IX. Mathematical Language

J.-R. Abrial (ETHZ)

February 2007

IX. Mathematical Language

1 Introduction

This chapter contains the definition of the *Mathematical Language* we use in this book. It is made of four sections introducing successively the Propositional Language (section 3), the Predicate Language (section 4), the Set-theoretic Language (section 5), and the Arithmetic Language (section 6). Each of these languages will be presented as an extension of the previous one. Before introducing these languages however, we shall give a brief summary of the Sequent Calculus (section 2).

2 Sequent Calculus

Before introducing the Sequent Calculus and Predicate Logic in a rigorous manner, it might be helpful to see how the ideas behind them are already implicitly present in "ordinary" mathematical proofs. For this purpose, we shall choose some examples in Geometry and Arithmetic. There are many ways of doing mathematical proofs. Here are a few of techniques, which are quite common:

1. We can perform the proof of a statement by deducing this statement from others which we have then to prove or which we have already proved. Usually the initial statement is to be proved under certain assumptions. Such a proof method is called *hypothetico-deductive*. It is very common.

2. Among the previously mentioned proofs, some are done by *contradiction*. For this we assume the negation of the statement we want to prove, and then deduce a contradiction.

3. Sometimes a proof decomposes itself in different exclusive cases: then we perform a, so-called, *proof by cases*.

4. In some occasions, it is useful to first prove something different from what we have to prove originally. Once it is done, we return to our original problem with an additional assumption corresponding to the new statement we have just proved. We have proved a, so called, *lemma*.

5. When dealing with Natural Numbers, we can do a proof *by induction*. This is to be done when the statement to be proved concerns all Natural Numbers n. We first prove the property for 0, and then we prove it for $n+1$ under the additional assumption that it holds for n. Such a proof method generalizes to any inductively generated set: sets of finite sequences, set of finite trees, and so on.

6. Sometimes a proof is just performed by doing an *algebraic calculation*. The calculation corresponds to applying some re-writing rules.

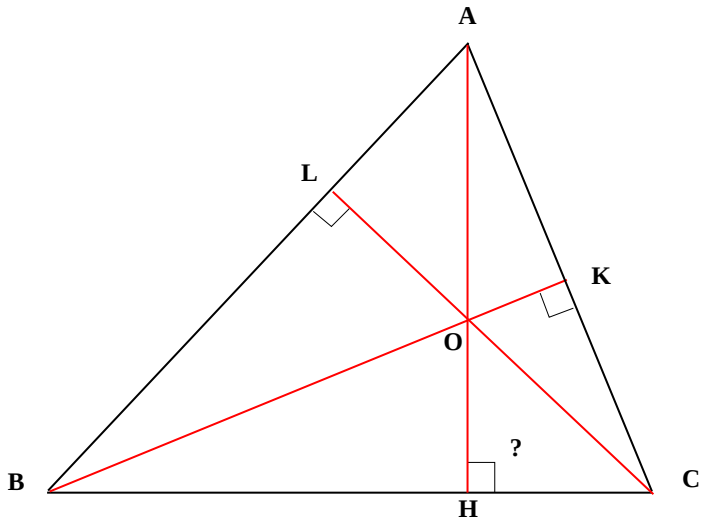
In what follows, we shall give some examples of such proof methods.

2.1 Deductive Proof in Geometry

Our first example is an hypothetico-deductive proof in Geometry. Here is what we want to prove:

Theorem: *The three altitudes of a non-right triangle meet in a single point.*

Given a triangle ABC, we draw two altitudes: BK and CL. They meet at point O. Now we have to prove that AO, intersecting BC in H, is also an altitude. All this is shown in the following figure:



We can be a little more precise, by stating what our *Hypotheses* are, namely:

- **Hypothesis 1**: BA and CL are perpendicular
- **Hypothesis 2**: L is on BA and distinct from B and A
- **Hypothesis 3**: BK and CA are perpendicular
- **Hypothesis 4**: K is on CA and distinct from C and A
- **Hypothesis 5**: BK and CL meet at O
- **Hypothesis 6**: AO meets BC in H

And now we can state what our *Goal* is:

- **Goal**: AH is perpendicular to BC

A statement to prove can always be expressed in this way. One has first to make very clear what our hypotheses and goal are. If the set of hypotheses is denoted by H and the goal by G, then the statement to prove, is called a *sequent*, and it is written as follows:

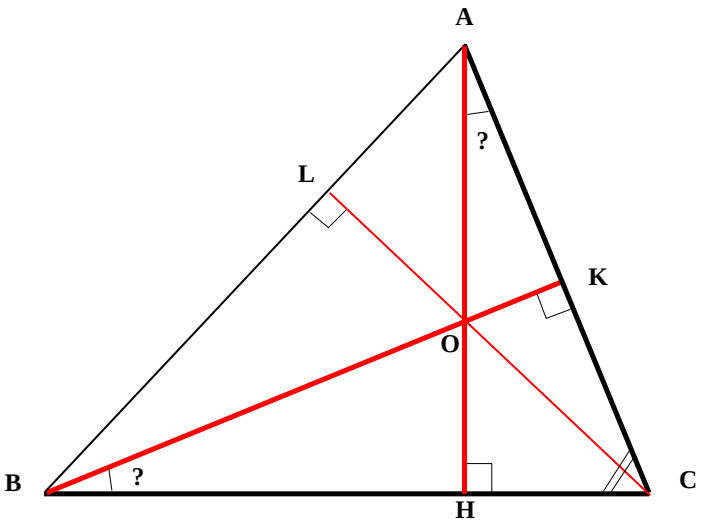
$$\mathsf{H} \vdash G$$

This is to be read: "H entails G". We could also say that we want to "prove the goal G under the hypotheses H". In our case, we can write it as follows:

BA and CL are perpendicular,
L is on BA and distinct from B and A,
BK and CA are perpendicular,
K is on CA and distinct from C and A, $\vdash$ AH is perpendicular to BC
BK and CL meet in O,
AO meets BC in H

In the rest of this section, we abbreviate this set of hypotheses by HYPS. In order to proceed, we have to see how we can take some facts in our mathematical knowledge of the triangle to find out a proof of the goal. As one knows, the sum of the interior angles of a triangle is constant. Let us consider the two

triangles KBC and HAC. Since they have the angle $\widehat{\text{BCA}}$ in common, and since BK is perpendicular to AC, then to prove that AH is perpendicular to BC, it is sufficient to prove that angles $\widehat{\text{KBC}}$ and $\widehat{\text{HAC}}$ are equal. In other words, to prove $\widehat{\text{AHC}} = \widehat{\text{BKC}}$, it is sufficient to prove $\widehat{\text{KBC}} = \widehat{\text{HAC}}$. This is illustrated in the following figure:



What we have implicitly applied in the previous statement is a *rule* allowing us to transform one sequent into other ones. More explicitly, this rule says that in order to prove that two angles are equal under certain hypotheses H, it is sufficient to prove that these angles are angles of two triangles having already their other two angles equal. Such a rule is called an *inference rule*. Given a triangle $t1$ with angles $\widehat{\text{a1}}$, $\widehat{\text{b1}}$ and $\widehat{\text{c1}}$, and a triangle $t2$ with angles $\widehat{\text{a2}}$, $\widehat{\text{b2}}$, and $\widehat{\text{c2}}$, our inference rule can be stated as follows:

$$\text{GEO1}\quad \frac{\mathsf{H} \vdash \widehat{\text{b1}} = \widehat{\text{b2}} \qquad \mathsf{H} \vdash \widehat{\text{c1}} = \widehat{\text{c2}}}{\mathsf{H} \vdash \widehat{\text{a1}} = \widehat{\text{a2}}}$$

More generally, an inference rule, named $\mathsf{r1}$, is made of two parts: the antecedent part and the consequent part. The antecedent part A is a set of sequents and the consequent part C is a single sequent. It is written like this:

$$\mathsf{r1}\quad \frac{\mathsf{A}}{C}$$

Such a rule can be read as follows: in order to have a proof of the sequent C it is sufficient to have a proof of each sequent in A. As can be seen our initial sequent to prove is now transformed by this rule into two sequents to prove. In our case, the two triangles are KBC and HAC, the angle $\widehat{\text{a1}}$ is $\widehat{\text{AHC}}$, the angle $\widehat{\text{a2}}$ is $\widehat{\text{BKC}}$, the angle $\widehat{\text{b1}}$ is $\widehat{\text{BCA}}$, the angle $\widehat{\text{b2}}$ is also $\widehat{\text{BCA}}$, finally the angle $\widehat{\text{c1}}$ is $\widehat{\text{KBC}}$, and the angle $\widehat{\text{b2}}$ is also $\widehat{\text{HAC}}$. As a consequence, our first sequent to prove is now transformed in the following two sequents:

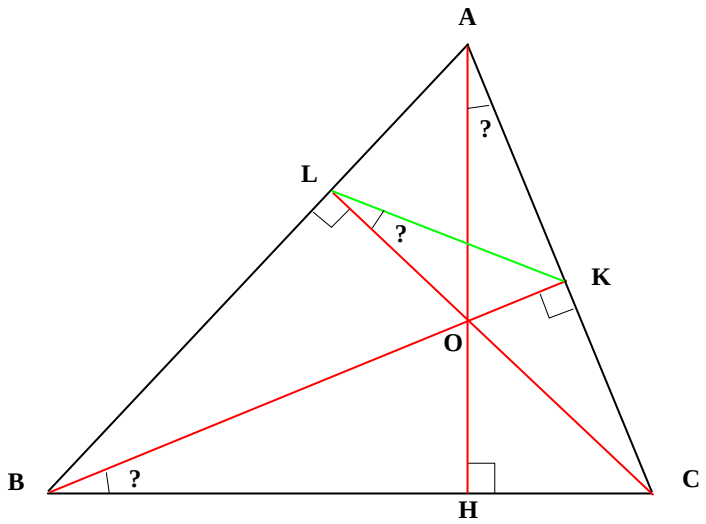
$$\mathsf{HYPS} \vdash \widehat{\mathrm{BCA}} = \widehat{\mathrm{BCA}}$$

$$\mathsf{HYPS} \vdash \widehat{\mathrm{KBC}} = \widehat{\mathrm{HAC}}$$

The first sequent is trivially proved by means of an inference rule on equality saying that every object is equal to itself. This can be stated as follows:

$$\mathsf{EQL1} \quad \frac{}{\mathsf{H} \vdash x = x}$$

We notice that this inference rule has an empty set of antecedents. It means that its consequent is proved without further proofs. In order to prove the second sequent, we draw the line LK, and try to prove that both angles $\widehat{\mathrm{KBC}}$ and $\widehat{\mathrm{HAC}}$ are equal to angle $\widehat{\mathrm{KLC}}$. This is illustrated in the following figure:



In doing that, we have implicitly applied another rule of equality which is the following:

$$\mathsf{EQL2} \quad \frac{\mathsf{H} \vdash x = z \qquad \mathsf{H} \vdash y = z}{\mathsf{H} \vdash x = y}$$

So we are left with the following two sequents to prove:

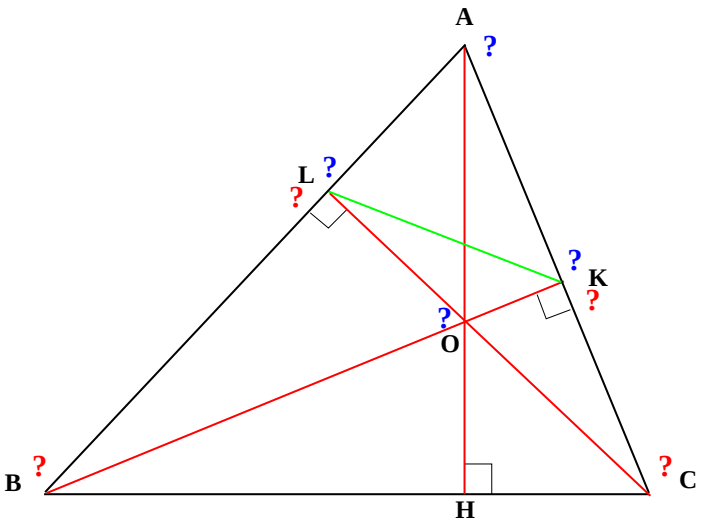
$$\mathsf{HYPS} \vdash \widehat{\mathrm{KBC}} = \widehat{\mathrm{KLC}}$$

$$\mathsf{HYPS} \vdash \widehat{\mathrm{KAO}} = \widehat{\mathrm{KLO}}$$

Note that the goal of the second sequent should have been $\widehat{\mathrm{HAC}} = \widehat{\mathrm{KLC}}$. We have replaced angle $\widehat{\mathrm{HAC}}$ by angle $\widehat{\mathrm{KAO}}$ and angle $\widehat{\mathrm{KLC}}$ by angle $\widehat{\mathrm{KLO}}$. This has been possible since (at least in the shown figure) O is situated in between A and H, K is situated between A and C, and O is situated between L and C.

Doing these replacements involve applying some more geometrical inference rules and another equality rule which we accept implicitly here to simplify matters.

In order to prove these equalities, we remember the following geometrical rule: when four points a, b, c, and d are on a circle in that order, then angles $\widehat{acd}$ and $\widehat{abd}$ are equal. We have then to prove that points K, L, B, and C are on a circle. And also that points O, L, A, K are on a circle. This is illustrated in the following figure:



This could be formalized by the following geometrical rule:

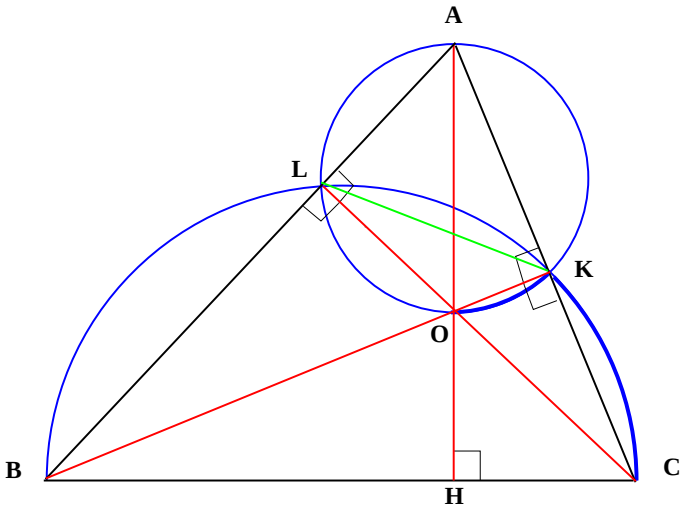
$$\textsf{GEO2}\quad \frac{\mathsf{H} \vdash \text{Points } a, b, c, \text{ and } d \text{ are on a circle}}{\mathsf{H} \vdash \widehat{acd} = \widehat{abd}}$$

We are now left to prove the following sequents

$$\mathsf{HYPS} \vdash \text{Points K, L, B, and C are on a circle}$$

$$\mathsf{HYPS} \vdash \text{Points K, L, A, and O are on a circle}$$

The final geometrical knowledge we are going to apply now concerns right triangles. Given two right triangles abc and adc, then a, b, c, and d are on the same circle. In other words, we have to prove that triangles BLC, BKC, OLA, and OKA are all right triangles. More precisely, we have then to prove that BL and CL are perpendicular as well as BK and CK, and also OL and AL as well as OK and AK. This is illustrated in the following figure:



This could be formalized by the following geometrical rule:

$$\textsf{GEO3}\quad \frac{\begin{array}{l}\mathsf{H} \vdash cb \text{ and } db \text{ are perpendicular}\\ \mathsf{H} \vdash ca \text{ and } da \text{ are perpendicular}\end{array}}{\mathsf{H} \vdash \text{Points } a, b, c, \text{ and } d \text{ are on a circle}}$$

We are now left to prove the following four sequents:

$$\textsf{HYPS} \vdash \text{BL and CL are perpendicular}$$

$$\textsf{HYPS} \vdash \text{BK and CK are perpendicular}$$

$$\textsf{HYPS} \vdash \text{AL and OL are perpendicular}$$

$$\textsf{HYPS} \vdash \text{AK and OK are perpendicular}$$

The first goal is easily *deductible from the hypotheses*. Hypothesis 1 tells us the BA and CL are perpendicular. And Hypothesis 2 tells us that L is on BA. We can thus replace BA by BL (if B and L are distinct which they are by hypothesis). This can be formalized by the following geometrical inference rule:

$$\textsf{GEO4}\quad \frac{\begin{array}{l}\mathsf{H} \vdash ad \text{ and } cb \text{ are perpendicular}\\ \mathsf{H} \vdash b \text{ is on } ad \text{ and distinct from } a\end{array}}{\mathsf{H} \vdash ab \text{ and } cb \text{ are perpendicular}}$$

So, for proving our first sequent, we are left to prove the following two sequents:

$$\textsf{HYPS} \vdash \text{BA and CL are perpendicular}$$

$$\textsf{HYPS} \vdash \text{L is on BA and distinct from B}$$

But these two sequents are "obvious" since they are included in our hypotheses. This can be formalized by means of the following general inference rule called HYP.

$$\textsf{HYP}\quad \frac{}{\mathsf{H}, P \ \vdash \ P}$$

The last three sequents that remain to be proved could be proved in a similar manner. It is now possible to summarize the proof in the following denser form:

1	HYPS $\vdash$ AH is perpendicular to BC	GEO1
2	HYPS $\vdash$ $\widehat{\mathsf{BCA}} = \widehat{\mathsf{BCA}}$	EQL1
3	HYPS $\vdash$ $\widehat{\mathsf{KBC}} = \widehat{\mathsf{HAC}}$	EQL2
4	HYPS $\vdash$ $\widehat{\mathsf{KBC}} = \widehat{\mathsf{KLC}}$	GEO2
5	HYPS $\vdash$ K, L, B, and C on a circle	GEO3
6	HYPS $\vdash$ BL is perpendicular to CL	GEO4
7	HYPS $\vdash$ BA is perpendicular to CL	HYP
8	HYPS $\vdash$ L is on BA and distinct from B	HYP
9	HYPS $\vdash$ BK is perpendicular to CK	GEO4
10	HYPS $\vdash$ ...	...
11	HYPS $\vdash$ $\widehat{\mathsf{KAO}} = \widehat{\mathsf{KLO}}$	GEO2
12	HYPS $\vdash$ K, L, A, and O on a circle	GEO3
13	HYPS $\vdash$ OL is perpendicular to AL	GEO4
14	HYPS $\vdash$ ...	...
15	HYPS $\vdash$ OK is perpendicular to AK	GEO4
16	HYPS $\vdash$ ...	...

As can be seen, each line contains a sequent and the first line contains the sequent we have to prove. Each line also contains the rule that is applied in order to prove the corresponding sequent. Then the new sequents to prove are shifted to indicate the dependency. For instance, on line 3 you can see the sequent HYPS $\vdash$ $\widehat{\mathsf{KBC}} = \widehat{\mathsf{HAC}}$ together with rule EQL2 at the end of the line. The two new sequents to be proved are then shown on lines 4 and line 11. This layout shows the *tree structure of the proof*.

2.2 A Proof by Contradiction in Arithmetic

The example we propose now is very famous. It has been known since the ancient Greeks. This is the very classical example of a proof by contradiction. We shall also make use of a lemma in this proof. We want to prove the following:

Theorem: $\sqrt{2}$ *is irrational*.

The proof is done by contradiction. It proceeds as follows: we suppose that $\sqrt{2}$ is rational. And we shall derive a contradiction. Here is thus our hypothesis

Hypothesis 1: $\sqrt{2}$ is rational

According to this hypothesis, $\sqrt{2}$ can be put under the following form:

$$\sqrt{2} \ = \ p/q \quad (1)$$

where p and q are two Natural Numbers which must fulfil the following two conditions, which are thus additional hypotheses:

Hypothesis 2: q is not equal to 0

Hypothesis 3: p and q have no common divisor

According to Hypothesis 2 we can multiply by q both parts of equality (1) and then square both parts of the result, yielding:

$$2q^2 \ = \ p^2 \quad (2)$$

Hence p^2 is even. We shall prove below a lemma telling us that when a square such as p^2 is even then so is p. As a consequence, p can be written as follows

$$p \ = \ 2n \quad (3)$$

Replacing p in (2) by its value in (3), we obtain:

$$2q^2 \ = \ 4n^2 \quad (4)$$

Hence, we have

$$q^2 \ = \ 2n^2 \quad (5)$$

Thus q^2 is even and also q according to the same lemma, which we have not yet proved. But what we have just proved is that both p and q are divisible by 2. In other words, they both have 2 as a divisor. This *contradicts* Hypothesis 3 which says that p and q have no common divisor. This achieves the proof of our theorem.

It remains for us to prove the lemma we mentioned.

Lemma: *If a Natural Number p is such that p^2 is even then p is even.*

Here we have an assumption

Hypothesis 1: p^2 is even

The proof proceeds by contradiction too. So we suppose that p is odd and try to derive a contradiction

Hypothesis 2: p is odd

From Hypothesis 2, we can write p as follows

$$p = 2n + 1$$

Thus

$$p^2 = 4n^2 + 4n + 1$$

Thus p^2 is odd too. But this contradicts Hypothesis 1 telling us that p^2 is even.

2.3 Recursive Definition and Proof by Induction in Arithmetic

Our next example shows a proof by induction on Natural Numbers. We first define a certain quantity recursively and then derive a property of this quantity by induction.

The sum of the n first natural numbers can be defined recursively as follows. First for 0, and then for $n+1$ in term of its value for n. Formally:

$$(1) \quad \sum_{i=1}^{0} i = 0 \qquad\qquad (2) \quad \sum_{i=1}^{n+1} i = (n+1) + \sum_{i=1}^{n} i$$

We want to prove the following:

Theorem: *Twice the sum of the first n Natural Numbers is equal to $n.(n+1)$.*

$$2.\sum_{i=1}^{n} i = n.(n+1)$$

The inductive proof proceeds as follows. We have to cases: the *base case*, and then the *inductive case*. In the base case, we prove that the theorem is true when n is equal to 0. In the inductive case, we prove that the theorem is true for $n+1$ under the hypothesis that it is already true for n.

Base Case

When n is 0, we have according to (1)

$$2.\sum_{i=1}^{0} i = 0 = 0.(0+1)$$

Inductive Case

We assume that the theorem holds for n

Hypothesis: $2.\sum_{i=1}^{n} i = n.(n+1)$

According to (2), we have

$$2.\sum_{i=1}^{n+1} i = 2.(n+1) + 2.\sum_{i=1}^{n} i$$

that is, according to the Hypothesis

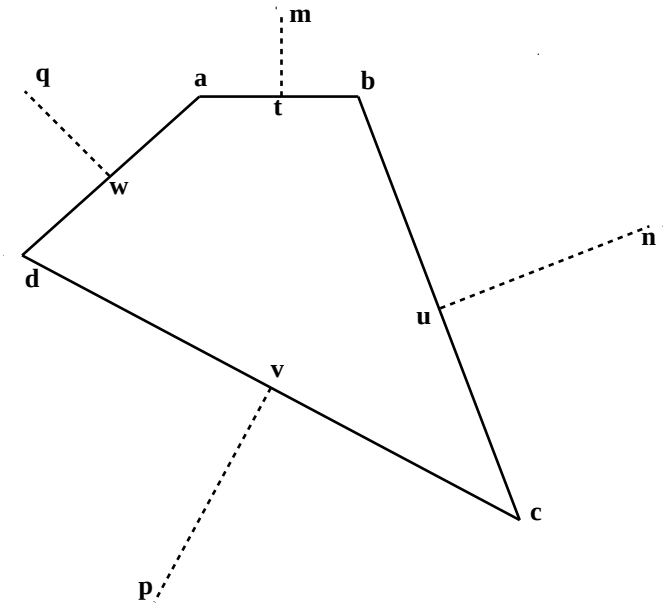
$$2.\sum_{i=1}^{n+1} i = 2.(n+1) + n.(n+1) = (n+1).(n+2)$$

2.4 Proof by Calculation

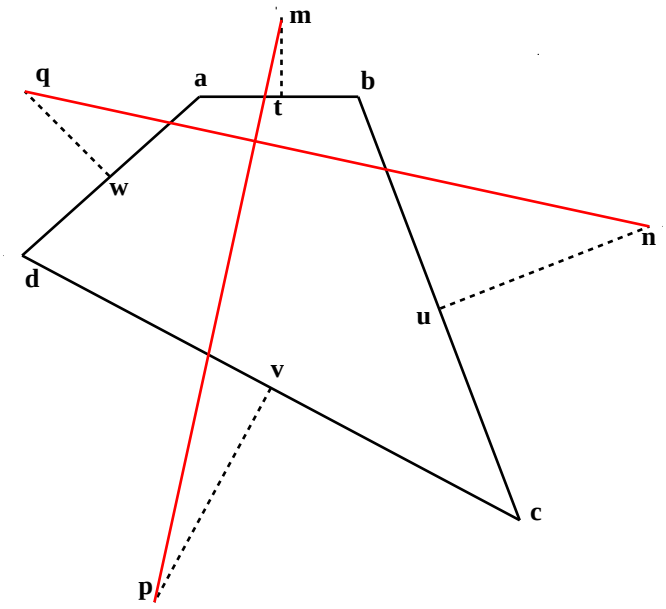
Our last example is again taken in Geometry. But the proof we give here is not at first glance a hypothetico-deductive proof as the one proposed in our first example. It is a proof which is entirely done by calculation. We have the feeling that such a proof, although certainly correct, is a bit frustrating. It is almost always the

case with a proof based on a calculation. In other words, we can "see" the proof but we do not understand the deeper geometrical reason for this theorem to be true. We shall show however that the calculation will help us finding this geometrical reason.

We are given 4 points a, b c and d on a plane. Let t, u, v, and w be the middle of ab, bc, cd, and da respectively. From these points we draw the segments tm, un, vp, and wq in such a way that these segments are perpendicular to ab, bc, cd, and da respectively. Moreover tm, un, vp, and wq are equal to the half of ab, bc, cd, and da respectively. This is indicated in the following figure:



And now we join m to p and q to n as indicated below:



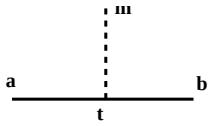
We want to prove the rather surprising fact:

Theorem: *mp and qn are equal and perpendicular*

We are going to do a vector calculation using complex numbers. Let $\boldsymbol{mp}$ be the complex number representing the vector mp. Likewise, let $\boldsymbol{qn}$ be the complex representation of the vector qn. The fact that mp and qn are equal and perpendicular implies proving the following:

$$\boldsymbol{qn} = i.\boldsymbol{mp}$$

What we are going to do now is simply to compute these vectors using complex number calculations. The following figure will help us to perform the initial calculation. Note that $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{t}$, and $\boldsymbol{m}$ are also complex numbers in this calculation.



$\boldsymbol{t}$	$=$	$(\boldsymbol{a}+\boldsymbol{b})/2$	t is the middle of ab
$\boldsymbol{tb}$	$=$	$(\boldsymbol{b}-\boldsymbol{a})/2$	$\boldsymbol{tb} = \boldsymbol{b} - \boldsymbol{t}$
$\boldsymbol{tm}$	$=$	$i.(\boldsymbol{b}-\boldsymbol{a})/2$	rotation: $\pi/2$
$\boldsymbol{m}$	$=$	$(\boldsymbol{a}+\boldsymbol{b})/2 + i.(\boldsymbol{b}-\boldsymbol{a})/2$	$\boldsymbol{m} = \boldsymbol{t} + \boldsymbol{tm}$
$\boldsymbol{p}$	$=$	$(\boldsymbol{c}+\boldsymbol{d})/2 + i.(\boldsymbol{d}-\boldsymbol{c})/2$	permutation: $a \mapsto c, b \mapsto d$
$\boldsymbol{mp}$	$=$	$((\boldsymbol{c}+\boldsymbol{d}) - (\boldsymbol{a}+\boldsymbol{b}))/2 + i.((\boldsymbol{d}+\boldsymbol{a}) - (\boldsymbol{b}+\boldsymbol{c}))/2$	$\boldsymbol{mp} = \boldsymbol{p} - \boldsymbol{m}$
$\boldsymbol{qn}$	$=$	$((\boldsymbol{b}+\boldsymbol{c}) - (\boldsymbol{d}+\boldsymbol{a}))/2 + i.((\boldsymbol{c}+\boldsymbol{d})) - (\boldsymbol{a}+\boldsymbol{b}))/2$	permutation: $a \mapsto d, b \mapsto a, c \mapsto b, d \mapsto c$
$i.\boldsymbol{mp}$	$=$	$((\boldsymbol{b}+\boldsymbol{c}) - (\boldsymbol{d}+\boldsymbol{a}))/2 + i.((\boldsymbol{c}+\boldsymbol{d})) - (\boldsymbol{a}+\boldsymbol{b}))/2$	reminder: $i^2 = -1$

Therefore, we have indeed the expected result, namely

$$\boldsymbol{qn} = i.\boldsymbol{mp}$$

Again, this result is frustrating because we do not understand the geometrical reason. Clearly, if this segments are equal and perpendicular, there must exist a 90 degrees rotation that maps n to m and p to q. The problem is to guess what the center of this rotation could be. Let o be the middle of the diagonal ac. We guess that o is the center of this rotation. If our guess is correct, we have then to prove the following vector equalities:

$$\boldsymbol{om} = i.\boldsymbol{on} \qquad \boldsymbol{op} = i.\boldsymbol{oq}$$

The calculation goes as follows:

$$
\begin{aligned}
\boldsymbol{o} &= (\boldsymbol{a}+\boldsymbol{c})/2 \\
\boldsymbol{n} &= (\boldsymbol{b}+\boldsymbol{c})/2 + i.(\boldsymbol{c}-\boldsymbol{b})/2 \\
\boldsymbol{m} &= (\boldsymbol{a}+\boldsymbol{b})/2 + i.(\boldsymbol{b}-\boldsymbol{a})/2 \\
\boldsymbol{on} &= (\boldsymbol{b}-\boldsymbol{a})/2 + i.(\boldsymbol{c}-\boldsymbol{b})/2 \\
\boldsymbol{om} &= (\boldsymbol{b}-\boldsymbol{c})/2 + i.(\boldsymbol{b}-\boldsymbol{a})/2 \\
i.\boldsymbol{on} &= (\boldsymbol{b}-\boldsymbol{c})/2 + i.(\boldsymbol{b}-\boldsymbol{a})/2
\end{aligned}
$$

permutation: $b \mapsto a, c \mapsto b$

$$\boldsymbol{on} = \boldsymbol{n} - \boldsymbol{o}$$

$$\boldsymbol{om} = \boldsymbol{m} - \boldsymbol{o}$$

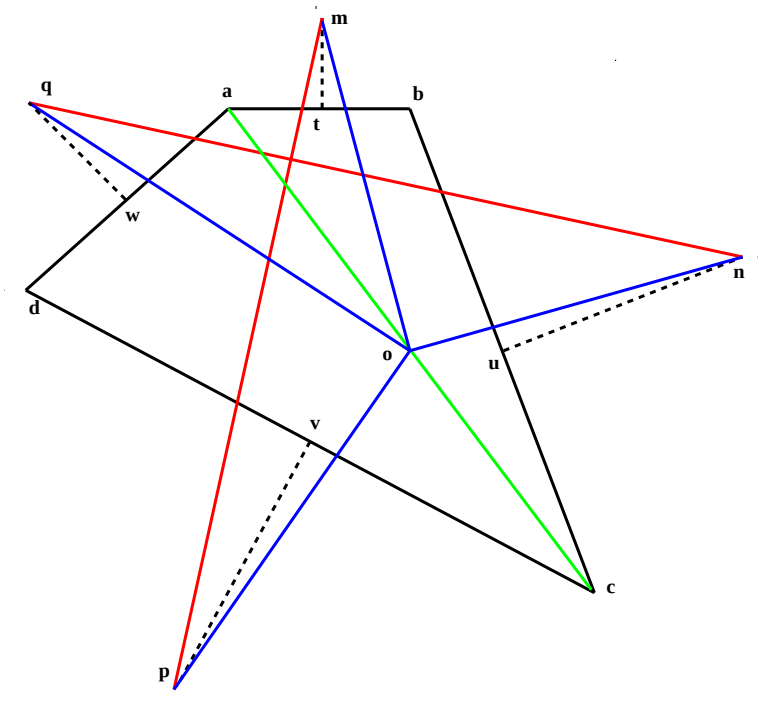
reminder: $i^2 = -1$

Thus we have indeed

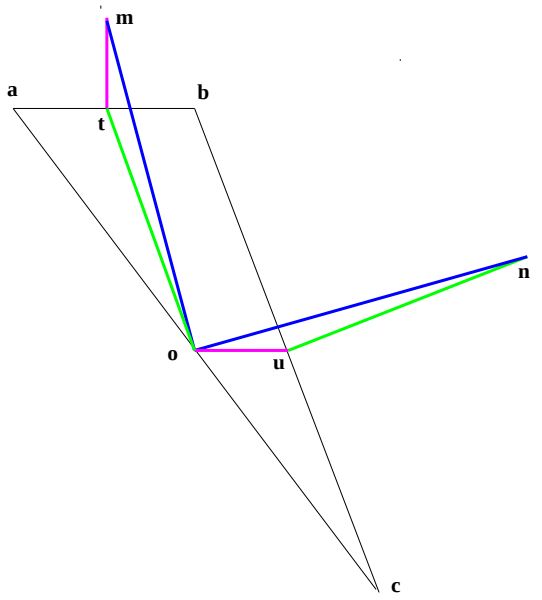
$$\boldsymbol{om} = i.\boldsymbol{on}$$

The calculations of $\boldsymbol{op}$ and $\boldsymbol{oq}$ could be conducted in a similar manner.

This can be illustrated as follows:



And now we understand the geometrical "deeper" reason. It is illustrated in the following figure, where it is now easy to prove geometrically that both triangles tmo and oun are equal and perpendicular.



This proof where we use the calculation to help us finding the geometrical reason is by analogy typical of the kinds of reasoning that we would like to perform in our computer system developments.

2.5 Definitions of the Sequent Calculus

In this section, we give some more formal definitions to refresh on what we have seen in the previous sections, particularly in the first example.

(1) A *sequent* is a generic name for "something we want to prove". For the moment, this is just an informally defined notion, which we shall refine later. In what follows we shall use identifiers such as $S1$, $S2$, etc. to denote sequents. The important thing to note at this point is that we can associate a *proof* with a sequent. For the moment, we do not know what a proof is however. It will only be defined at the end of this section.

(2) An *inference rule* is a device used to construct proofs of sequents. It is made of two parts: the *antecedent* part and the *consequent* part. The antecedent denotes a finite set of sequents while the consequent denotes a single sequent. An inference rule, named say **r1**, with antecedent A and consequent C is usually written as follows:

$$\textbf{r1} \quad \frac{\mathsf{A}}{C}$$

It is to be read:

> Inference Rule **r1** yields a proof of sequent C as soon as we have proofs of each sequent of A

Note that the antecedent A might be empty. In this case, the inference rule, named say **r2**, is written as follows:

$$\textbf{r2}\ \ \frac{}{C}$$

And it is to be read:

Inference Rule **r2** yields a proof of sequent C

(3) A *Theory* is a set of inference rules.

(4) It is now possible to give the definition of the *proof of a sequent* within a theory T. It is simply a finite tree with certain constraints. The nodes of such a tree have two components: a sequent s and a rule r of the theory T. Here are the constraints for each node of the form (s, r): the consequent of the rule r is s, and the children of this node are nodes whose sequents are exactly all the sequents of the antecedent of rule r. As a consequence, the leaves of the tree contain rules with no antecedent. Moreover, the top node of the tree contains the sequent to be proved. As an example, let be given the following theory:

r1	$\frac{}{S2}$	**r2**	$\frac{S7}{S4}$	**r3**	$\frac{S2\ \ S3\ \ S4}{S1}$	**r4**	$\frac{}{S5}$	**r5**	$\frac{S5\ \ S6}{S3}$	**r6**	$\frac{}{S6}$	**r7**	$\frac{}{S7}$

Here is a proof of the sequent $S1$:

$$\begin{array}{ccccc} & & (S1, \textbf{r3}) & & \\ & \swarrow & \downarrow & \searrow & \\ (S2, \textbf{r1}) & & (S3, \textbf{r5}) & & (S4, \textbf{r2}) \\ & \swarrow & \downarrow & & \downarrow \\ (S5, \textbf{r4}) & & (S6, \textbf{r6}) & & (S7, \textbf{r7}) \end{array}$$

As can be seen, the root of the tree has sequent $S1$, which is the one we want to prove. And it is easy to check that each node, say node $(S3, \textbf{r5})$, is indeed such that the consequent of its rule is the sequent of the node: $S3$, in this case, is the consequent of rule **r5**. Moreover, we can check that the sequents of the child nodes of node $(S3, \textbf{r5})$, namely, $S5$ and $S6$, are exactly the sequents forming the antecedents of rule **r5**. This tree can be represented vertically as indicated below. In this text, we shall adopt this representation.

$S1$	**r3**
$\quad S2$	**r1**
$\quad S3$	**r5**
$\quad\quad S5$	**r4**
$\quad\quad S6$	**r6**
$\quad S4$	**r2**
$\quad\quad S7$	**r7**

2.6 Sequents for a Mathematical Language

We now refine our notion of sequent in order to define the way we shall make proofs with our Mathematical Language. Such a language contains constructs called *Predicates*. For the moment, this is all what we know about our Mathematical Language. Within this framework, a sequent S, as defined in the previous section, now becomes a more complex object. It is made of two parts: the *hypotheses* part and the *goal* part. The hypothesis part denotes a finite set of predicates while the goal part denotes a single predicate. A sequent with hypotheses H and goal G is written as follows:

$$\mathsf{H} \vdash G$$

This sequent is to be read as follows:

Under the set of hypotheses H, prove the goal G

This is the sort of sequents we want to prove. It is also the sort of sequents we shall have in the theories associated with our Mathematical Language.

Note that the set of hypotheses of a sequent might be empty. As a practical notation, a set of hypotheses H to which we add an extra hypothesis P, is simply written as: H,P

2.7 Initial Theory

We now have enough elements at our disposal to define the first rules of our proving theory. Note again that we still don't know what a predicate is. We just know that predicates are constructs we shall be able to define within our future Mathematical Language. We start with three basic rules which we first state informally. They are called **HYP**, **MON**, and **CUT**.

HYP: If the goal P of a sequent belongs to the set of hypotheses H of this sequent, then it is proved.

MON: Once a sequent is proved, any sequent with the same goal and more hypotheses is also proved.

CUT: If you succeed in proving a predicate P under a set of hypotheses H, then P can be added to the set of hypotheses H for proving a goal Q.

These rules can be encoded as follows:

$$\textbf{HYP}\ \frac{}{\mathsf{H},\ P\ \vdash\ P} \qquad \textbf{MON}\ \frac{\mathsf{H}\ \vdash\ Q}{\mathsf{H},\ P\ \vdash\ Q} \qquad \textbf{CUT}\ \frac{\mathsf{H}\ \vdash\ P \qquad \mathsf{H},\ P\ \vdash\ Q}{\mathsf{H}\ \vdash\ Q}$$

The previous theory can be given a more convenient tabular form, which we shall adopt in what follows. We name it $T0$:

	Antecedents	Consequent
HYP		$\mathsf{H},\ P\ \vdash\ P$
MON	$\mathsf{H}\ \vdash\ Q$	$\mathsf{H},\ P\ \vdash\ Q$
CUT	$\mathsf{H}\ \vdash\ P$ $\mathsf{H},\ P\ \vdash\ Q$	$\mathsf{H}\ \vdash\ Q$

$T0$

Note that in the previous rules, the letter H, P and Q are, so-called, *meta-variables*. The letter H is a meta-variable standing for a finite set of predicates, whereas the letter P and Q are meta-variables standing for predicates. Clearly then, each of the previous "rules" stands for more than just one rule: it is better to call it a *rule schema* or a generic rule. This will always be the case in what follows.

3 The Propositional Language

In this section we present a first simple version of our Mathematical Language, it is called the Propositional Language. It will be later refined to more complete versions.

3.1 Syntax

Our first version is built around three constructs called *conjunction*, *implication*, and *negation*. Given two predicates P and Q, we can construct their conjunction $P \wedge Q$ and their implication $P \Rightarrow Q$. And given a predicate P, we can construct its negation $\neg P$. This can be formalized by means of the following syntax:

$$\begin{array}{rcl} predicate & ::= & \neg\, predicate \\ & & predicate \wedge predicate \\ & & predicate \Rightarrow predicate \end{array} \qquad SYNTAX1$$

This syntax is clearly ambiguous, but we do not care about it at this stage. In this text, in order to avoid ambiguities, we shall provide as many pairs of parentheses as needed. Also note that this syntax does not contain any "base" predicate: such predicates will come later.

3.2 Enlarging the Initial Theory

The initial theory $T0$ of section 2.7 is enlarged with the following inference rules forming Theory $T1$:

	Antecedents	Consequent
R1	$\mathsf{H} \vdash P$ $\mathsf{H} \vdash Q$	$\mathsf{H} \vdash P \wedge Q$
R2	$\mathsf{H} \vdash P \wedge Q$	$\mathsf{H} \vdash P$
R3	$\mathsf{H} \vdash P \wedge Q$	$\mathsf{H} \vdash Q$
R4	$\mathsf{H}, P \vdash Q$	$\mathsf{H} \vdash P \Rightarrow Q$

$T1$

	Antecedents	Consequent
R5	$\mathsf{H} \vdash P \Rightarrow Q$	$\mathsf{H}, P \vdash Q$
R6	$\mathsf{H}, \neg Q \vdash P$ $\mathsf{H}, \neg Q \vdash \neg P$	$\mathsf{H} \vdash Q$
R7	$\mathsf{H}, Q \vdash P$ $\mathsf{H}, Q \vdash \neg P$	$\mathsf{H} \vdash \neg Q$

$T1$

As can be seen, rules **R1**, **R2**, and **R3** are used to eliminate or introduce the $\wedge$ operator. Rules **R4** and **R5** are used to eliminate or introduce the $\Rightarrow$ operator. And rules **R6** and **R7** are used to do proofs by contradiction.

3.3 Replacing the Previous Theory

The previous Theory $T1$ is very natural and clearly in full accordance with our intuitive understanding of conjunction, implication and negation. But it suffers a very important drawback: it is not very convenient to use it to construct practical proofs because it offers too many possibilities.

We shall then propose another Theory called $S1$. Theory $S1$ can be constructed systematically from Theory $T1$ but we shall not do this construction here. Theory $S1$ is certainly far less natural than Theory $T1$, but it offers the great advantage over Theory $T1$ to be almost deterministic. Almost only, but, at the end of this section, we shall indicate a certain way of using it which makes it completely deterministic: it means that at each step of the proof tree construction we shall have only one possibility of choosing an applicable inference rule or no possibility at all in case of failure.

In fact, Theory $S1$ defines a, so-called, *proof procedure* for the simple Proposition Language so far defined. It can be easily mechanized. But before doing this, we have to extend our proposition Language with one predicate: $\perp$. This predicate is just used in this theory. In other words, users of the Mathematical Language are not allowed to use it. Here is Theory $S1$:

	Antecedents	Consequent
INI	$\mathsf{H} \vdash \neg R \Rightarrow \perp$	$\mathsf{H} \vdash R$
AXM		$\mathsf{H}, P, \neg P \vdash R$
AND1	$\mathsf{H} \vdash \neg Q \Rightarrow R$ $\mathsf{H} \vdash \neg P \Rightarrow R$	$\mathsf{H} \vdash \neg(P \wedge Q) \Rightarrow R$
AND2	$\mathsf{H} \vdash P \Rightarrow (Q \Rightarrow R)$	$\mathsf{H} \vdash (P \wedge Q) \Rightarrow R$

$S1$

	Antecedents	Consequent
IMP1	$\mathsf{H} \vdash P \Rightarrow (\neg Q \Rightarrow R)$	$\mathsf{H} \vdash \neg(P \Rightarrow Q) \Rightarrow R$
IMP2	$\mathsf{H} \vdash Q \Rightarrow R$ $\mathsf{H} \vdash \neg P \Rightarrow R$	$\mathsf{H} \vdash (P \Rightarrow Q) \Rightarrow R$
NEG	$\mathsf{H} \vdash P \Rightarrow R$	$\mathsf{H} \vdash \neg\neg P \Rightarrow R$
DED	$\mathsf{H}, P \vdash R$	$\mathsf{H} \vdash P \Rightarrow R$

$S1$

This theory has to be used with a, so-called, tactic telling us in which order the rules have to be applied. Here is the tactic, where RULES is one of **AXM**, **IMP1**, **IMP2**, **AND1**, **AND2**, **NEG**:

INI ; (RULES* ; **DED**)*

This tactic has to be read as follows: First use rule **INI** once. Then start the following process: use any rule in RULES as long as it is possible, then use **DED** once, and finally re-start this process.

3.4 Example

Here is an example of using the previous proof procedure. It is used to prove the following sequent:

$$\vdash (A \Rightarrow B) \Rightarrow ((A \wedge C) \Rightarrow (B \wedge C))$$

Proof

1	$\vdash (A \Rightarrow B) \Rightarrow ((A \wedge C) \Rightarrow (B \wedge C))$	**INI**
2	$\vdash \neg((A \Rightarrow B) \Rightarrow ((A \wedge C) \Rightarrow (B \wedge C))) \Rightarrow \bot$	**IMP1**
3	$\vdash (A \Rightarrow B) \Rightarrow (\neg((A \wedge C) \Rightarrow (B \wedge C)) \Rightarrow \bot)$	**IMP2**
4	$\vdash B \Rightarrow (\neg((A \wedge C) \Rightarrow (B \wedge C)) \Rightarrow \bot)$	**DED**
5	$B \vdash \neg((A \wedge C) \Rightarrow (B \wedge C)) \Rightarrow \bot$	**IMP1**
6	$B \vdash (A \wedge C) \Rightarrow (\neg(B \wedge C) \Rightarrow \bot)$	**AND2**
7	$B \vdash A \Rightarrow (C \Rightarrow (\neg(B \wedge C) \Rightarrow \bot))$	**DED**
8	$B, A \vdash C \Rightarrow (\neg(B \wedge C) \Rightarrow \bot)$	**DED**
9	$B, A, C \vdash \neg(B \wedge C) \Rightarrow \bot$	**AND1**
10	$B, A, C \vdash \neg C \Rightarrow \bot$	**DED**
11	$B, A, C, \neg C \vdash \bot$	**AXM**
12	$B, A, C \vdash \neg B \Rightarrow \bot$	**DED**
13	$B, A, C, \neg B \vdash \bot$	**AXM**
14	$\vdash \neg A \Rightarrow (\neg((A \wedge C) \Rightarrow (B \wedge C)) \Rightarrow \bot)$	**DED**
15	$\neg A \vdash \neg((A \wedge C) \Rightarrow (B \wedge C)) \Rightarrow \bot$	**IMP1**
16	$\neg A \vdash (A \wedge C) \Rightarrow (\neg(B \wedge C) \Rightarrow \bot)$	**AND2**
17	$\neg A \vdash A \Rightarrow (C \Rightarrow (\neg(B \wedge C) \Rightarrow \bot))$	**DED**
18	$\neg A, A \vdash C \Rightarrow (\neg(B \wedge C) \Rightarrow \bot)$	**AXM**

As can be seen, this proof procedure must be mechanized: there is no point in doing such a proof manually since it is easily mechanizable.

3.5 Methodology

The method we are going to use to build our Mathematical Language will be very systematic. It consists in subsequently augmenting our syntax, and correlatively augmenting our available inference rules. At each step of the construction, we shall have a *natural* Theory T_i and a corresponding *practical* Theory S_j which is derived from T_i.

We shall use two different approaches for extending our language. Either the extension corresponds to a simple facility. In other words, the new construct can entirely be defined in terms of existing ones. In that case, we shall only augment the current practical Theory S_j. Or the new construct is definitely new and not related to any previous construct. In that case, we shall proceed differently. We first augment the current natural Theory T_i and then derive from it a corresponding augmentation of the current practical Theory S_j.

3.6 Extending the Proposition Language

The Proposition Language is now extended by adding two more constructs called *disjunction* and *equivalence*. Given two predicates P and Q, we can construct their disjunction $P \vee Q$ and their equivalence $P \Leftrightarrow Q$. We also add one predicate: $\top$. As a consequence, our syntax is now the following, where we have underlined the new constructs:

$$
\begin{array}{rcl}
predicate & ::= & \bot \\
 & & \underline{\top} \\
 & & \neg\, predicate \\
 & & predicate \wedge predicate \\
 & & \underline{predicate \vee predicate} \\
 & & \underline{predicate \Rightarrow predicate} \\
 & & \underline{predicate \Leftrightarrow predicate}
\end{array}
\qquad SYNTAX2
$$

Such extensions are defined in terms of previous ones by mere rewriting rules:

Predicate	Rewritten
$\top$	$\neg \bot$
$P \vee Q$	$\neg P \Rightarrow Q$
$P \Leftrightarrow Q$	$(P \Rightarrow Q) \wedge (Q \Rightarrow P)$

For convenience, Theory $S1$ can be extended with the following rules which can be established easily after applying the previous rewriting rules:

	Antecedents	Consequent
OR1	$\mathsf{H} \vdash \neg P \Rightarrow (\neg Q \Rightarrow R)$	$\mathsf{H} \vdash \neg(P \lor Q) \Rightarrow R$
OR2	$\mathsf{H} \vdash Q \Rightarrow R$ $\mathsf{H} \vdash P \Rightarrow R$	$\mathsf{H} \vdash (P \lor Q) \Rightarrow R$
EQV1	$\mathsf{H} \vdash P \Rightarrow (\neg Q \Rightarrow R)$ $\mathsf{H} \vdash \neg P \Rightarrow (Q \Rightarrow R)$	$\mathsf{H} \vdash \neg(P \Leftrightarrow Q) \Rightarrow R$
EQV2	$\mathsf{H} \vdash P \Rightarrow (Q \Rightarrow R)$ $\mathsf{H} \vdash \neg P \Rightarrow (\neg Q \Rightarrow R)$	$\mathsf{H} \vdash (P \Leftrightarrow Q) \Rightarrow R$

$S2$

4 The Predicate Language

In this section, we introduce the Predicate Language. The syntax is extended with a number of new kinds of predicates and also with the introduction of two new syntactic categories called *expression* and *variable*. A variable is a simple identifier. Given a list of variables x made of distinct identifiers and a predicate P, the construct $\forall x \cdot P$ is called a *universally quantified predicate*. Given a list of variables x made of distinct identifiers, a list of expressions E of the same size as that of x, and a predicate P, the construct $[x := E]P$ is called a *substituted predicate*. An expression is either a variable, a *substituted expression* formed in exactly the same way as a substituted predicate, or else a *paired expression* $E \mapsto F$, where E and F are two expressions. Note that substituted predicates and expressions are given here in order to be able to define the inference rules of quantified predicates in section 4.5. In fact, such constructs will never be used by users of the Mathematical Language. Here is this new syntax:

$$
\begin{array}{lcl}
\mathit{predicate} & ::= & \bot \\
 & & \top \\
 & & \neg\, \mathit{predicate} \\
 & & \mathit{predicate} \land \mathit{predicate} \\
 & & \mathit{predicate} \lor \mathit{predicate} \\
 & & \mathit{predicate} \Rightarrow \mathit{predicate} \\
 & & \mathit{predicate} \Leftrightarrow \mathit{predicate} \\
 & & \forall \mathit{var\_list} \cdot \mathit{predicate} \\
 & & \underline{[\mathit{var\_list} := \mathit{exp\_list}]\, \mathit{predicate}} \\
 & & \\
\underline{\mathit{expression}} & ::= & \mathit{variable} \\
 & & [\mathit{var\_list} := \mathit{exp\_list}]\, \mathit{expression} \\
 & & \mathit{expression} \mapsto \mathit{expression} \\
 & & \\
\underline{\mathit{variable}} & ::= & \mathit{identifier}
\end{array}
$$

$SYNTAX3$

Note that we have not defined any syntactic structures for the two syntactic categories var_list and exp_list. They respectively denote finite sequences of variables and finite sequences of expressions.

4.1 Predicates and Expressions

It might be useful at this point to clearly identify the distinction between a predicate and an expression. A predicate P is a piece of formal text which can be *proved* when embedded within a sequent as in:

$$\vdash P$$

A predicate does not denote anything. This is not the case of an expression which always denote an *object*. An expression cannot be "proved". Hence predicate and expression are incompatible. Note that for the moment the possible expressions we can define are quite limited. This will be considerably extended in the Set-theoretic Language defined in section 5.

4.2 Substitutions and Quantified Predicates

A construct such as $x := E$, embedded into the predicate $[x := E]P$, where x is a list of variables made of distinct identifiers, E is a list of expressions of the same size as x, and P is a predicate, is called a *substitution*. The construct $[x := E]P$ denotes a transformation of the predicate P obtained by replacing in P the *free occurrences* of the variables of the list x by the corresponding expression of the list E. In section 4.3, we shall formally define what we mean by a free occurrence of a variable in a predicate or expression. And in section 4.4 we define rules to be applied in order to systematically perform such a transformation. Similar substitutions can be used in an expression.

A predicate such as $\forall x \cdot P$, where x is a list of variables made of distinct identifiers and P is a predicate, is called a *universally quantified predicate*. The predicate P is the *scope* of the variables of x in the quantified predicate $\forall x \cdot P$. The variables of x are said to be *bound* in P. Other variables having occurrences in P which are not bound are said to be *free*. Informally speaking for the moment, saying that such a universally quantified predicate is proved means that all predicates of the form $[x := E]P$ are then also proved. This will be formalized by two inference rules in section 4.5.

4.3 Free and Bound Variable Occurrences

The non-freeness of a list of variables in a formula can be calculated by means of a number of rules. These rules are defined on the structure of our Predicate Language. More precisely, the rules give meaning to the syntactic condition "l $\underline{\mathsf{nfin}}$ K", where l is a list of variables and K is a predicate or an expression. A construct such as "l $\underline{\mathsf{nfin}}$ K" is to be read "variables of the list l are not free in K". Note that being "not free" is the same as being "bound".

In the following table, x and y are meta-variables standing for a variable, l is a meta-variable standing for a list of variables, P and Q are meta-variables standing for a predicate, E and F are meta-variables standing for an expression, L is a meta-variable standing for a list of expressions, and K is a meta-variable standing for a predicate or an expression.

	Non-freeness	Condition
NF1	x $\underline{\mathsf{nfin}}$ y	x and y are distinct identifiers
NF2	(l, y) $\underline{\mathsf{nfin}}$ P	l $\underline{\mathsf{nfin}}$ P and y $\underline{\mathsf{nfin}}$ P

$NF1$

	Non-freeness	Condition	
NF3	$(l,\, y)$ $\underline{\mathsf{nfin}}$ E	l $\underline{\mathsf{nfin}}$ E and y $\underline{\mathsf{nfin}}$ E	
NF4	x $\underline{\mathsf{nfin}}$ $(P \wedge Q)$	x $\underline{\mathsf{nfin}}$ P and x $\underline{\mathsf{nfin}}$ Q	
NF5	x $\underline{\mathsf{nfin}}$ $(P \Rightarrow Q)$	x $\underline{\mathsf{nfin}}$ P and x $\underline{\mathsf{nfin}}$ Q	
NF6	x $\underline{\mathsf{nfin}}$ $\neg P$	x $\underline{\mathsf{nfin}}$ P	
NF7	x $\underline{\mathsf{nfin}}$ $\forall x \cdot P$		
NF8	x $\underline{\mathsf{nfin}}$ $\forall y \cdot P$	x $\underline{\mathsf{nfin}}$ y and x $\underline{\mathsf{nfin}}$ P	
NF9	x $\underline{\mathsf{nfin}}$ $(\forall l, y \cdot P)$	x $\underline{\mathsf{nfin}}$ $(\forall l \cdot \forall y \cdot P)$	$NF1$
NF10	x $\underline{\mathsf{nfin}}$ $[x := E]K$	x $\underline{\mathsf{nfin}}$ E	
NF11	x $\underline{\mathsf{nfin}}$ $[y := E]K$	x $\underline{\mathsf{nfin}}$ y and x $\underline{\mathsf{nfin}}$ E and x $\underline{\mathsf{nfin}}$ K	
NF12	x $\underline{\mathsf{nfin}}$ $[l, x := L, E]K$	x $\underline{\mathsf{nfin}}$$[l := L]K$ and x $\underline{\mathsf{nfin}}$ E	
NF13	x $\underline{\mathsf{nfin}}$ $[l, y := L, E]K$	x $\underline{\mathsf{nfin}}$ y and x $\underline{\mathsf{nfin}}$$[l := L]K$ and x $\underline{\mathsf{nfin}}$ E	
NF14	x $\underline{\mathsf{nfin}}$ (l, y)	x $\underline{\mathsf{nfin}}$ l and x $\underline{\mathsf{nfin}}$ y	
NF15	x $\underline{\mathsf{nfin}}$ (L, E)	x $\underline{\mathsf{nfin}}$ L and x $\underline{\mathsf{nfin}}$ E	
NF16	x $\underline{\mathsf{nfin}}$ $(E \mapsto F)$	x $\underline{\mathsf{nfin}}$ E and x $\underline{\mathsf{nfin}}$ F	

4.4 Substitution Rules

Substituted predicates or expressions can be calculated by means of the following rules defined on the structure of the Predicate Language. In this table, we use the same meta-variable conventions as in the previous table.

	Substitution	Definition
SUB1	$[x := E]\, x$	E
SUB2	$[x := E]\, y$	y if x $\underline{\mathsf{nfin}}$ y
SUB3	$[x := E]\,(P \wedge Q)$	$[x := E]\, P \ \wedge\ [x := E]\, Q$
SUB4	$[x := E]\,(P \Rightarrow Q)$	$[x := E]\, P \ \Rightarrow\ [x := E]\, Q$
SUB5	$[x := E]\, \neg\, P$	$\neg\, [x := E]\, P$
SUB6	$[x := E]\, \forall\, x \cdot P$	$\forall\, x \cdot P$
SUB7	$[x := E]\, \forall\, y \cdot P$	$\forall\, y \cdot [x := E]\, P$ if y $\underline{\mathsf{nfin}}$ x and y $\underline{\mathsf{nfin}}$ E
SUB8	$[x := E]\, \forall\, l, y \cdot P$	$[x := E]\, \forall\, l \cdot \forall\, y \cdot P$
SUB9	$[x := E]\,(F \mapsto G)$	$[x := E]\, F \ \mapsto\ [x := E]\, G$
SUB10	$[l, x := L, E]\, F$	$[z := E]\, [l := L]\, [x := z]\, F$ if z $\underline{\mathsf{nfin}}$ (l, x) and z $\underline{\mathsf{nfin}}$ (L, E, F)

$SUB1$

The application of these rules makes it possible to completely eliminate substitutions. Note that it is always possible to transform a quantified predicate such as $\forall x \cdot P$ into the following equivalent one $\forall y \cdot [x := y]P$ provided y is not free in P. This transformation is called a *change of variables*.

4.5 Universally Quantified Predicate Inference Rules

We now enlarge the initial theory $T1$ of section 3.2 with the following specific rules for universally quantified predicates:

	Antecedents	Consequent
R8	x $\underline{\mathsf{nfin}}$ Q for each Q in H $\mathsf{H} \vdash P$	$\mathsf{H} \vdash \forall x \cdot P$

$T2$

	Antecedents	Consequent
R9	$\mathsf{H} \vdash \forall x \cdot P$	$\mathsf{H} \vdash [x := E]\, P$

$T2$

Note that in Rule **R8**, the side condition might be false when the quantified variable x is free in the hypotheses H. One can easily cure this by doing a change of variable in the universal quantification $\forall x \cdot P$. More precisely, one can replace $\forall x \cdot P$ by $\forall y \cdot [x := y]\, P$, where y is a new variable that is not free in H. Such a change of variable is always possible.

4.6 Replacing the Previous Theory Extension

As for the case of the Proposition Language in section 3.3, we can replace the extension of previous section by this one which is more convenient to use for mechanizing the proof construction of our Predicate Language. Here is this extension of the practical Theory $S2$:

	Antecedents	Consequent
ALL1	if x $\underline{\mathsf{nfin}}$ R and x $\underline{\mathsf{nfin}}$ Q for each Q in H $\mathsf{H} \vdash \neg P \Rightarrow R$	$\mathsf{H} \vdash \neg (\forall x \cdot P) \Rightarrow R$
ALL2	$\mathsf{H} \vdash (\forall l, y \cdot P) \Rightarrow R$	$\mathsf{H} \vdash (\forall l \cdot \forall y \cdot P) \Rightarrow R$
INS	$\mathsf{H},\ \forall x \cdot P \vdash [x := E]\, P \Rightarrow \bot$	$\mathsf{H},\ \forall x \cdot P \vdash \bot$

$S3$

This theory has to be used with the following tactic, where RULES is one of **AXM**, **IMP1**, **IMP2**, **AND1**, **AND2**, **NEG**, **OR1**, **OR2**, **ALL1**, and **ALL2**:

INI ; ((RULES* ; **DED**)*;**INS**)*

Again when the side condition of rule **ALL1** is false, one can do a change of variable as explained at the end of the previous section.

4.7 Extending the Predicate Language: Existential Quantification

The Predicate Language is now extended by introducing *existential quantification* of predicates. Given a predicate P and a list of variables x, made of distinct identifiers, we can construct the predicate $\exists x \cdot P$. The new syntax is thus now as follows:

$$
\begin{array}{lll}
predicate & ::= & \bot \\
 & & \top \\
 & & \neg\, predicate \\
 & & predicate \wedge predicate \\
 & & predicate \vee predicate \\
 & & predicate \Rightarrow predicate \\
 & & predicate \Leftrightarrow predicate \\
 & & \forall var\_list \cdot predicate \\
 & & \exists var\_list \cdot predicate \\
 & & \overline{[var\_list := exp\_list]}\, predicate \\
 & & \\
expression & ::= & variable \\
 & & [var\_list := exp\_list]\, expression \\
 & & expression \mapsto expression \\
 & & \\
variable & ::= & identifier
\end{array}
\qquad SYNTAX4
$$

This extension is defined as follows by a rewriting rule in terms of the universally quantified predicate:

Predicate	Rewritten
$\exists x \cdot P$	$\neg \forall x \cdot \neg P$

The previous practical Theory $S3$ can then be extended as follows after applying the previous rewriting rule:

	Antecedents	Consequent
XST1	H $\vdash$ $(\forall x \cdot \neg P) \Rightarrow R$	H $\vdash$ $\neg (\exists x \cdot P) \Rightarrow R$
XST2	if x nfin R and x nfin H H $\vdash$ $P \Rightarrow R$	H $\vdash$ $(\exists x \cdot P) \Rightarrow R$

$S4$

Again a change of variable can be done when the side condition of rule **XST2** is false.

4.8 Extending the Predicate Language: Equality

The Predicate Language is once again extended by adding a new predicate, the *equality predicate*. Given two expressions E and F, we define the following construct $E = F$. Here is the extension of our syntax:

$$
\begin{array}{lll}
predicate & ::= & \bot \\
 & & \top \\
 & & \neg\, predicate \\
 & & predicate \ \wedge \ predicate \\
 & & predicate \ \vee \ predicate \\
 & & predicate \ \Rightarrow \ predicate \\
 & & predicate \ \Leftrightarrow \ predicate \\
 & & \forall var\_list \cdot predicate \\
 & & \exists var\_list \cdot predicate \\
 & & [var\_list := exp\_list]\, predicate \\
 & & \underline{expression = expression} \\
 & & \\
expression & ::= & variable \\
 & & [var\_list := exp\_list]\, expression \\
 & & expression \mapsto expression \\
 & & \\
variable & ::= & identifier
\end{array}
$$

$SYNTAX5$

We extend in a similar manner the rules of non-freeness as follows:

	Non-freeness	Condition
NF17	x nfin $(E = F)$	x nfin E and x nfin F

$NF2$

We also extend the substitution rules as follows:

	Substitution	Definition
SUB11	$[\, x := C \,]\,(D = E)$	$[x := C]D \ = \ [x := C]E$

$SUB2$

The natural Theory $T2$ is extended with two inference rules for proving equality predicates:

	Antecedents	Consequent
R10	H $\vdash$ $E = F$ H $\vdash$ $[x := E]P$	H $\vdash$ $[x := F]P$
R11		H $\vdash$ $E = E$

$T3$

Finally, we extend accordingly our practical Theory $S4$:

	Antecedents	Consequent
EQL1		$\mathsf{H} \vdash \neg (E = E) \Rightarrow R$
LBZ1	$\mathsf{H},\ E = F,\ [x := E]P \vdash [x := F]P \Rightarrow \bot$	$\mathsf{H},\ E = F,\ [x := E]P \vdash \bot$
LBZ2	$\mathsf{H},\ E = F,\ [x := F]P \vdash [x := E]P \Rightarrow \bot$	$\mathsf{H},\ E = F,\ [x := F]P \vdash \bot$

$S5$

5 The Set-theoretic Language

Our next language, the Set-theoretic Language is now presented as an extension to the previous Predicate Language.

5.1 Syntax

We introduce another predicate the *membership predicate* and a new syntactic category: *set*. Given an expression E and a set s, the construct $E \in s$ is a membership predicate. We also enlarge the expression syntactic category by adding that an expression can be a set. Finally, we introduce the set constructs. Given two sets s and t, the construct $s \times t$ is a set called the *Cartesian product* of s and t. Given a set s, the construct $\mathbb{P}(s)$ is a set called the *power set of* s. Finally, given a list of variables x with distinct identifiers, a predicate P, and an expression E, the construct $\{x \cdot P \mid E\}$ is called a *set defined in comprehension*. Here is our new syntax:

$$
\begin{array}{lll}
predicate & ::= & \bot \\
 & & \top \\
 & & \neg\, predicate \\
 & & predicate \ \wedge \ predicate \\
 & & predicate \ \vee \ predicate \\
 & & predicate \ \Rightarrow \ predicate \\
 & & predicate \ \Leftrightarrow \ predicate \\
 & & \forall\, var\_list \cdot predicate \\
 & & \exists\, var\_list \cdot predicate \\
 & & [var\_list := exp\_list]\, predicate \\
 & & expression \ = \ expression \\
 & & \underline{expression \ \in \ set} \\
 & & \\
expression & ::= & variable \\
 & & [var\_list := exp\_list]\, expression \\
 & & expression \mapsto expression \\
 & & \underline{set}
\end{array}
$$

$SYNTAX6$

$$
\begin{array}{lll}
variable & ::= & identifier \\
\\
\underline{set} & ::= & set \times set \\
 & & \mathbb{P}(set) \\
 & & \{\, var\_list \cdot predicate \mid expression \,\} \\
 & & variable
\end{array}
$$

$SYNTAX6$

5.2 Non-freeness and Substitution Rules

We first enlarge the non-freeness set of rules by adding the following ones:

	Non-freeness	Condition
NF18	x nfin $(s \times t)$	x nfin s and x nfin t
NF19	x nfin $\mathbb{P}(s)$	x nfin s
NF20	x nfin $\{l \cdot P \mid E\}$	x nfin $(\forall l \cdot P \Rightarrow E = E)$

$NF3$

Likewise, we add the following rules to the substitution ones:

	Substitution	Definition
SUB12	$[\, x := E \,]\,(s \times t)$	$[\, x := E \,]s \;\times\; [\, x := E \,]t$
SUB13	$[\, x := E \,]\,\mathbb{P}(s)$	$\mathbb{P}([\, x := E \,]s)$
SUB14	$[\, x := E \,]\,\{x \cdot P \mid F\}$	$\{\, x \cdot P \mid F \,\}$
SUB14	$[\, x := E \,]\,\{y \cdot P \mid F\}$	$\{\, y \cdot [\, x := E \,]P \mid [\, x := E \,]F \,\}$ if y nfin x and y nfin E
SUB15	$[\, x := E \,]\,\{l, x \cdot P \mid F\}$	$\{l, x \cdot P \mid F\}$
SUB15	$[\, x := E \,]\,\{l, y \cdot P \mid F\}$	$\{\, l, y \cdot [x := E]P \mid [\, x := E \,]\, F \,\}$ if x nfin y and l nfin x and l, y nfin E

$SUB3$

5.3 Inference Rules

The inference rules for the set-theoretic Language are given under the form of equivalences to various set memberships. They are all defined in terms of rewriting rules. Note that the first of this rule defines equality for sets. It is called the extensionality axiom.

Operator	Predicate	Rewritten
Set equality	$s = t$	$s \in \mathbb{P}(t) \ \wedge \ t \in \mathbb{P}(s)$
Cartesian product	$E \mapsto F \in s \times t$	$E \in s \ \wedge \ F \in t$
Power set	$E \in \mathbb{P}(s)$	$\forall x \cdot (\, x \in E \ \Rightarrow \ x \in s \,)$ if x $\underline{\mathsf{nfin}}$ E and x $\underline{\mathsf{nfin}}$ s
Set comprehension	$E \in \{ \, x \cdot P \mid F \, \}$	$\exists x \cdot (\, P \ \wedge \ E = F \,)$ if x $\underline{\mathsf{nfin}}$ E

$SET1$

As a special case, set comprehension can sometimes be written $\{ \, F \mid P \, \}$, which can be read as follows: "the set of objects with shape F when P holds". However, as you can see, the list of variables x has now disappeared. In fact, these variables are then *implicitly determined* as being all the free variables in F. When we want that x represent only *some*, but not all, of these free variables we cannot use this shorthand.

A more special case is one where the expression F is exactly x, that is $\{ \, x \cdot P \ \mid \ x \, \}$. As a shorthand, this can be written $\{ \, x \mid P \, \}$, which is very common in informally written mathematics. But again, notice that, contrarily to intuition, the list of variables x has disappeared. Again, it will be determined as the free variables of expression x. And then $E \in \{ \, x \mid P \, \}$ becomes $[x := E]P$.

From now on, all extensions of the Set-theoretic Language will take the form of "simple facilities", as explained in section 3.5. And most of them are extensions of the *set* syntactic category. As a consequence, the new syntax will be presented differentially. The new constructs will be presented under the form of rewriting rules. And since most of the new construct are sets, the rewriting rules will transform some set membership predicates into simpler ones.

5.4 Elementary Set Operators

In this section, we introduce the classical set operators: inclusion, union, intersection, difference, extension, and the empty set.

$$
\begin{array}{lll}
predicate & ::= & \ldots \\
 & & set \subseteq set \\
\ldots & & \\
set & ::= & \ldots \\
 & & set \cup set \\
 & & set \cap set \\
 & & set \setminus set \\
 & & \{exp\_list\} \\
 & & \varnothing
\end{array}
$$

$SYNTAX7$

Operator	Predicate	Rewritten
Inclusion	$S \subseteq T$	$S \in \mathbb{P}(T)$
Union	$E \in S \cup T$	$E \in S \ \lor \ E \in T$
Intersection	$E \in S \cap T$	$E \in S \ \land \ E \in T$
Difference	$E \in S \setminus T$	$E \in S \ \land \ \neg (E \in T)$
Set extension	$E \in \{a, \ldots, b\}$	$E = a \ \lor \ \ldots \ \lor \ E = b$
Empty set	$E \in \varnothing$	$\bot$

$SET2$

5.5 Generalization of Elementary Set Operators

The next series of operators consists in generalizing union and intersection to sets of sets. This takes the forms either of an operator acting on a set or of a quantifier.

$$
\begin{array}{lll}
\ldots & & \\
set & ::= & \ldots \\
 & & \mathrm{union}(set) \\
 & & \bigcup var\_list \cdot (predicate \mid set) \\
 & & \mathrm{inter}(set) \\
 & & \bigcap var\_list \cdot (predicate \mid set)
\end{array}
$$

$SYNTAX8$

Operator	Predicate	Rewritten
Generalized intersection	$E \in \mathsf{union}\,(S)$	$\exists s \cdot (\, s \in S \ \wedge \ E \in s \,)$ if s nfin S and s nfin E
Quantified union	$E \in \bigcup\, x \cdot (\, P \mid T \,)$	$\exists x \cdot (\, P \ \wedge \ E \in T \,)$ if x nfin E
Generalized intersection	$E \in \mathrm{inter}\,(S)$	$\forall s \cdot (\, s \in S \ \Rightarrow \ E \in s \,)$ if s nfin S and s nfin E
Quantified intersection	$E \in \bigcap\, x \cdot (\, P \mid T \,)$	$\forall x \cdot (\, P \ \Rightarrow \ E \in T \,)$ if x nfin E

$SET3$

The last two rewriting rules require that the set $\mathrm{inter}(S)$ and $\bigcap\ x \cdot (\, P \mid T \,)$ be *well defined*. This is defined in the following table:

Set construction	Well-definedness condition
$\mathrm{inter}\,(S)$	$S \neq \varnothing$
$\bigcap\, x \cdot (\, P \mid T \,)$	$\{x \cdot \ P \mid T \,\} \neq \varnothing$

$WF1$

5.6 Binary Relation Operators

We now define a first series of binary relation operators: the set of binary relations built on two sets, the domain and range of a binary relation, and then various sets of binary relations.

$$
\begin{array}{l}
\ldots \\
\\
set \ ::= \ \ldots \\
\qquad\quad set \leftrightarrow set \\
\qquad\quad \mathrm{dom}(set) \\
\qquad\quad \mathrm{ran}(set) \\
\qquad\quad set \nleftrightarrow set \\
\qquad\quad set \leftrightarrow\!\!\!\!\!\rightarrow set \\
\qquad\quad set \nleftrightarrow\!\!\!\!\!\rightarrow set
\end{array}
$$

$SYNTAX9$

Operator	Predicate	Rewritten
Set of all binary relations	$r \in S \leftrightarrow T$	$r \subseteq S \times T$
Domain	$E \in \mathrm{dom}\,(r)$	$\exists y \cdot (E \mapsto y \ \in \ r)$ if y nfin E and y nfin r
Range	$F \in \mathrm{ran}\,(r)$	$\exists x \cdot (x \mapsto F \ \in \ r)$ if x nfin E and x nfin r
Set of all total relations	$r \in S \nleftrightarrow T$	$r \in S \leftrightarrow T \ \wedge \ \mathrm{dom}\,(r) = S$
Set of all surjective relations	$r \in S \leftrightarrow\!\!\!\!\!\rightarrow T$	$r \in S \leftrightarrow T \ \wedge \ \mathrm{ran}\,(r) = T$
Set of all total and surjective relations	$r \in S \nleftrightarrow\!\!\!\!\!\rightarrow T$	$r \in S \nleftrightarrow T \ \wedge \ r \in S \leftrightarrow\!\!\!\!\!\rightarrow T$

$SET4$

The next series of binary relations operators define the converse of a relation, various relation restrictions and the image of a set under a relation.

$$
\begin{array}{l}
\ldots \\
\\
set \ ::= \ \ldots \\
\qquad\quad set^{-1} \\
\qquad\quad set \lhd set \\
\qquad\quad set \rhd set \\
\qquad\quad set \lhd\!\!\!\!- set \\
\qquad\quad set \rhd\!\!\!\!- set \\
\qquad\quad set[set]
\end{array}
$$

$SYNTAX10$

Operator	Predicate	Rewritten
Converse	$E \mapsto F \in r^{-1}$	$F \mapsto E \in r$
Domain restriction	$E \mapsto F \ \in \ S \lhd r$	$E \in S \ \wedge \ E \mapsto F \in r$
Range restriction	$E \mapsto F \ \in \ r \rhd T$	$E \mapsto F \in r \ \wedge \ F \in T$

$SET5$

Operator	Predicate	Rewritten
Domain subtraction	$E \mapsto F \ \in \ S \lhd r$	$E \notin S \ \ \wedge \ \ E \mapsto F \in r$
Range subtraction	$E \mapsto F \ \in \ r \rhd T$	$E \mapsto F \in r \ \ \wedge \ \ F \notin T$
Relational Image	$F \in r[w]$	$\exists x \cdot (\, x \in w \ \ \wedge \ \ x \mapsto F \in r \,)$ if x nfin F and x nfin r and x nfin w

$SET5$

Our next series of operators defines the composition of two binary relations, the overriding of a relation by another one, and the direct and parallel products of two relations.

$$
\begin{array}{l}
\ldots \\
\\
set \ ::= \ \ldots \\
\qquad\quad set \, ; \, set \\
\qquad\quad set \circ set \\
\qquad\quad set \lhd\!\!\!\!- \ set \\
\qquad\quad set \otimes set \\
\qquad\quad set \parallel set
\end{array}
$$

$SYNTAX11$

Operator	Predicate	Rewritten
Forward composition	$E \mapsto F \in p \, ; \, q$	$\exists x \cdot (\, E \mapsto x \in p \ \ \wedge \ \ x \mapsto F \in q \,)$ if x nfin E and x nfin F and x nfin p and x nfin q
Backward composition	$E \mapsto F \in q \circ p$	$E \mapsto F \in p \, ; \, q$
Overriding	$E \mapsto F \in p \lhd\!\!\!\!- \ q$	$E \mapsto F \in (\mathrm{dom} \, (q) \lhd p) \ \cup \ q$
Direct product	$E \mapsto (F \mapsto G) \in p \otimes q$	$E \mapsto F \in p \ \ \wedge \ \ E \mapsto G \in q$
Parallel product	$(E \mapsto F) \mapsto (G \mapsto H) \in p \parallel q$	$E \mapsto G \in p \ \ \wedge \ \ F \mapsto H \in q$

$SET6$

5.7 Functional Operators

In this section we define various function operators: the sets of all partial and total functions, partial and total injections, partial and total surjections, and bijections. We also introduce the two projection functions as well as the identity function.

$$
\begin{array}{l}
\ldots \\
\\
set \ ::= \ \ldots \\
\qquad\quad set \pfun set \\
\qquad\quad set \fun set \\
\qquad\quad set \pinj set \\
\qquad\quad set \inj set \\
\qquad\quad set \psur set \\
\qquad\quad set \sur set \\
\qquad\quad set \bij set \\
\qquad\quad \mathrm{prj}_1(set) \\
\qquad\quad \mathrm{prj}_2(set) \\
\qquad\quad \mathrm{id}(set)
\end{array}
\qquad SYNTAX12
$$

Operator	Predicate	Rewritten
Set of all partial functions	$f \in S \pfun T$	$f \in S \rel T \ \land \ (f^{-1} ; f) = \mathrm{id}\,(\mathrm{ran}\,(f))$
Set of all total functions	$f \in S \fun T$	$f \in S \pfun T \ \land \ S = \mathrm{dom}\,(f)$
Set of all partial injections	$f \in S \pinj T$	$f \in S \pfun T \ \land \ f^{-1} \in T \pfun S$
Set of all total injections	$f \in S \inj T$	$f \in S \fun T \ \land \ f^{-1} \in T \pfun S$
Set of all partial surjections	$f \in S \psur T$	$f \in S \pfun T \ \land \ T = \mathrm{ran}\,(f)$
Set of all total surjections	$f \in S \sur T$	$f \in S \fun T \ \land \ T = \mathrm{ran}\,(f)$
Set of all bijections	$f \in S \bij T$	$f \in S \inj T \ \land \ f \in S \sur T$
First projection	$(E \mapsto F) \mapsto G \in \mathsf{prj_1}(r)$	$E \mapsto F \in r \ \land \ G = E$

$SET7$

Operator	Predicate	Rewritten
Second projection	$(E \mapsto F) \mapsto G \in \mathsf{prj}_2(r)$	$E \mapsto F \in r \ \ \wedge \ \ G = F$
Identity	$E \mapsto F \in \mathrm{id}\,(S)$	$E \in S \ \ \wedge \ \ F = E$

$SET7$

5.8 Lambda Abstraction and Function Invocation

We now define *lambda abstraction*, which is a way to construct functions, and also function invocation, which is a way to call functions. But first we have to define the notion of *pattern of variables*. A pattern of variables is either an identifier or a pair made of two patterns of variables. Moreover, all variables composing the pattern must be distinct. For example, here are three patterns of variables:

$$\mathsf{abc}$$

$$\mathsf{abc} \mapsto \mathsf{def}$$

$$\mathsf{abc} \mapsto (\mathsf{def} \mapsto \mathsf{ghi})$$

Given a pattern of variables x, a predicate P, and an expression E, the construct $\lambda\, x \cdot (P \mid E)$ is a lambda abstraction, which is a function. Given a function f and an expression E, the construct $f(E)$ is an expression denoting a function invocation. Here is our new syntax

$$
\begin{array}{lll}
\ldots & & \\
\mathit{expression} & ::= & \ldots \\
 & & \mathit{set}(\mathit{expression}) \\
\mathit{set} & ::= & \ldots \\
 & & \lambda\, \mathit{pattern} \cdot \mathit{predicate} \mid \mathit{expression} \\
\mathit{pattern} & ::= & \mathit{variable} \\
 & & \mathit{pattern} \mapsto \mathit{pattern}
\end{array}
$$

$SYNTAX13$

In the following table, l stands for the list of variables in the pattern L.

Operator	Predicate	Rewritten
Lambda abstraction	$F \mapsto G \ \in \ \lambda L \cdot P \mid E$	$F \mapsto G \ \in \ \{L \cdot P \mid L \mapsto E\}$
Function invocation	$F = f(E)$	$E \mapsto F \ \in \ f$

$SET8$

The function invocation construct $f(E)$ requires a well-formedness condition, which is the following:

Expression	Well-formedness condition
$f(E)$	$E \in \mathrm{dom}(f)$

$WF2$

6 Arithmetic Language

6.1 Syntax

We add a new syntactic category: $number$. An expression might be a number. Numbers are either 0, the sum, product, or power of two numbers. We also add the sets $\mathbb{N}$ and succ.

$$
\begin{array}{lll}
\ldots & & \\
expression & ::= & \ldots \\
 & & number \\
set & ::= & \ldots \\
 & & \mathbb{N} \\
 & & \text{succ} \\
number & ::= & 0 \\
 & & number + number \\
 & & number * number \\
 & & number \mathbin{\widehat{}} number
\end{array}
$$

$SYNTAX14$

6.2 Peano Axioms and Recursive Definitions

The following predicates are added systematically to the hypotheses of sequents to prove:

$$
\begin{array}{l}
0 \in \mathbb{N} \\
\\
\text{succ} \in \mathbb{N} \rightarrowtail\!\!\!\!\!\rightarrow \mathbb{N} \setminus \{0\} \\
\\
\forall S \cdot \left(\begin{array}{l} S \subseteq \mathbb{N} \\ 0 \in S \\ \forall n \cdot (n \in S \Rightarrow \text{succ}(n) \in S) \\ \Rightarrow \\ \mathbb{N} \subseteq S \end{array} \right)
\end{array}
$$

$ARITH1$

$$\forall a \cdot (\, a \in \mathbb{N} \;\Rightarrow\; a + 0 \;=\; a \,)$$

$$\forall a \cdot (\, a \in \mathbb{N} \;\Rightarrow\; a * 0 \;=\; 0 \,)$$

$$\forall a \cdot (\, a \in \mathbb{N} \;\Rightarrow\; a \hat{} 0 \;=\; \text{succ}(0) \,)$$

$$\forall a, b \cdot (\, a \in \mathbb{N} \;\wedge\; b \in \mathbb{N} \;\Rightarrow\; a + \text{succ}(b) \;=\; \text{succ}(a + b) \,)$$

$$\forall a, b \cdot (\, a \in \mathbb{N} \;\wedge\; b \in \mathbb{N} \;\Rightarrow\; a * \text{succ}(b) \;=\; (a * b) + a \,)$$

$$\forall a, b \cdot (\, a \in \mathbb{N} \;\wedge\; b \in \mathbb{N} \;\Rightarrow\; a \hat{} \text{succ}(b) \;=\; (a \hat{} b) * a \,)$$

$ARITH1$

6.3 Extension of the Arithmetic Language

We introduce the classical binary relations on numbers, the finiteness predicate, the interval between two numbers, and the subtraction, division, and modulo constructs.

$$\begin{array}{lll}
\ldots & & \\
predicate & ::= & \ldots \\
 & & number \leq number \\
 & & number < number \\
 & & \text{finite}(set) \\
set & ::= & \ldots \\
 & & number \,..\, number \\
number & ::= & \ldots \\
 & & number - number \\
 & & number \,/\, number \\
 & & number \;\text{mod}\; number \\
 & & \text{card}(set)
\end{array}$$

$SYNTAX15$

Operator	Predicate	Rewritten
smaller than or equal	$a \leq b$	$\exists c \cdot (\, c \in \mathbb{N} \;\wedge\; b = a + c \,)$
smaller than	$a < b$	$a \leq b \;\wedge\; a \neq b$
interval	$c \in a \,..\, b$	$a \leq c \;\wedge\; c \leq b$

$ARITH2$

Operator	Predicate	Rewritten
subtraction	$c = a - b$	$a = b + c$
division	$c = a/b$	$\exists r \cdot (r \in \mathbb{N} \ \land \ r < b \ \land \ a = c * b + r)$
modulo	$r = a \ \mathrm{mod} \ b$	$a = (a/b) * b + r$
finiteness	$\mathrm{finite}(s)$	$\exists n, f \cdot (n \in \mathbb{N} \ \land \ f \in 1 \mathinner{..} n \rightarrowtail\!\!\!\!\!\!\rightarrow s)$
cardinality	$n = \mathrm{card}(s)$	$\exists f \cdot f \in 1 \mathinner{..} n \rightarrowtail\!\!\!\!\!\!\rightarrow s$

$ARITH2$

The subtraction, division, modulo, and cardinal constructs are subjected to some well-formedness conditions, which are the following:

Number	Well-definedness condition
$a - b$	$b \leq a$
a/b	$b \neq 0$
$a \ \mathrm{mod} \ b$	$b \neq 0$
$\mathrm{card}(s)$	$\mathrm{finite}(s)$

$WF3$