, they're just the same Arithmetic model with different names.

Paris-Harrington Theorem (1977). Paris and Harrington have found an interesting theorem of Arithmetics, which is true in the usual model of natural numbers N but it's not provable in Peano Arithmetics. Suppose we have a model $\mathcal{M}$ of Peano Arithmetics: we can define the following function using the induction principle:

$$f: N \rightarrow m \quad (f\ is\ injective\ and\ preserve\ operations)$$
$$f(0) = 0^m$$
$$f(n+1) = f(n) +^m 1$$

The function f is **injective** and **preserves operations**, meaning that $f(x+y) = f(x) +^m f(y)$ and $f(x * y) = f(x) *^m f(y)$. Now, we would like to prove that f is **surjective**: if it is, then, it means that the model of natural numbers and the model $\mathcal{M}$ are the same: $\mathcal{M}$ would be just another name for the set of natural numbers N. Now, the surjective property can be expressed as:

$$\forall z \in \mathcal{M}, \exists y \in N\ (f(y) = z)$$

…meaning that for every element z of the codomain $\mathcal{M}$ there exists a corresponding element y in the domain N such that $f(y) = z$. But look at this property:

$$P(z) \equiv\ \exists y \in N\ (f(y) = z)$$

z is a variable ranging over $\mathcal{M}, \mathcal{M}$ is a model of Peano Arithmetic so we would like to apply the induction principle; unfortunately, the induction principle can be used only for properties which can be expressed in terms of Arithmetics – and $P(z)$ can't be expressed using the properties of Arithmetics! In fact, assume P

can be expressed in the language of Arithmetics: is it true that there exists a natural number such that it's mapped to 0^m? No. In fact, assume another constant, C such that:

$C > 0$

$C > 1$

…

$C > n$

We thus obtain an **infinite set of axioms** that we can add to the model of Arithmetics: this way, the first-order Peano Arithmetics plus the infinite set of axioms provided by C becomes a **consistent set**. Now, if we recall the *compactness theorem of first order logic*, we know that a set is consistent if and only if every finite subset of it is consistent. Every finite subset is thus consistent, and then the compactness theorem tells me the full set is consistent too. But in the set of natural numbers we do not have an infinite element greater than any other – hence P can't be expressed in the language of Arithmetics.

4 – Relation between types & proofs.

So far, we've analized formulas and properties of natural deduction and quantifiers; however, there's a certain **similarity between these rules and types**, which are studied in *Type theory*. In Type theory, every term has a type, and operations are restricted to terms of a certain type. Of course, to work with different elements we need their types to be compatible: this behaviour is mirrored in mathematics, where functions with complex domains and codomains can work only on compatible variables. For example, consider the function $Double(-)$, which takes in input another function and produces the composition of the function in input:

$$Double(f) = f \circ f$$

Given a certain function $f: A \to B$, $Double(f)$ is not possible since A and B are different sets (for example, a set of strings and a set of integers). $Double(f)$ must explicitly use these types to work correctly:

$$double\colon\ (A \to A) \to (A \to A)$$

…where A is a variable representing the **most general type** the function *double* can use.

Cartesian product: $\wedge_i$, $\wedge_e$. Recall the definitions of the Cartesian product × and the connective ∧:

- ***Cartesian product***: $A \times B = \{(a, b)\colon a \in \mathrm{A},\ b \in B\}$
- ***Logic connective*** ∧: $\frac{A \quad B}{A \wedge B}[\wedge_i]$

The Cartesian product – which produces couples of elements taken from different sets - can be expressed in terms of ∧:

$$\frac{a \in A \qquad b \in B}{(a, b) \in A \times B}$$

Given an element a of type A and an element b of type B, the Cartesian product produces a couple (a, b) of types $A \times B$. The same thing can be expressed using **type notation:**

$$\frac{P\colon A \quad Q\colon B}{(P, Q)\colon A \wedge B}$$

This means that if we have a proof P for A and a proof Q for B, then (P, Q) is a proof for $A \wedge B$. Consider now the definition for ∧-elimination:

$$\frac{A \wedge B}{A}[\wedge_e] \qquad \frac{A \wedge B}{B}[\wedge_e]$$

Both formulas can be rewritten introducing *proofs*:

$$\frac{P\colon A \wedge B}{\pi_1(P)\colon A}[\wedge_e] \qquad \frac{P\colon A \wedge B}{\pi_2(P)\colon B}[\wedge_e]$$

…where $\pi_1(P)$ is the **first variation of proof P** and $\pi_2(P)$ is the **second variation of the proof P**. Note that $\big(\pi_1(P), \pi_2(P)\big)$ is the original proof P.

Disjoint union, $\vee_i$. The disjoint union U̇ is related to the or-introduction rule:

- ∨ ***introduction***: $\frac{A}{A \vee B}[\vee_i] \qquad \frac{B}{A \vee B}[\vee_i]$
- ***Disjoint union*** U̇: $A \dot{\cup} B = \{(x, i)\colon \ either\ (x \in A\ and\ i = 0)\ or\ (x \in B\ and\ i = 1)\}$

$A \dot{\cup} B$, distinguishes whether the element chosen belongs to A or B using a label i: if the element is from the first set $i = 0$, otherwise $i = 1$. The same distinction is made when we want to prove, given A, the weaker assumpton $A \vee B$:

$$\frac{P\colon A}{(P, 0)\colon A \vee B}[\vee_i] \qquad \frac{P\colon B}{(P, 1)\colon A \vee B}[\vee_i]$$

This way we remember whether the proof from $A \vee B$ comes from the proof of A (label 0) or the proof of B (label 1).

Implication. Consider the natural deduction rule of arrow-introduction:

$$\frac{A \qquad A \to B}{B}[\to_i]$$

Like before, consider a proof X for A and a proof Y for $A \to B$, then the proof for B is $Y(X)$:

$$\frac{X\colon A \qquad Y\colon A \to B}{Y(X)\colon B}[\to_e]$$

Another example of the implication, using instead the $double(-)$ function, is:

$$\frac{f\colon A \to A \quad double(A \to A) \to (A \to A)}{double(f)\colon A \to A}[\to_e]$$

Note that the "simpler" function or element is always on the left, and "inside" the more complex function.

Contradiction and error type. Consider the proof for the contradiction, assuming $\neg(A \vee \neg A)$ and A:

$$\frac{\dfrac{[A]}{A \vee \neg A} \qquad (A \vee \neg A) \to \bot}{\bot}[\neg_e]$$

Assume we have proofs for A and $\neg(A \vee \neg A)$: we can use the proof in natural deduction exactly like before, assuming A is proved by X and $A \vee \neg A$ is proved by Y.

$$\frac{\dfrac{\boldsymbol{X}\colon A}{(\boldsymbol{X}, \boldsymbol{0})\colon A \vee \neg A} \qquad \boldsymbol{Y}\colon \big((A \vee \neg A) \to \bot\big)}{Y(< X, 0 >)\ \bot}$$

Note that $\bot$ can be intended as the **error type** – the type of functions mapping a certain domain to the codomain of contradiction $\bot$. Note that this rule has two variables: it means that any program implementing this procedure will use **only two variables** to prove the absurd.

4 - Curry-Howard corrispondence for Intuitionistic logic, λ-calculus

Curry and Howard have established the following correspondence:

$$p \text{ is a proof of } A \leftrightarrow p \text{ is a program of type } A$$

In other words, formulas and types are two different interpretations of the same thing[7]. For example, the implication becomes:

$$p \text{ is a proof of } A \to B \text{ means that } p \text{ is a function from } A \text{ into } B.$$

For example, consider this formula:

$$(A \to B) \to \big(A \to (B \to C)\big) \to A \to C$$

We want to find a program (or a function) that, given the following inputs…

A	How to find the inputs: read the formula **from right to left**, and see what's in front of the arrow.
$A \to (B \to C)$	
$A \to B$	

…produce the output C. Let's **associate a variable** to each input:

$x: A$
$y: A \to (B \to C)$
$z: A \to B$

Note that y and z can be also viewed ad functions (for example, z is a function from the domain A to the codomain B), so to get the output C we can:

- Apply z to x obtaining an output of type B: $zx: B$
- Apply y to x obtaining an output of type $B \to C$: $yx: B \to C$
- Apply yx to zb to get an output of type C: $(yx)(zx): C$

…and therefore the program $(yx)(zx)$ is a proof for C. We can represent these passages as follows:

$$\frac{\dfrac{x: A \quad z: A \to B}{zx: B}[\to_e] \qquad \dfrac{x: A \quad y: A \to (B \to C)}{yx: B \to C}[\to_e]}{(yx)(zx): C}[\to_e]$$

Suppose now that we want c: $\phi = (A \to B) \to \big(A \to (B \to C)\big) \to A \to C$. We must assume a **top-down approach**, starting from our assumptions, ad use the previous operations we've listed:

$$\frac{\dfrac{\dfrac{x: A \quad z: A \to B}{zx: B}[\to_e] \qquad \dfrac{x: A \quad y: A \to (B \to C)}{yx: B \to C}[\to_e]}{(yx)(zx): C}[\to_e]}{\lambda x.\,(yx)(zx): A \to C}$$

[7] Note that we use the notation "$p: A$" for "p is a proof for A" and "p is a program of type A".

The operator λx **constructs a function of x from the expression** $(yx)(zx)$, which has type C. To understand the meaning of λ, recall what's the meaning of a certain function f.

> *Suppose we have a rule or expression E which depends on the variable x. We can construct the function whose values are given by the expression E by introducing a new symbol, f, which is the name of the function*:
>
> $$f(x) = E$$
>
> We can do the same thing in lambda calculus, with a slightly different notation:
>
> $$\lambda x.\, E$$
>
> λ is the **variable-binding operator** of functional abstraction, with respect to the variable x. As an example, consider this arithmetical expression $x^2 + 1$. We can thus write $\lambda x.\,(x^2 + 1)$

Let's get back to our previous example: the function can be constructed as follows.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x\colon A \quad z\colon A \to B}{zx\colon B}[\to_e] \qquad \dfrac{x\colon A \quad y\colon A \to (B \to C)}{yx\colon B \to C}[\to_e]}{(yx)(zx)\colon C}[\to_e]}{\lambda x.\,(yx)(zx)\colon A \to C}[\to_i]}{\lambda y\lambda x.\,(yx)(zx)\colon (A \to (B \to C)) \to A \to C}[\to_i]}{\lambda z\lambda y\lambda x.\,(yx)(zx)\colon (A \to B) \to (A \to (B \to C)) \to A \to C}[\to_i]$$

…and therefore the λ-term $\lambda z\lambda y\lambda x.\,(yx)(zx)$is a proof of the formula ϕ.

Exercise #1 (Peirce Law). Recall the Peirce formula, $\big((A \to B) \to A\big) \to A$. We know that it's not provable in intuitionistic logic: this means that there's no way to construct an output of type A from a function $f\colon (A \to B) \to A$, because we do not have an input of type $A \to B$, just $\big((A \to B) \to A\big)$. We can, however, consider the intuitionistic version of Peirce Law and prove it:

$$\big((A \to B) \to A\big) \to \neg\neg A \;\; = \big((A \to B) \to A\big) \to (\mathrm{A} \to \bot) \to \bot$$

In fact, $\neg A \;\; = A \to \bot$, meaning that $\neg\, \neg A \;\; = (A \to \bot) \to \bot$. In this case, the inputs are:

$$y\colon A \to \bot$$
$$x\colon (A \to B) \to A$$

Let's also assume another input $z\colon A$, which **must** be eventually discharged. A function for the sentence $\phi = \big((A \to B) \to A\big) \to (\mathrm{A} \to \bot) \to \bot$ can be obtained as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[z\colon A] \quad y\colon A \to \bot}{\bot}[\to_e]}{\mathcal{A}(\mathrm{yz})\colon B}[\bot_{\mathrm{e}}]}{\lambda z.\,\mathcal{A}(\mathrm{yz})\colon A \to B}[\to_i] \qquad x\colon (A \to B) \to B}{x\big(\lambda z.\,\mathcal{A}(\mathrm{yz})\big)\colon A}[\to_e] \quad y\colon A \to \bot}{y\big(x\big(\lambda z.\,\mathcal{A}(\mathrm{yz})\big)\colon \bot}[\to_e]}{\lambda y.\,y\big(x\big(\lambda z.\,\mathcal{A}(\mathrm{yz})\big)\colon (A \to \bot) \to \bot}[\to_i]}{\lambda x\lambda y.\,y\big(x\big(\lambda z.\,\mathcal{A}(\mathrm{yz})\big)\colon \big((A \to B) \to A\big) \to (A \to \bot) \to \bot}[\to_i]$$

Note that:

- $yz: \bot$ it's the equivalent of an arrow-elimination $\frac{z:A \quad y:A\to\bot}{\bot}[\to_e]$ used to get $\bot$.
- $\mathcal{A}(\mathrm{yz}): B$ from $\bot$ I can prove everything, including B. $\boldsymbol{\mathcal{A}}$ **is an operator transforming the absurd in any type we want**, including B. It applies the bottom elimination rule.
- $\lambda z.\mathcal{A}(\mathrm{yz}): A \to B$, given that z is a variable of type A, if we consider it as a function it trasforms B in $A \to B$.
- $[z: A]$ is the only input discharged, as it's an assumption. Every assumption we add must be discharged.

Example #1 (formalization). Consider the arrow introduction rule, where $[A]$ is an assumption we can discharge to get a proof for $A \to B$:

$$\frac{\begin{array}{c}[A]\\ \dots \\ B\end{array}}{A \to B}[\to_i]$$

Suppose we want to define a program or a function for the arrow-introduction rule over inputs of type Nat (which stands for "Natural number"). The result is:

$$\frac{\begin{array}{c}[x: Nat]\\ \dots \\ (3+x, x): Nat\ x\ Nat\end{array}}{\lambda x.(3+x, x): Nat \to (Nat\ x\ Nat)}[\to_i]$$

This way, by writing $\lambda x.$ before the expression $(3+x, x)$, we define a function with types $Nat \to (Nat\ x\ Nat)$. Note that, in this way, $\boldsymbol{x}$ **is no more a free variable**, but it's bound to the "λx" (this is the same behavior of quantifiers).

Exercises and examples

Part 1 - 2014-2015

Roberta Prendin, 819575

robertaprendin@gmail.com

Exercises on natural deduction.

Exercise #1. We want to prove $A \wedge B, B \wedge A \rightarrow C \vdash C \vee D$. The idea is to write the proof as a **top-down tree** where the root is the formula we want to prove, $C \vee D$, and the leaves are the assumptions we'going to make. The solution is:

$$\dfrac{\dfrac{\dfrac{[A \wedge B]}{A}[\wedge_e] \qquad \dfrac{[A \wedge B]}{B}[\wedge_e]}{B \wedge A} \qquad [B \wedge A \rightarrow C]}{\dfrac{C}{C \vee D}[\vee_i]}[\rightarrow_e]$$

In fact:

- Step 1: $C \vee D$ this must be an *or-introduction* and thus we obtain C. Of course we could've obtained D, but C is our assumptions so it's wiser to use C.
- Step 2: C can be proved using he assumption $B \wedge A \rightarrow C$, $B \wedge A$ and arrow-elimination. $B \wedge A \rightarrow C$ is also one of the assumptions we're bound to use, so we can discharge it.
- Step 3: $B \wedge A$ is a and-introduction using B and A.
- Step 4: to prove B and A separately, we can use the given assumption $A \wedge B$ and and-elimination.

Exercise #2. We want to prove that the formulas $\neg A$ and $A \rightarrow \perp$ are logically equivalent: in other words we want to prove $\neg A$ using $A \rightarrow \perp$ and viceversa.

Part one: $A \rightarrow \perp$ by using $\neg A$.

$$\dfrac{\dfrac{\neg A \quad [A]}{\perp}[\neg_e]}{A \rightarrow \perp}[\rightarrow_i]$$

In fact:

- Step 1: : $A \rightarrow \perp$ is an arrow-introduction, and therefore A becomes an assumption we will need to use later.
- Step 2: to prove $\perp$ we have at least three possible solutions:
  - **Not-elimination.** Useful when all else fails or when we have simple proofs of two contradictorial sentences:

$$\dfrac{\neg A \quad \boldsymbol{A}}{\perp}[\neg_e]$$

  - **Arrow-introduction**. Useful when we have implications among our assumptions.

$$\dfrac{A \quad A \rightarrow \perp}{\perp}[\neg_e]$$

  - **Or-elimination**. Useful whe we have a $\vee$ among our assumptions.

$$\dfrac{\begin{matrix} & [A] & [B] \\ \mathrm{A \vee B} & \dots & \dots \\ & \perp & \perp \end{matrix}}{\perp}[\vee_e]$$

  In this case, we can use $\neg A$ and A, and therefore we have the solution.

Part two: $\neg A$ by using $A \rightarrow \perp$.

$$\dfrac{\dfrac{[A] \quad A \to \bot}{\bot}[\to_e]}{\neg A}[\neg_i]$$

In fact:

- Step 1: $\neg A$ is a not-introduction: we assume A and use $\bot$.
- Step 2: $\bot$ can be proved by using three different solutions. In this case we have $A \to \bot$ among our assumptions, end therefore we can use the arrow introduction rule to prove $\bot$.

Exercise #3. We want to prove $A \to \neg\neg A$. The result is:

$$\dfrac{\dfrac{\dfrac{A \quad \neg A}{\bot}[\neg_e]}{\neg(\neg A)}[\neg_i]}{A \to \neg\neg A}[\to_i]$$

In fact:

- Step 1: $A \to \neg\neg A$ is an implication, so this is clearly an arrow-introduction. We thus assume A and have $\neg\ \neg A$.
- Step 2: $\neg\ \neg A$ can be seen as $\neg(\neg A)$, and therefore this is an not-introduction. We assume $\neg A$ and have $\bot$.
- Step 3: $\bot$ can be obtained by using one of three solutions. In this case we have assumed both A and $\neg A$, so we can use a not-elimination.

Exercise #4. We want to prove DeMorgan's rule $(\neg A \wedge \neg B) \to \neg(A \vee B)$.

$$\dfrac{\dfrac{[A \vee B] \quad \dfrac{\dfrac{(\neg A \wedge \neg B)}{\neg A}[\wedge_e] \quad [A]}{\bot}[\neg_i] \quad \dfrac{\dfrac{(\neg A \wedge \neg B)}{\neg B}[\wedge_e] \quad [B]}{\bot}[\neg_i]}{\bot}[\vee_e]}{\neg(A \vee B)}[\neg_i]}{(\neg A \wedge \neg B) \to \neg(A \vee B)}[\to_i]$$

In fact:

- Step 1: $(\neg A \wedge \neg B) \to \neg(A \vee B)$ is an implication, so this is an arrow-introduction. We thus assume $\neg A \wedge \neg B$ and have $\neg(A \vee B)$.
- Step 2: $\neg(A \vee B)$ is a not-introduction. We assume $A \vee B$ and have $\bot$.
- Step 3: $\bot$ can be obtained by using one of three solutions. Given that we have $A \vee B$ among our assumptions, we can use the or-elimination rule: we assume A, B and obtain $A \vee B, \bot, \bot$:

$$\dfrac{\dfrac{[A \vee B] \quad \bot \quad \bot}{\bot}[\vee_e]}{\neg(A \vee B)}[\neg_i]}{(\neg A \wedge \neg B) \to \neg(A \vee B)}[\to_i]$$

- Step 4: we now need to prove both $\bot$s. Given that we have A, B we can use the not-elimination:

$$\dfrac{\dfrac{[A \vee B] \quad \dfrac{\neg A \quad [A]}{\bot}[\neg_i] \quad \dfrac{\neg B \quad [B]}{\bot}[\neg_i]}{\bot}}{\neg(A \vee B)}[\neg_i]}{(\neg A \wedge \neg B) \to \neg(A \vee B)}[\to_i]$$

- Step 5: we need to prove $\neg B$ and $\neg A$. We have $\neg A \wedge \neg B$ among our assumptions, so we can use a simple and-elimination:

$$\dfrac{\dfrac{[A \vee B] \quad \dfrac{\dfrac{(\neg A \wedge \neg B)}{\neg A}[\wedge_e] \quad [A]}{\bot}[\neg_i] \quad \dfrac{\dfrac{(\neg A \wedge \neg B)}{\neg B}[\wedge_e] \quad [B]}{\bot}[\neg_i]}{\bot}[\vee_e]}{\dfrac{\neg(A \vee B)}{(\neg A \wedge \neg B) \rightarrow \neg(A \vee B)}[\rightarrow_i]}[\neg_i]$$

Exercise #5. We want to prove $\neg\neg(A \vee \neg A)$ assuming A, $\neg A$ and $\neg(A \vee \neg A)$. The solution is:

$$\dfrac{\dfrac{[\neg(A \vee \neg A)] \quad \dfrac{A \quad \neg A}{A \vee \neg A}[\vee_i]}{\bot}[\neg_e]}{\neg\neg(A \vee \neg A)}[\neg_i]$$

In fact:

- Step 1: we have $\neg\neg(A \vee \neg A)$ which can be seen as $\neg\big((A \vee \neg A)\big)$. This is a not-introduction, so we have $\bot$ and assume $\neg(A \vee \neg A)$.
- Step2: $\bot$ can be proved using one of the usual three solutions. Given that we have $\neg(A \vee \neg A)$ among our assumptions, we use a simple not-elimination.
- Step 3: we now need to prove $A \vee \neg A$, which is an or-introduction using assumptions A and $\neg A$.

Exercise #6. Prove $\neg\neg A$ from A in *intuitionistic logic* only. The result is:

$$\dfrac{\dfrac{[\neg A] \quad A}{\bot}[\neg_e]}{\neg\neg A}[\neg_i]$$

In fact:

- Step 1. Given that cannot use RRA as it's a rule of Classical logic, the only similar rule we can use is not-introduction: we thus have $\bot$ and $\neg A$ is assumed.
- Step 2. $\bot$ can be obtained using three different solutions. Given that we have $\neg A$ and A among our assumptions, we use and-introduction.

Exercise #7. Prove A from $\neg\neg A$. The result is:

$$\dfrac{\dfrac{\neg\neg A \quad \neg A}{\bot}}{A}[RRA]$$

In fact:

- Step 1: A can be interpreted as the result of a RRA. We get $\bot$ and assume $\neg A$.
- Step 2: $\bot$ can be obtained using three different solutions. Given that we have $\neg\neg A$ and $\neg A$, then we use not-elimination.

Exercise #8. Prove that $A \rightarrow (B \rightarrow A)$. The solution is:

$$\dfrac{\dfrac{[A]}{B \rightarrow A}[\rightarrow_i]}{A \rightarrow (B \rightarrow A)}[\rightarrow_i]$$

In fact:

- Step 1: $A \rightarrow (B \rightarrow A)$ is an implication, and therefore we have an arrow-introduction. We thus assume A and have $B \rightarrow A$.
- Step 2: $B \rightarrow A$ is another implication, , and therefore we have another arrow-introduction. We thus assume B and have A.
- Step 3: A is an assumption, and therefore we do not need to prove it. We thus discharge A.

Note that given we do not have occurrences of B, B is just assumed and never discharged.

Exercise #9 (*not introduction*). Prove that $\neg(A \wedge \neg A)$

$$\dfrac{\left(\dfrac{A \wedge \neg A}{A}[\wedge_e] \qquad \dfrac{A \wedge \neg A}{\neg A}[\wedge_e]\right)}{\dfrac{\bot}{\neg(A \wedge \neg A)}}[\neg_{\mathrm{i}}]$$

In fact:

- Step 1. $\neg(A \wedge \neg A)$ is a not-introduction: we get $\bot$ and assume $A \wedge \neg A$.
- Step 2. $\bot$ can be proved in three different ways: given that we have $A \wedge \neg A$, we can use A and $\neg A$.
- Step 3. A and $\neg A$ can be proved using the assumption $A \wedge \neg A$ and a and-elimination. The assumption is discharged.

Exercise #10 (*not introduction and elimination*). Prove that $A \rightarrow \neg\neg A$. The result is:

$$\dfrac{\dfrac{\dfrac{[A] \quad \neg A}{\bot}[\neg_e]}{\neg\neg A}[\neg_i]}{A \rightarrow \neg\neg A}[\rightarrow_{\mathrm{i}}]$$

In fact:

- Step 1. $A \rightarrow \neg\neg A$ is an arrow-introduction: we get $\neg\neg A$ and assume A.
- Step 2. $\neg\neg A$ is a not-introduction: we get $\bot$ and assume $\neg A$.
- Step 3. As usual, $\bot$ can be proved using different techniques. Given that, among our assumptions, we have both A and $\neg A$, which are both discharged.

Exercise #11 (*RRA*). Prove that $\neg\neg A \rightarrow A$. The result is:

$$\dfrac{\dfrac{\dfrac{[\neg\neg A] \qquad [\neg A]}{\bot}[\neg_e]}{A}[RRA]}{\neg\neg A \rightarrow A}[\rightarrow_i]$$

In fact:

- Step 1. $\neg\neg A \rightarrow A$ is an arrow-introduction, so we get A and assume $\neg\neg A$.
- Step 2. A can be introduced using RRA: we get $\bot$ and assume $\neg$A.
- Step 3. $\bot$ can be introduced using our assumptions, $\neg\neg$A and $\neg$A, which are thus discharged.

Exercise #12 (*not introduction*). Prove that $A \wedge \neg$ B $\rightarrow$ $\neg(\neg A \vee B)$. The result is:

$$\dfrac{\dfrac{\dfrac{\neg \mathrm{A} \vee B \qquad \dfrac{\dfrac{[A \wedge \neg \mathrm{B}]}{A}[\vee_{\mathrm{e}}] \quad [\neg A]}{\bot}[\neg_{\mathrm{e}}] \qquad \dfrac{[B] \quad \dfrac{[A \wedge \neg \mathrm{B}]}{\neg B}[\vee_{\mathrm{e}}]}{\bot}[\neg_{\mathrm{e}}]}{\bot}[\vee_i]}{\neg(\neg A \vee B)}[\neg_i]}{A \wedge \neg B \rightarrow \neg(\neg A \vee B)}[\rightarrow_i]$$

In fact:

- Step 1. $A \wedge \neg B \rightarrow \neg(\neg A \vee B)$ is an arrow-introduction, so we get $\neg(\neg A \vee B)$ and assume $A \wedge \neg B$.
- Step 2. $\neg(\neg A \vee B)$ is a not-introduction, so we get $\bot$ and assume $\neg A \vee B$.
- Step 3. $\bot$ can be proved in different ways. Given that we have $\neg A \vee B$ among our assumptions, we can use or-elimination, assuming $\neg \mathrm{A}$ and B. Note that $\neg A \vee B$ is discharged.
- Step 4. A can be proved using $A \wedge \neg B$ and A with an or-elimination. The same applies with B, and thus both $A \wedge \neg \mathrm{B}$ are discharged .

Exercise #13. Prove this formula: $\neg(A \wedge B) \rightarrow \neg A \vee \neg B$. The result is:

$$\dfrac{\dfrac{\dfrac{[\neg(A \wedge B)] \qquad \dfrac{[A] \quad [B]}{A \wedge B}[\wedge_i]}{\bot}[\neg_e]}{\neg A}[\neg_i] \qquad \dfrac{\dfrac{[\neg(A \wedge B)] \qquad \dfrac{[A] \quad [B]}{A \wedge B}[\wedge_i]}{\bot}[\neg_e]}{\neg B}[\neg_i]}{\neg A \vee \neg B}[\vee_i]}{\neg(A \wedge B) \rightarrow \neg A \vee \neg B}[\rightarrow_i]$$

In fact:

- Step 1. $\neg(A \wedge B) \rightarrow \neg A \vee \neg B$ is an arrow-introduction, so we get $\neg A \vee \neg B$ and assume $\neg(A \wedge B)$.
- Step 1. $\neg A \vee \neg B$ can be seen as an or-introduction using $\neg A$ and $\neg B$.
- Step 2. Let's focus on $\neg A$: this is a not-introduction, so we get $\bot$ and assume A. The same can be applied to B, which is a not-introduction with $\bot$ and the assumption of B.
- Step 3. $\bot$ can be obtained in many different ways. Given that we have $\neg(A \wedge B)$ among our assumptions, a not-introduction. $\neg(A \wedge B)$ is also discharged.
- Step 4. $\mathrm{A} \wedge \mathrm{B}$ can be proved using our previous assumptions, A and B with an and-introduction. Both are thus discharged.

Exercise #14. Prove this formula: $A \vee A \leftrightarrow A \vee \bot$. The first step is to divide this formula into two parts:

1. $A \vee A \rightarrow A \vee \bot$
2. $A \vee \bot \rightarrow A \vee A$

The proof of *part 1* is:

$$\dfrac{\dfrac{[A \vee A] \qquad \dfrac{A}{A \vee \bot}[\vee_{\mathrm{i}}] \qquad \dfrac{A}{A \vee \bot}[\vee_i]}{A \vee \bot}[\vee_e]}{A \vee A \rightarrow A \vee \bot}[\rightarrow_i]$$

In fact:

- Step 1. $A \vee A \rightarrow A \vee \bot$ is an arrow-introduction, so we get $A \vee \bot$ and assume $A \vee A$.
- Step 2. $A \vee \bot$ could be an or-introduction, but let's consider it as the result of or-elimination: therefore we can use the assumption $A \vee A$, which is discharged:

$$\frac{A \vee B \qquad \begin{matrix}[A][B]\\ \dots \quad \dots \\ C \quad C\end{matrix}}{C}[\vee_e]$$

  In our case, $C = A \vee \bot$, and we can use this rule assuming A and A.
- Step 3. Now that we have the assumption A, we can easily prove $A \vee \bot$ with a simple or-introduction.

The proof for part 2 is similar:

$$\frac{\dfrac{A \vee \bot \qquad \dfrac{[A]}{A \vee A}[\vee_i] \qquad \dfrac{\dfrac{[\bot]}{A}[\bot_e]}{A \vee A}[\vee_i]}{A \vee A}[\vee_e]}{A \vee \ \bot \ \rightarrow A \vee A}[\rightarrow_i]$$

In fact:

- Step 1. $A \vee \bot \rightarrow \ A \vee A$ is an arrow-introduction, so we get $A \vee A$ and assume $A \vee \ \bot$.
- Step 2. $A \vee A$ can be proved using $A \vee \bot$ and an or-elimination, following the previous rule. We thus assume A and $\bot$.
- Step 3. $A \vee A$ can be proved exactly like before with an or-introduction. One A is thus discharged; the other must be obtained by $\bot$ with a bottom-elimination. $\bot$ is thus discharged. Note that **we can discharge just *one A***: for the other we must use the bottom elimination.

Exercise #15. We want to prove $(A \wedge B) \wedge C \ \vdash A \wedge (B \wedge C)$. It means that given $(A \wedge B) \wedge C$ we can prove $A \wedge (B \wedge C)$. The result is:

$$\frac{\dfrac{[(A \wedge B) \wedge C]}{\dfrac{A \wedge B}{A}[\wedge_e]}[\wedge_e] \qquad \dfrac{\dfrac{[(A \wedge B) \wedge C]}{\dfrac{A \wedge B}{B}[\wedge_e]}[\wedge_e] \qquad \dfrac{[(A \wedge B) \wedge C]}{C}[\wedge_e]}{B \wedge C}[\wedge_i]}{A \wedge (B \wedge C)}[\wedge_i]$$

In fact:

- Step 1. $A \wedge (B \wedge C)$ is an or-introduction obtained using A and $B \wedge C$.
- Step 2. A is the result of an and-elimination with $A \wedge B$, which is also the result of an and-elimination from $(A \wedge B) \wedge C$. $(A \wedge B) \wedge C$ is an assumption, so we can stop the proof of this part here.
- Step 3. $B \wedge C$ is the result of an or-introduction with B and C.

- Step 4. B is again the result of an or-elimination with $A \wedge B$, which is also the result of an and-elimination from $(A \wedge B) \wedge C$. $(A \wedge B) \wedge C$ is an assumption, so we can stop the proof of this part here.
- Step 5. C is the result of an or elimination with $(A \wedge B) \wedge C$, which is again an assumption. We can stop the proof of this part here.

Exercise #16. We want to prove the formula $\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$ using the *minimal intuitionistic logic*. The solution is:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[A \to (B \to C)] \quad [A]}{B \to C}[\to_e] \quad \dfrac{[A \to B] \quad [A]}{B}[\to_e]}{C}}{A \to C}[\to_i]}{(A \to B) \to (A \to C)}[\to_i]}{\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)}[\to_i]$$

In fact:

- Step 1. $\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)$ is an arrow introduction, so we get $\big((A \to B) \to (A \to C)\big)$ and assume $\big(A \to (B \to C)\big)$.
- Step 2. $(A \to B) \to (A \to C)$ is another arrow-introduction, so we get $(A \to C)$ and assume $(A \to B)$.
- Step 3. $(A \to C)$ is yet another arrow-introduction, so we get C and assume A.
- Step 4. C must be the result of an arrow-elimination. We can use $B \to C$ and B.
- Step 5. $B \to C$ is an arrow-introduction with $A \to (B \to C)$ and A, which ar both discharged;
- Step 6. B is an arrow-introduction with $A \to B$ and A, which are both discharged.

Exercise #17. Prove $A \to B \vdash \neg B \to \neg A$. The result is:

$$\dfrac{\dfrac{\dfrac{\dfrac{[A \to B] \quad [A]}{B}[\to_e] \quad [\neg \mathrm{B}]}{\bot}[\neg_e]}{\neg A}[\neg_i]}{\neg B \to \neg A}[\to_e]$$

In fact:

- Step 1. $\neg B \to \neg A$ is an arrow-introduction, so we get $\neg A$ and assume $\neg B$.
- Step 2. $\neg A$ is a not-introduction, and thus we get $\bot$ and assume A.
- Step 3. $\bot$ can be obtained in many different ways. Given that we have $\neg B$ among our assumptions, we can use not-elimination. $\neg B$ is thus discharged.
- Step 4. B is the result of an arrow-elimination between $A \to B$ and A. The former is an assumption we were given, the latter is an assumption we got from step 2, so the proof is complete.

Exercises on formalizations and formal languages

Exercise #0 (formalization of Arithmetics). Consider Arithmetics as a formal language.

Solution. We have:

- Constants: $0, 1, 2, 3,4,5$ … belonging to the set of natural numbers N.
- Operations: $+^N$ (sum), $*^N$ (product)
- Relations: $=^N$ (equality)

Note that other predicates such as "divides", "is multiple of" and "less or equal than" are not primitives of this model and thus need to be defined in terms of $+^N$ and $*^N$:

- "$x\ divides\ y$": $\exists z(y = x * z)$
- "$x\ is\ multiple\ of\ y$": $\exists z(y * z = x)$
- "$x\ is\ less\ than\ z$": $\exists z(x = y + z \wedge x \neq 0)$
- "$x\ is\ even$": $\exists k(x = 2 * k)$
- "$x\ is\ odd$": $\exists k(x = 2 * k + 1)$

For example, a formalized sentence of this language is:

- "$Every\ multiple\ of\ 6\ is\ multiple\ of\ 3$": $\forall x \exists k(x = 6k \rightarrow x = 3k)$

Exercise #1. Formalize the **Set Theory** as a formal language.

Solution. We have:

- **Constant**: $\emptyset \quad (empty\ set)$
- **Functions**: ∪ (arity 2), ∩ (arity 2), \ (difference, arity 2)
- **Relations:** ∈, $=$ (arity 2)

Again, note that the predicate ⊆ is not a primitive of the model and must be defined as follows, using the few instruments we have:

- $X \subseteq Y\ \forall x(x \in X \rightarrow x \in Y)$ meaning that every element of X also belongs to the set Y.

Other interesting sentences to formalize are:

- "$There\ exists\ the\ empty\ set$": $\exists X \forall x(\neg x \in X)$
- "$There\ exists\ at\ least\ one\ element$": $\exists x(x = x)$
- "$There\ exist\ at\ least\ two\ elements$": $\exists x, y\ (x \neq y)$
- "$There\ exist\ at\ least\ three\ elements$"": $\exists x, y, z(x \neq y \wedge x \neq z \wedge y \neq z)$
- "$There\ exists\ at\ most\ one\ element$": $\neg\exists x, y(x \neq y)$
- "$There\ exist\ at\ most\ two\ elements$": $\neg\exists x, y, z(x \neq y \wedge x \neq z \wedge y \neq z)$
- "$the\ universe\ is\ empty$": $\neg\exists x(x = x)$
- "$The\ universe\ has\ exactly\ one\ element$": $\exists x(x \in\ Universe\ \wedge\ \forall y(y \in Universe\ \rightarrow x = y)$

An example of formula is $\forall x\ \forall y(z(x \cup y) = (z\backslash x) \cap (z\backslash y))$ .Note that z is a free variable because it's **not in scope of a quantifier**; x and y are instead bound.

Exercise #2. Consider the following model:

- Constants: $Mary$
- Binary predicate: $likes(-,-)$
- Unary predicate: $Professor(-)$

Formalize these sentences:

1. *Mary likes every professor*: $\forall x\big(Professor(x) \rightarrow likes(Mary,x)\big)$.
2. *Mary likes some professor:* $\exists x\big(Professor(x) \wedge likes(Mary,x)\big)$
3. *Mary likes only the professors:* this sentence is equivalent to "Mary likes every professor and if someone is not a professor Mary does not like them". The result is:

$$\forall x\big(Professor(x) \rightarrow likes(Mary,x) \wedge \neg Professor(x) \rightarrow \neg likes(Mary,x)\big)$$

4. *Only one professor likes Mary:* this sentence means "only one professor likes Mary; if some other professor likes Mary then that professor must be the only one that likes Mary". The result is:

$$\exists x(Professor(x) \wedge likes(x,Mary) \wedge \forall y(Professor(y) \wedge likes(y,Mary) \rightarrow x = y)$$

5. *Two professors like Mary:*

$$\exists\, x,y(Professor(x) \wedge Professor(y) \wedge likes(x,Mary) \wedge likes(y,Mary) \wedge x \neq y.$$

Note that we must specify that these two professors are **two separate entities**.

6. *Only two professors like Mary*:

$$\exists x,y(Professor(x) \wedge Professor(y) \wedge likes(x,Mary) \wedge likes(y,Mary)$$
$$\wedge x \neq y \wedge \forall z(Professor(z) \wedge likes(z,Mary) \rightarrow z = x \vee z = y)$$

In this case we must specify that any other professor liking Mary is one of the other two linking Mary.

Exercise #3. Consider Ca' Foscari's database. We have:

- **Constants**: id numbers (843210, etc), names ($Antonio$, $Giovanni$), course ($Matematica\ Discreta$, $Mathematical\ Logic$, $Programming$etc).
- **Functions**: $teaches(-,-)$, $follows(-,-)$
- **Relations**: $matricola(\_,\_)$, $student(-)$, $professor(-)$, $administrative(-)$
- **Model**: $Professor(Salibra)$, $Student(Giovanni)$

Some formulas are:

- $Salibra\ teaches(Mathematical\ Logic\ )$ (true in our model)
- $Salibra\ teaches(Programming)$ (false)
- $Matricola\ (Giovanni, 842310)$ (true in our model)

In the language of mathematics, the sentence "*every student is a professor*" can be translated as "$for\ every\ x, if\ x\ is\ a\ student\ then\ it's\ a\ professor$":

$$\forall x\big(student(x) \rightarrow professor(x)\big)$$

Example #4 (*no-one, nothing*). Consider the following model:

- Constants: the set containing all the students of a certain University.

- Relations: $likes(-,-)$
- Functions: $Course(-), Student(-)$

Formalize the following sentences:

1. *No student likes every course*. To translate this sentence correctly we must negate "*there exists a student who likes every course*":

$$\neg\exists x(Student(x) \wedge \forall y\big(Course(x) \rightarrow likes(x,y)\big)$$

2. *No course is loved by all students*.

$$\neg\exists x(Course(x) \wedge \forall y\big(Student(y) \rightarrow likes(x,y)\big)$$

Exercise #5. How do we translate in a formal language the sentence "*There exists a set with exactly two elements*"?

Solution. Generally speaking, our sentence must have this structure:

$$\exists X\ \big(X \text{ has two and only two elements}\big)$$

If the set X has two and only two elements, then it means that given a and b:

- Both a and b belong to the set X;
- a is different than b (otherwise X would have *at least* one element, not exactly two);
- If a third element z belongs to X, then the element z must be equal to either a or b:

$$\exists X\ (\exists a,\ b(a \in X \wedge\ b \in X \wedge\ \neg(a = b) \wedge \forall z(z \in X \rightarrow (z = a \vee z = b))$$

Exercise #6. Suppose we want to analyze **Geometry** through mathematical Logic. We have:

- **Constants**: point, line, triangle, etc.
- **Functions**: unary predicates (to be a point $P(-)$, to be a line, $L(-)$), binary predicates ($=$, $\in$, I), ternary predicates $B(-,-,-)$ (a point is between two other points).

An example of formula is "$two\ points\ belong\ exactly\ to\ a\ line$" which can be translated as follows:

$$\forall x\forall y(P(x) \wedge P(y) \wedge x\ \neq y \rightarrow \exists!\, z(L(z) \wedge x \in z \wedge y \in z)$$

Example #7 (*for all*). Translate the following sentence: "*every even number is odd*".

Solution. "*Every even number is odd*" becomes "*for every natural number x, if x is even then it's also odd*" and can be translated as $\forall x\big(even(x) \rightarrow odd(x)\big)$.

Example #8 (*for all*). Consider the sentence *"Every odd number higher than 5"*. How can we translate it?

Solution. We can transform the sentence in "*for every natural number, if it's odd then it's greater than 5*". The result is $\forall x(odd(x) \rightarrow x \geq 5)$.

Example #9. Let's check the truth value of the previous sentence. Given that x can be $0, 1, 2, \ldots$ we have:

- $(odd(0) \rightarrow 0 \geq 5)$ false premise, true by default.
- $(odd(1) \rightarrow 1 \geq 5)$ true premise, false conclusion; implication: false.
- …

…which means that the sentence is false.

Example #10 ($there\ exists$). Let's consider the sentence "*there is an even number which is odd*". How can we translate it?

Solution. We must use the ∃ quantifier and the connective ∧: $\exists x\big(\ even(x) \wedge odd(x)\big)$.

Example #11 ($at\ least,\ at\ most$). Sentences with "at least two elements", "at least one element", "at most one element" can be difficult to translate, but we can define a certain path to make the job easier for us. Consider the translations of these sentences, for example:

1. There exists at least one element: $\exists x(x = x)$

2. There exist at least two elements: $\exists x, y(x \neq y)$

3. There exist at least three elements: $\backslash\exists x, y, z(x \neq z \wedge z \neq y \wedge x \neq y)$

Note that "there exists at least one element" is different than "there exists *only* one element, as we've seen before. Consider now these sentences:

.4 There exists at most one element: this is equivalent to the **negation** *of there exist at least two element*: $\neg\exists x, y(x \neq y)$. It can also be translated as $\forall x, y(x = y)$.

Example #12. Let's consider this example:

- The universe is the set $A = \{dog, cat, house\}$;
- $=$ is the symbol for equality, a binary relation where $dog = dog$ is true, $dog = cat$ is false, etc;
- $P(-)$ is a unary relation where $P(dog)\ =\ true, P(cat) = false, P(home) = true$.

Let's consider the variable x which scope is the entire set A; a complex formula is, for example:

$$P(dog) \rightarrow \exists\, y(dog = y)$$

What is the meaning of this formula? Is it true or false in the universe A? A is a finite and quite limited set of elements, so we can consider every possible substitution to answer these questions:

$P(dog) \rightarrow \exists\, \boldsymbol{dog}(dog = dog)$: the premise is true, the conclusion is true, the implication is true.

$P(cat) \rightarrow \exists y(dog = cat)$: the premise is false, the conclusion is true by default.

$P(home) \rightarrow \exists y(\mathrm{dog} = home)$: the premise is true, the conclusion is true, the implication is true.

Let's call these three instances of the same sentence a, b and c respectively: $a \wedge b \wedge c$ is true and therefore the sentence is true. Of course, given an infinite set of elements, we wouldn't do a similar analysis. We can also simplify the sentence:

$$P(dog) \rightarrow (dog = dog\ \vee\ dog = cat\ \ \vee\ dog = \mathrm{home})$$

…or make other implications:

$$[P(dog) \rightarrow (dog = dog\ \vee\ dog = cat\ \ \vee dog = \mathrm{home})]\ \wedge$$
$$[P(cat) \rightarrow (cat = dog\ \vee cat = cat\ \ \vee cat = \mathrm{home})] \wedge$$
$$[P(\mathrm{home}) \rightarrow (\mathrm{home} = dog\ \vee \mathrm{home} = cat\ \ \vee \mathrm{home} = \mathrm{home})]$$

Example #13. Let's consider the sentence "*Mary eats every vegetable*". Let's specify:

- **Universe**: the universe of all people and all possible kinds of food.
- **Constant**: $Mary$ is a string which denotes an element of the universe.

- **Binary relations**: $-eats-$ [8];
- **Unary relation:** $vegetable(-)$.

To make the translation easier, let's transform the sentence in "*Every vegetable is eaten by Mary":*

$$\forall x(vegetable(x) \rightarrow \ Mary\ eats\ x)$$

Note: there's no need to define Mary ("$there\ exists\ Mary\ and\ Mary\ eats\ x$ …") because "$Mary$" is a constant term defined in the chosen universe.

Example #14. Let's consider the truth value of the previous sentence "*Mary eats every vegetable"*:

- $Vegetable(John) \rightarrow Mary\ eats\ John$: the premise il false therefore the sentence is true by default.
- $Vegetable(Bob) \rightarrow Mary\ eats\ Bob$: the premise il false therefore the sentence is true by default.
- $Vegetable(Meat) \rightarrow Mary\ eats\ Meat$: the premise il false therefore the sentence is true by default.
- $Vegetable(Tomato) \rightarrow Mary\ eats\ Tomato$: both the premise and the conclusion are true, hence the implication is true.

In the universe where Mary likes tomatoes the sentence is true. However, we have at least one false implication: it follows that the sentence is false.

Example #15. Let's consider these symbols:

$$0,1,2,+,*,=$$

How can we classify them?

- 0 is a constant, and thus a term; the same applies to 1, 2, 3, … which are symbols for the natural number 0, 1, 2, etc.
- $+$ is symbol for a **binary operation** (and not a binary relation, as 5+6 would be a truth value when it's in fact a number) and interprets the addition operation. The same applies to *, which interprets the multiply operation;
- $=$ is a binary relation: $5 = 6$ has a truth value, *false*.

We could decide that these symbols are interpreted by the set of natural numbers N:

$N\ = \{0,1,2,\dots\}$

"0" $is\ interpreted\ by\ 0$

"1" $is\ interpreted\ by\ 1$

…

I could however interpret this language with another notation, for example:

$N\ = \{0,1\}$

…meaning that the universe is interpreted by bits only $(0\ /\ 1)$: the constant "0" is interpreted by 0, the constant "1" is interpreted by 1, the constant 2 is interpreted by 1 (it's a decision on our part: we could say

[8] Note the notation $-eats-$ to show the number of arguments of the verb "eat".

that even numbers are interpreted by 1, odd numbers by 0). The binary operation $+$ could be interpreted by this table:

+	**0**	**1**
0	*0*	*1*
1	*1*	*0*

Example #16. Let's consider this sentence, given the model of the previous example (where 5 is 1):

$$\forall x(x = 5 \quad \rightarrow \quad 5 + x = x + x)$$

Is this true? Let's see:

- $0 = 1 \rightarrow 1 + 0 = 0 + 1$: the premise is false, the implication is true by default;
- $1 = 1 \rightarrow 1 + 1 = 1 + 1$: the premise is true, the conclusion is true, the implication is true.
- …

…which means that the sentence is actually true.

Example #17. Let's consider this formula, written in the natural language:

The father of John drives the car.

Let's analyze this formula:

- "*The father of John*" and "*car*" are expressions denoting a person and an object respectively;
- "*drives*" is a verb, a relation between objects.

Example #18. Let's consider this complex formula:

Every natural number greater than 4 is equal to the addition of three prime numbers.

In the formal language, the formula is:

$$\forall x\big(x > 4 \rightarrow \exists a, b, c\ (\ prime(a) \wedge\ prime(b) \wedge\ prime(c) \wedge\ x = (a + b) + c)\big)$$

Note that we don't have symbols such as $>$ or the property $prime(-)$ in our language, so we must define them:

- $x > y \quad iff \quad \exists z\left(x = y + z\ \wedge\ \neg(z = 0)\right)$
- $\forall z(z\ divides\ x \rightarrow (z = 1\ \vee\ z = x))$

Example #19. How can we represent the empty set $\emptyset$?

$$\exists X\ \forall z\ \neg(z \in X)$$

Exercises in class.

Consider the universe as the set of natural numbers N. We have the following language:

symbol	***meaning***
a	The number 3 (constant)
b	The number 10 (constant)
W	"to be a prime" (unary predicate)
E	"to be a square" (unary predicate)
L	"greater than" (binary predicate)

Starting from these symbols, we can obtain *expressions* and *formulas* - for example:

x is an expression

a is an expression

b is an expression

$a + b$ is not an expression, as we do not have $+$ in our languages.

Expressions are just constants and variables, nothing more. Examples of formulas are:

$W(a)$ is a formula. It means "$the\ number\ 3\ is\ a\ prime\ number$". It's true in our universe.

$E(a)$ is a formula. It means "$the\ number\ 3\ is\ a\ square$". It's false in our universe.

By using logical connectives, we can build more complex formulas - for example:

$W(a) \wedge E(a)$. It's a formula and it means

"$the\ number\ 3\ is\ prime\ and\ the\ number\ 3\ is\ a\ square$", which is false in our Universe.

Exercise #1. Using the language we've seen previously, translate the following formula:

Every square is greater than some prime number

Let's focus on quantifiers we have *every* and *some*. The first is equal to $\forall$, the other is $\exists$. Note that a binary relation can be written as $L(x, y)$ or as xLy, which is more intuitive. A possible translation is:

$$\forall x\big(E(x) \rightarrow \exists y(W(y) \wedge xLy)\big)$$

Question. Is it possible to translate the sentence with $\forall x \exists y(E(x) \wedge W(y) \rightarrow xLy)$? No, because these formalizations are not equivalent. We can prove this by analizing these two cases:

x	
0	$\forall x\big(E(0) \rightarrow \exists y(W(y) \wedge 0Ly)\big)$ $\boldsymbol{false}$

1	$\forall x\big(E(1) \to \exists y(W(y) \wedge 1Ly)\big)$ $\boldsymbol{false}$
2	$\forall x\big(E(2) \to \exists y(W(y) \wedge 2Ly)\big)$ $true$
3	
…	
n	

x	
0	$\forall x \exists y(E(x) \wedge W(y) \to xLy)$ $\boldsymbol{true}$
1	$\forall x \exists y(E(x) \wedge W(y) \to xLy)$ $\boldsymbol{true}$
2	$\forall x \exists y(E(x) \wedge W(y) \to xLy)$ $true$
3	…
…	
n	

To put the existential quantifier at the beginning of the sentence is $\forall x \exists y(E(x) \to W(y) \wedge xLy)$.

Exercise #2. Formalize the following sentence: "every natural number greater than 3 is the square of a prime number, which is less than 10".

$$\forall x\big(xLa \to \exists y(W(y) \wedge E(x) \wedge x = y * y \ \wedge bLy)\big)$$

Note that to say "square of a number" we've introduced $*$ and =, otherwise the sentence can't be formalized:

$\boldsymbol{symbol}$	$\boldsymbol{meaning}$
a	The number 3 (constant)
b	The number 10 (constant)

W	"to be a prime" (unary predicate)
E	"to be a square" (unary predicate)
L	"greater than" (binary predicate)
*	multiplication (binary predicate)
=	equality (binary predicate)

Exercise #3. Find another model for the following language, where the universe is the set N of natural numbers:

$\boldsymbol{symbol}$	$\boldsymbol{meaning}$
a	The number 3 (constant)
b	The number 10 (constant)
W	"to be a prime" (unary predicate)
E	"to be a square" (unary predicate)
L	"greater than" (binary predicate)
*	multiplication (binary predicate)
=	equality (binary predicate)

We define another model $\mathcal{M}$ where the universe is $A = \{0,1\}$ and:

$\boldsymbol{symbol}$	$\boldsymbol{meaning}$
a	The bit 0 (constant)
b	The bit 1 (constant)
W	"W(0) true, W(1) false"(unary predicate)
E	"E(0) true, E(1) true" (unary predicate)
L	"0L0 true, 0L1 false, 1L0 true, !L! false" (binary predicate)

*	"0*0=1, 0*1=0, 1*0=0, 1*1=1" (binary relation, like XOR)
=	equality (binary predicate)

Note: given that the model is finite, the meaning of the predicates can be written explicitly. Note that we can value the meaning of this sentence:

x	
0	$\exists y(E(x) \to W(y) \wedge xLy) \quad true$
1	$\exists y(E(x) \to W(y) \wedge xLy) \quad true$

…and therefore $\forall x \exists y(E(x) \to W(y) \wedge xLy)$ is true in the model we've build, $\mathcal{M}$.

Example #4. Consider the following axiomatic set:

- $W(a)$, $meaning$ a is a prime number
- $E(b)$
- $\forall x \exists y(xLy)$
- $\forall x\big(E(x) \to W(x)\big)$

Is $\forall x \exists y(E(x) \to W(y) \wedge xLy)$ a logical consequence of these axioms? To give a proper answer, we have to

$$\frac{\dots}{\forall x \exists y(E(x) \to W(y) \wedge xLy)}\,[\exists_i]$$

The second solution is to use Godel's *Completeness Theorem* according to which, given a set of axioms in a first-order language, from these axioms we can prove any formula if and only if for very model m, if m satisfies the axioms, then m satisfies the formula.It also means that we do not have a proof of the formula if and only if there exists a model M of the axioms Ax where the sentence is false. We can try to see if our sentence is *not* provable with these axioms.

Let's choose a model:

- Universe $A = \{0,1,2\}$

symbol	***meaning***
a	0 (constant)
b	1 (constant)
W	"W(1) true"
E	"E(1) true"
L	"0L0 true, 0L1 false, 1L0 true, !L! false" (binary predicate)
*	"0*0=1, 0*1=0, 1*0=0, 1*1=1" (binary relation, like XOR)
=	equality (binary predicate)

Lets make true these two axioms:

- $\forall x \exists y (xLy)$
- $\forall x \big(E(x) \rightarrow W(x)\big)$