Notes for the Proof Theory Course

Master 1 "Informatique", Univ. Paris 13

Damiano Mazza

Contents

Preliminaries

In the following we will make a heavy use of inductive definitions, in which one introduces a set of objects by starting from a set of basic objects and giving rules for building new objects from already defined objects.

For example, if one wants to define binary trees, one may proceed by saying that:

- the empty tree, denoted by $\circ$, is a binary tree;
- if t and t' are binary trees, then (t, t') is also a binary tree;
- nothing else is a binary tree.

A typical way of presenting such definitions more compactly is the *formal grammar* notation. For instance, in the case of binary trees, we write

$$t, t' ::= \circ \mid (t, t').$$

Another typical form of inductive definition is a *derivation system*, in which one manipulates expressions generically called *judgments*, let us denote them by $J, K \ldots$, and derives judgments from previously derived ones by means of rules, presented as follows:

$$\frac{J_1 \quad J_2 \quad J_3 \quad \ldots}{K}$$

$J_1, J_2, J_3 \ldots$ are called the *premises* of the rule; their number is usually finite and is called the *arity* of the rule. K is called the *conclusion* of the rule. A derivation system usually comprises one or more nullary rules, *i.e.*, rules of arity zero, corresponding to the basic judgments.

For instance, the above definition of binary tree may be presented as a derivation system manipulating judgments of the form t tree, intuitively meaning "t is a binary tree", with the following rules:

$$\frac{}{\circ \ \mathsf{tree}} \qquad\qquad \frac{t \ \mathsf{tree} \qquad t' \ \mathsf{tree}}{(t, t') \ \mathsf{tree}}$$

A *derivation* in a derivation system is a tree obtained by successively applying rules. For example, the following derivation says that, if t is a binary tree, then $((\circ, \circ), t)$ is also a binary tree:

$$\dfrac{\dfrac{\dfrac{}{\circ \ \mathsf{tree}} \quad \dfrac{}{\circ \ \mathsf{tree}}}{(\circ, \circ) \ \mathsf{tree}} \qquad t \ \mathsf{tree}}{((\circ, \circ), t) \ \mathsf{tree}}$$

The judgment at the root of the derivation tree is the *conclusion* of the derivation; the judgments labeling the leaves are the *hypotheses* of the derivation. For instance, the conclusion of the above derivation is $((\circ, \circ), t)\ \mathsf{tree}$ and its only hypothesis is $t\ \mathsf{tree}$. A derivation δ of hypotheses $J_1, \ldots, J_n$ and conclusion K is generically written

$$\begin{array}{cccc} J_1 & \ldots & J_n \\ & \vdots\ \delta & \\ & K & \end{array}$$

A *proof* is a derivation with no hypotheses, *i.e.*, the leaves are all conclusions of nullary rules. A judgment is *provable* if it is the conclusion of some proof. For instance, a judgment $t\ \mathsf{tree}$ is provable in the above system iff t is indeed a binary tree.

Chapter 1

Propositional Classical Logic

1.1 Formulas and truth semantics

Definition 1.1 (propositional formula) Propositional formulas *are defined by the following grammar:*

$$\begin{array}{rll} A, B ::= & P, Q, R \ldots & \textit{(propositional constants)} \\ & \mid \neg A & \textit{(negation)} \\ & \mid \top \mid A \wedge B & \textit{(truth and conjunction)} \\ & \mid \bot \mid A \vee B & \textit{(falsehood and disjunction)} \end{array}$$

Remark 1.1 *For simplicity, in the context of classical logic we choose not to include implication as a primitive connective but we define it as*

$$A \Rightarrow B \quad := \quad \neg A \vee B.$$

Logical equivalence (the "if and only if" connective) is a double implication and is defined by

$$A \Leftrightarrow B \quad := \quad (A \Rightarrow B) \wedge (B \Rightarrow A).$$

When we will define intuitionistic logic we will re-introduce implication as a primitive connective, at which point we will discard negation and define it as $\neg A := A \Rightarrow \bot$.

Intuitively, a formula in classical logic is something that may be either true or false. Whether it is true or false depends on the truth value we assign to its constants, as specified in the following:

Definition 1.2 (valuation, model) *A* valuation *is a function* v *from propositional constants to* $\{0, 1\}$. *The relations* $v \models A$ (v is a model of A, *meaning that* A *is true under the valuation* v*) and* $v \not\models A$ (v is a countermodel of A, *meaning that* A *is false under the valuation* v*) are defined by means of the derivation system shown in Fig. 1.1.*

$$\frac{}{v \models P}\ {\scriptstyle v(P)=1} \qquad \frac{v \not\models A}{v \models \neg A} \qquad \frac{}{v \not\models P}\ {\scriptstyle v(P)=0} \qquad \frac{v \models A}{v \not\models \neg A}$$

$$\frac{}{v \models \top} \qquad \frac{v \models A}{v \models A \wedge B} \qquad \frac{v \models A}{v \models A \vee B} \qquad \frac{v \models B}{v \models A \vee B}$$

$$\frac{}{v \not\models \bot} \qquad \frac{v \not\models A \quad v \not\models B}{v \not\models A \vee B} \qquad \frac{v \not\models A}{v \not\models A \wedge B} \qquad \frac{v \not\models B}{v \not\models A \wedge B}$$

Figure 1.1: The truth semantics of propositional classical logic.

Proposition 1.1 *For every valuation v and formula A, $v \not\models A$ iff it is not the case that $v \models A$.* □

Some standard terminology:

- a formula is *valid* (also called a *tautology*) if $v \models A$ for every valuation v; in that case, we write $\models A$;
- a formula is *satisfiable* if $v \models A$ for some valuation v;
- a formula is *unsatisfiable* (also called a *contradiction*) if it has no models; in that case, we write $\not\models A$.

Obviously, if A is a tautology (resp. a contradiction) then $\neg A$ is a contradiction (resp. a tautology).

Example 1.1 *Some examples of tautologies:*

1. $\top$ *(truth, the simplest tautology);*
2. $A \vee \neg A$ *(excluded middle);*
3. $(\neg A \Rightarrow \bot) \Rightarrow A$ *(reduction to absurdity);*
4. $(\neg A \Rightarrow A) \Rightarrow A$ *(strong reduction to absurdity);*
5. $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$ *(Peirce's law);*
6. $\neg\neg A \Leftrightarrow A$ *(involutivity of negation);*
7. $\neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B$ *(one of De Morgan's laws);*
8. $\neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B$ *(the other De Morgan's law);*
9. $A \Rightarrow B \Rightarrow A$ *(first intuitionistic axiom);*
10. $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$ *(second intuitionistic axiom).*

Example 1.2 *Some examples of satisfiable formulas:*

1. $A \Rightarrow \neg A$*;*
2. $(A \Rightarrow \neg A) \Rightarrow A$*.*

Remark 1.2 *Note that, although a valuation is defined as a function from* all *propositional constants to* $\{0,1\}$*, to define* $v \models A$ *it only suffices for* v *to be defined on those constants appearing in* A*. Therefore, a valuation is in practice an object of essentially the same size as* A*.*

Indeed, checking whether a formula is satisfiable is the typical NP *problem: given a formula* A *of size*[1] n*, a certificate of its satisfiability is given by a valuation* v *s.t.* $|A|_v = 1$*; now, the size of* v *is* $O(n)$ *and, moreover, computing* $|A|_v$ *may be done in* $O(n)$ *time. In fact, satisfiability is* NP*-complete, i.e., it is "the hardest" problem in* NP*. Dually, checking whether a formula is a tautology is in* coNP*, because the negation of a tautology is a formula admitting a countermodel (again, the problem is actually* coNP*-complete).*

In the following chapters, we will study proof systems which are sound and complete with respect to valid formulas, *i.e.*, a formula is provable in the system iff it is valid. Combined with the above remark, this gives a proof-theoretic perspective on the NP vs. coNP problem, as follows. First of all, we observe that any proof system must provide finite proofs, otherwise it would be impossible to check their validity. Therefore, proofs may be encoded as finite strings on some finite alphabet Σ (indeed, we may always take $\Sigma = \{0,1\}$). In this context, we call a finite string on Σ a *putative proof*.

Proposition 1.2 $\mathsf{NP} = \mathsf{coNP}$ *iff there exists a sound and complete proof system* $\mathcal{S}$ *for classical propositional logic and two polynomials* p, q *such that:*

1. *for every putative proof* π *of size* n*, it is possible to check whether* π *is indeed a proof of* $\mathcal{S}$ *in time* $O(p(n))$*;*
2. *for every tautology* A *of size* n*, there exists a proof of* A *in* $\mathcal{S}$ *of size* $O(q(n))$*.*

Proof. Suppose $\mathsf{NP} = \mathsf{coNP}$. In particular, the tautology problem is in NP, which means that there exist two polynomials p', q' such that each tautology A of size n has a certificate c_A (a binary string) of size $O(q'(n))$ and which is verifiable in time $O(p'(n))$. Define $\mathcal{S}$ to be the proof system consisting of exactly one proof for each tautology A, of the form (A, c_A). $\mathcal{S}$ is trivially sound and complete and obviously satisfies property 1 (take $p := p'$) and property 2 (take $q := q'$).

Suppose that a sound and complete proof system $\mathcal{S}$ satisfying properties 1 and 2 exists. Then, the tautology problem is in NP: given a tautology A, a proof of A in $\mathcal{S}$ (which exists because $\mathcal{S}$ is complete) is, by definition, a succinct certificate (property 2) verifiable in polynomial time (property 1) of the fact that A is a tautology (because $\mathcal{S}$ is sound). Since the tautology problem is coNP-complete, $\mathsf{NP} = \mathsf{coNP}$ follows. □

The above observation spurred the birth of *proof complexity*, a subfield of structural proof theory focusing on the size of proofs of families of tautologies in various proof systems. In particular, since it is widely believed that $\mathsf{NP} \neq \mathsf{coNP}$, the effort has been directed towards finding "hard" families of tautologies for

[1] In the context of satisfiability, the size of A is usually taken to be the number of constants appearing in A. However, the number of total symbols appearing in A is an equally suitable notion of size.

each given sound and complete proof system, *i.e.*, families of formulas $(A_n)_{n\in\mathbb{N}}$ such that the size of A_n is $\Theta(n)$ but the size of the shortest proof of A_n in the system is asymptotically super-polynomial in n.

1.2 Atomic negation

In classical logic, it is convenient to dispense also with negation as a primitive connective. This is possible by observing the following equivalences (which are tautologies, see Example 1.1, points 6, 7 and 8):

$$\neg\neg A \Leftrightarrow A \qquad \neg(A \wedge B) \Leftrightarrow \neg A \vee \neg B \qquad \neg(A \vee B) \Leftrightarrow \neg A \wedge \neg B.$$

The latter two, called De Morgan's laws, allow us to "push" negation inside formulas, up to the atoms. The first one, which states that negation is an involution, allows us to remove superfluous negations, replacing an odd number of negations with just one negation and an even number of negations with no negation at all. This suggests the following definition.

Definition 1.3 (propositional formula, atomic negation) *We define* propositional formulas atomic negation *by means of the following grammar:*

$$A, B ::= P, Q, R \ldots \mid \overline{P}, \overline{Q}, \overline{R} \ldots \mid \top \mid A \wedge B \mid \bot \mid A \vee B.$$

The negation *of a formula A, denoted by $\neg A$, is defined by induction as follows:*

$$\begin{array}{cc} \neg P := \overline{P} & \neg\overline{P} := P \\ \neg\top := \bot & \neg\bot := \top \\ \neg(A \wedge B) := \neg A \vee \neg B & \neg(A \vee B) := \neg A \wedge \neg B \end{array}$$

Remark 1.3 *It is important to understand the difference between Definition 1.1 and Definition 1.3. In Definition 1.1, negation is a primitive connective, it is a constructor building a formula $\neg A$ from a previously defined formula A. One consequence is that, if we define the size of a formula to be the number of nodes in its syntactic tree, and if the size of A is n, then the size of $\neg A$ is $n + 1$. In particular, A and $\neg\neg A$ are* different *formulas according to Definition 1.1, otherwise it would not even make sense to state the equivalence $\neg\neg A \Leftrightarrow A$.*

Definition 1.3 incorporates the involutivity of negation and De Morgan's laws. This is done by adding a negated constant *$\overline{P}$ beside each constant P and by defining negation as a* function *acting on formulas as prescribed by De Morgan's laws and the involutivity of negation: it exchanges $\wedge$ and $\vee$ and each atom with its negated form.*

Please observe that negated constants are atoms: $\overline{\cdot}$ is not *a connective, it makes* no *sense to write $\overline{A}$ for an arbitrary formula A. In particular, $\neg A$ and A have the same size according to Definition 1.3. Also, the involutivity of negation and De Morgan's laws become trivialities in Definition 1.3, because $\neg\neg A$ and A are literally the same formula, as are $\neg(A \wedge B)$ and $\neg A \vee \neg B$, etc. This makes Definition 1.3 only suitable for classical logic, because it is incapable of dealing with systems in which negation is not involutive and De Morgan's laws do not hold (such as intuitionistic logic).*

The truth semantics of formulas with atomic negation is defined as follows:

$$\frac{}{v \models P}\ {\scriptstyle v(P)=1} \qquad\qquad \frac{}{v \models \overline{P}}\ {\scriptstyle v(P)=0}$$

$$\frac{v \models A \quad v \models B}{v \models A \wedge B} \qquad \frac{v \models A}{v \models A \vee B} \qquad \frac{v \models B}{v \models A \vee B}$$

Figure 1.2: The truth semantics of propositional classical logic, with atomic negation

Definition 1.4 (valuation, atomic negation) *A* valuation *is a function from non-negated propositional atoms to* $\{0, 1\}$*. The relation* $v \models A$ *is defined in Fig. 1.2. The relation* $\not\models$ *is defined as the negation of* $\models$*.*

Remark 1.4 *We see here a first example of how atomic negation simplifies definitions (and proofs): with the formulation of Definition 1.3, the truth semantics of propositional logic only requires 5 rules instead of 12, because negation is not a primitive connective (hence there is one inductive case less) and because the relation* $\models$ *does not require the simultaneous definition (by mutual induction) of the relation* $\not\models$*, as is the case with Definition 1.1 (see Fig. 1.1). Instead,* $\not\models$ *is defined simply as the complement of* $\models$*, i.e.,* $v \not\models A$ *iff it is not the case that* $v \models A$*. In particular, Proposition 1.1 becomes trivial, i.e., it holds by definition.*

Chapter 2

Sequent Calculus

2.1 Two-sided formulation

Definition 2.1 (sequent) *A* sequent *is an expression of the form* $\Gamma \vdash \Delta$*, where* Γ *and* Δ *are lists of formulas, as defined in Definition 1.1.*

Remark 2.1 *In a sequent* $\Gamma \vdash \Delta$*, one or both of* Γ *and* Δ *may be empty.*

The intuition behind a sequent $\Gamma \vdash \Delta$ is "from the conjunction of all formulas in Γ, it is possible to prove at least one of the formulas in Δ". Therefore, $\vdash A$ intuitively means "A is provable".

Introduced by Gerhard Gentzen in 1934, *sequent calculus* is a collection of rules allowing to infer the validity of a sequent from the the validity of other sequents. Some sequents are considered to be always valid and therefore constitute the "starting point" of the calculus. The rules of sequent calculus for classical propositional logic, which is abbreviated as **LK**, are given in Fig. 2.1. The sequents above a rule are called its *premises* and the sequent below is called its *conclusion*.

Rules are divided in three groups:

identity rules: essentially, they state the equivalence between an occurrence of formula on the left and on the right of $\vdash$;

structural rules: they concern the structure of sequents, not the logical connectives; in particular, they state that the order in which formulas appear is irrelevant and they allow changing the number of repetitions of a formula in a sequent;

logical rules: they introduce the logical connectives. In fact, one may say that the meaning of a logical connectives is given by its rules. Each connective has two rules, introducing it on the left or on the right of $\vdash$.

Let us discuss the individual rules more thoroughly and check their validity at an intuitive level:

identity: it is one of the initial rules, stating that the validity of the sequent $A \vdash A$ is supposed to always hold, for any A. This is arguably natural, as the sequent states that "from A one can prove A". The rule also says that A on the left is stronger than A on the right.

Identity rules:

$$\frac{}{A \vdash A}\ \mathsf{id} \qquad \frac{\Gamma \vdash \Delta, A \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ \mathsf{cut}$$

Structural rules:

$$\frac{\Gamma, A, B, \Gamma' \vdash \Delta}{\Gamma, B, A, \Gamma' \vdash \Delta}\ \mathsf{x}\vdash \qquad \frac{\Gamma \vdash \Delta, A, B, \Delta'}{\Gamma \vdash \Delta, B, A, \Delta'}\ \vdash\mathsf{x}$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\ \mathsf{w}\vdash \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\ \vdash\mathsf{w}$$

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ \mathsf{c}\vdash \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\ \vdash\mathsf{c}$$

Logical rules:

$$\frac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\ \neg\vdash \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \Delta, \neg A}\ \vdash\neg$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\ \top\vdash \qquad \frac{}{\vdash \top}\ \vdash\top$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge\vdash \qquad \frac{\Gamma \vdash \Delta, A \quad \Gamma' \vdash \Delta', B}{\Gamma, \Gamma' \vdash \Delta, \Delta', A \wedge B}\ \vdash\wedge$$

$$\frac{}{\bot \vdash}\ \bot\vdash \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \bot}\ \vdash\bot$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \vee B \vdash \Delta, \Delta'}\ \vee\vdash \qquad \frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \vee B}\ \vdash\vee$$

Figure 2.1: Sequent calculus for classical propositional logic (**LK**)

cut: it states the transitivity of logical consequence. One may recognize *modus ponens* as a special case:

$$\frac{\vdash A \quad A \vdash B}{\vdash B}$$

i.e., if A is provable and from A one may prove B, then B is provable. It therefore says that A on the right is stronger than A on the left.

exchange: the rules ($\mathsf{x}\vdash$) and ($\vdash \mathsf{x}$) state that, on both sides of the sequent, the order in which formulas appear does not matter. This is morally sound, because we said that commas should be understood as conjunctions on the left and disjuctions on the right, and conjunction and disjunction are both commutative. These rules are technically important because they allow us to state the other rules in a more compact way: without them,

the cut rule would for instance have to be stated as

$$\frac{\Gamma \vdash \Delta, A, \Delta' \quad \Gamma', A, \Gamma'' \vdash \Delta''}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

and every other rule would also have to be stated in a similar, more verbose way. Although useful from the formal point of view, in practice exchange rules are always left implicit.

weakening: on the left ($\mathsf{w} \vdash$) it states that one may freely add useless hypotheses: if Γ alone already proves a formula in Δ, then Γ together with A is obviously still enough. On the right ($\vdash \mathsf{w}$), it says that one may freely add unreachable conclusions: if Γ proves a formula in Δ, then it also proves a formula in Δ or A. Retrospectively (*i.e.*, in light of the cut-elimination theorem and the Curry-Howard correspondence), weakening asserts that logical truth may be discarded, ignored.

contraction: on the left ($\mathsf{c} \vdash$) it states that having used a hypothesis twice is the same as having used it once. On the right ($\vdash \mathsf{c}$), it states that having obtained a conclusion twice is the same as having obtained it once. Retrospectively, contraction asserts that logical truth is inexhaustible, *i.e.*, that it may be used arbitrarily many times without ever being tarnished: the Pythagorean theorem is just as true now as it was three thousand years ago, in spite of people having invoked it countless times in the meanwhile.

negation: the left rule ($\neg \vdash$) states that if A is a consequence of Γ, then assuming both Γ and $\neg A$ at the same time generates a contradiction. In particular, it comprises the *law of non-contradiction* ($A \wedge \neg A$ is contradictory). The right rule ($\vdash \neg$) states that if A proves one of Δ, then one of Δ (in case A holds) or $\neg A$ (in case A does not hold) is provable. In particular, it embodies the *law of excluded middle* ($A \vee \neg A$ is provable). The rules also say that $\Gamma \vdash \Delta$ is the same as $\Gamma, \neg\Delta \vdash$ and $\vdash \neg\Gamma, \Delta$ (where $\neg\Gamma$ and $\neg\Delta$ denote the lists obtained from Γ and Δ by negating all occurrences of formulas).

truth and conjunction: the left rules reflect the intuitive meaning of a sequent, according to which a comma on the left is a conjunction ($\wedge \vdash$), of which $\top$ (truth) is the neutral element ($\top \vdash$). The right rules follow the intuitive meaning of the logical connectives: rule $\vdash \top$ states that truth is always provable and rule $\vdash \wedge$ states that, if Γ proves one of Δ or A and Γ' proves one of Δ or B, then the conjunction of Γ, Γ' proves one of Δ, Δ' or $A \wedge B$ (this is especially evident when Δ and Δ' are empty, but the reader is encouraged to check that the rule is consistent with the intuitive meaning in all cases).

falsehood and disjunction: the rules are perfectly symmetric with respect to those of truth and conjunction, following the intuition that commas on the right are disjunctions and that hypotheses and conclusions are dual (as also stated by the negation rules).

Definition 2.2 (derivation) *A* derivation *is a tree whose nodes are labeled by sequents and which is built from the rules of Fig. 2.1.*

Definition 2.3 (derivable and admissible rules) *A rule*

$$\frac{\Gamma_1 \vdash \Delta_1 \quad \dots \quad \Gamma_n \vdash \Delta_n}{\Gamma \vdash \Delta}\ \mathsf{R}$$

is derivable *if there exists a derivation with hypotheses* $\Gamma_1 \vdash \Delta_1, \dots, \Gamma_n \vdash \Delta_n$ *and endsequent* $\Gamma \vdash \Delta$.

A rule R *as above is* admissible *if, for every proof using the rule* R*, there exists a proof of the same endsequent not using* R.

Remark 2.2 *Both derivability and admissibility of a rule* R *may be understood as saying that* R *is redundant, i.e., provability is not affected by the presence or absence of* R. *However, derivability is stronger: if* R *is derivable by a derivation* δ*, take a proof using* R *and "expand" each instance of* R *into* δ. *The resulting proof obviously has the same endsequent, so* R *is admissible. Hence derivable intuitively means "redundant because definable as a macro".*

In particular, if R *is derivable by* δ *and* π *is a proof of size* s *containing* R*, there is a proof* π' *not containing* R *of size at most* $s \cdot c$*, where* c *is the size of* δ. *On the other hand, when* R *is admissible, there may in general be no obvious relationship between* π *(the proof using* R*) and* π' *(the proof of the same endsequent not using* R*).*

Example 2.1 *In case implication is considered as a primitive connective, it has the following rules:*

$$\frac{\Gamma \vdash \Delta, A \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \Rightarrow B \vdash \Delta, \Delta'}\ {\Rightarrow}{\vdash} \qquad\qquad \frac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta, A \Rightarrow B}\ {\vdash}{\Rightarrow}$$

Let us verify that the definition $A \Rightarrow B := \neg A \vee B$ *is sound, i.e., let us check that the above rules are derivable in our formulation of* **LK***:*

$$\frac{\dfrac{\Gamma \vdash \Delta, A}{\Gamma, \neg A \vdash \Delta}\ {\neg}{\vdash} \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', \neg A \vee B \vdash \Delta, \Delta'}\ {\vee}{\vdash} \qquad\qquad \frac{\dfrac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta, \neg A, B}\ {\vdash}{\neg}}{\Gamma \vdash \Delta, \neg A \vee B}\ {\vdash}{\vee}$$

2.2 One-sided formulation

The use atomic negation, that is, working with Definition 1.3 instead of Definition 1.1, induces great simplifications to the formulation of sequent calculus. The idea comes from the following:

Lemma 2.1 $\Gamma \vdash \Delta$ *is provable iff* $\vdash \neg\Gamma, \Delta$ *is provable, where* $\neg\Gamma$ *is the list obtained by adding a negation to all formulas in* Γ.

Proof. The forward implication is immediate: just take a proof of $\Gamma \vdash \Delta$ and add n ($\vdash \neg$) rules to it, where n is the number of occurrences of formulas in Γ.

For the converse, from a proof of $\vdash \neg\Gamma, \Delta$ we obtain a proof of $\neg\neg\Gamma \vdash \Delta$ by adding n ($\neg \vdash$) rules. Now, it is immediate to see that $A \vdash \neg\neg A$ is provable for all A, so we just cut one by one the formulas in $\neg\neg\Gamma$ and obtain $\Gamma \vdash \Delta$, as desired. □

Identity rules:

$$\frac{}{\vdash \neg A, A}\ \mathsf{id} \qquad\qquad \frac{\vdash \Gamma, \neg A \quad \vdash \Delta, A}{\vdash \Gamma, \Delta}\ \mathsf{cut}$$

Structural rules:

$$\frac{\vdash \Gamma}{\vdash \Gamma, A}\ \mathsf{w} \qquad\qquad \frac{\vdash \Gamma, A, A}{\vdash \Gamma, A}\ \mathsf{c}$$

Logical rules:

$$\frac{}{\vdash \top}\ \top \qquad\qquad \frac{\vdash \Gamma}{\vdash \Gamma, \bot}\ \bot$$

$$\frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \wedge B}\ \wedge \qquad\qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \vee B}\ \vee$$

Figure 2.2: One-sided propositional **LK**

Lemma 2.1 tells us that it is in principle enough to work with *one-sided* sequents, *i.e.*, sequents of the form $\vdash \Gamma$, where the left of $\vdash$ is empty. Of course, the proof of Lemma 2.1 uses negation rules, which need two-sided sequents. This is where Definition 1.3 enters the picture: if negation is no longer a primitive connective, we no longer need a rule for it and we may therefore work with derivations which treat one-sided sequents only.

One-sided **LK** is defined by the rules of Fig. 2.2. Compared with Fig. 2.1, the number of rules is halved: 8 instead of 16.

Remark 2.3 *The symmetric version of Lemma 2.1 also holds, i.e., we may in principle work with one-sided sequents of the form $\Gamma \vdash$. So why did we prefer the left side to be empty instead of the right? This is because we are ultimately interested in judgments stating that "A is provable", i.e., sequents of the form $\vdash A$. On the contrary, $A \vdash$ means "a contradiction may be proved from A", which is not as natural for our purposes.*

Example 2.2 *Let us give an example of admissibility. Consider the calculus consisting of only the rules $\wedge$ and $\vee$ of one-sided* **LK***, plus the "weakened" identity rule*

$$\frac{}{\vdash \Gamma, \neg A, A}$$

This is a version of the identity rule incorporating weakening: it is obviously derivable in one-sided **LK** *from an* id *rule and several* w *rules. Let us call such a calculus $\mathcal{C}$. It is possible to show that the weakening rule* w *is admissible in $\mathcal{C}$, although not derivable. Intuitively, in the calculus $\mathcal{C} + \mathsf{w}$ weakening rules may be "pushed up" the derivation tree, until they are situated just below identity rules. At that point, weakenings may be "absorbed" into the identity rules, which are modified precisely for this purpose. (This is actually true for all of* **LK***, we considered here a simplified calculus because the admissibility result is quicker to prove).*

Chapter 3

First-order Quantification

3.1 Formulas and truth semantics

Definition 3.1 (first-order language) *A* first-order language $\mathcal{L}$ *is given by*

- *a collection of* function symbols*, ranged over by* f*;*
- *a collection of* predicate symbols*, ranged over by* P*;*

such that each symbol σ *(function or predicate) is associated with a non-negative integer called its* arity*, denoted by* $\mathrm{ar}(\sigma)$*. Function symbols of arity zero are called* first-order constants*, whereas predicate symbols of arity zero are called* propositional constants*.*

Definition 3.2 (term) *The* terms *on a first-order language* $\mathcal{L}$ *are defined by the following grammar:*

$$t, u ::= x \mid f(t_1, \ldots, t_n),$$

where x *ranges over a countable set of* first-order variables *and* f *ranges over the function symbols of* $\mathcal{L}$*, with* $n = \mathrm{ar}(f)$*.*

Example 3.1 (the language of basic arithmetic) *A very important example of first-order language is the language of basic arithmetic, denoted by* $\mathcal{L}_{\mathrm{arith}}$*. It consists of:*

- *a first-order constant* $\mathsf{0}$*, a unary function symbol* S *and two binary function symbols* $+$ *and* $\times$*, which are usually written in infix form, i.e., one writes* $t + u$ *and* $t \times u$ *instead of* $+(t, u)$ *and* $\times(t, u)$*, respectively;*
- *two binary predicate symbols* $=$ *and* $<$*, which are also used in infix form; their negated forms are written* $\neq$ *and* $\geq$*.*

The intended meaning of the function symbols is as follows: $\mathsf{0}$ *represents the number* 0*,* S *the successor function,* $+$ *addition and* $\times$ *multiplication. The equality symbol stands, quite obviously, for equality between non-negative numbers, and* $<$ *stands for the strictly-less-than relation.*

We inductively associate with every $n \in \mathbb{N}$ *a term of* $\mathcal{L}_{\mathrm{arith}}$*, called* numeral*, as follows:* $\overline{0} := \mathsf{0}$ *and* $\overline{n+1} := \mathsf{S}(\overline{n})$*. The parentheses of the* S *symbol are usually omitted and one writes for instance* $\mathsf{SSS0}$ *for* $\overline{3}$*. The intended meaning is that* $\overline{n}$ *represents the number* n*.*

For conciseness, we adopt the atomic negation approach, so we extend Definition 1.3 instead of Definition 1.1.

Definition 3.3 (first-order formula) *Given a first-order language $\mathcal{L}$, we extend it by adding a* negated predicate symbol $\overline{P}$ *for each predicate symbol P of $\mathcal{L}$, such that* $\mathrm{ar}(\overline{P}) = \mathrm{ar}(P)$. *The* first-order formulas *on $\mathcal{L}$ are generated by the following grammar:*

$$A, B ::= P(t_1, \dots, t_n) \mid \overline{P}(t_1, \dots, t_n) \mid \top \mid A \wedge B \mid \bot \mid A \vee B \mid \forall x.A \mid \exists x.A,$$

where $t_1, \dots, t_n$ range over terms on $\mathcal{L}$ and $n = \mathrm{ar}(P)$.

The set of free first-order variables *of a formula A is denoted by* $\mathrm{fv}(A)$ *and defined inductively as follows. Firs one defines, given a term t,* $\mathrm{fv}(t)$ *as the set of first-order variables appearing in t; then one sets*

$$\begin{aligned} \mathrm{fv}(P(t_1, \dots, t_n)) := \mathrm{fv}(\overline{P}(t_1, \dots, t_n)) &:= \bigcup_{i=1}^{n} \mathrm{fv}(t_i) \\ \mathrm{fv}(\top) := \mathrm{fv}(\bot) &:= \emptyset \\ \mathrm{fv}(A \wedge B) := \mathrm{fv}(A \vee B) &:= \mathrm{fv}(A) \cup \mathrm{fv}(B) \\ \mathrm{fv}(\forall x.A) := \mathrm{fv}(\exists x.A) &:= \mathrm{fv}(A) \setminus \{x\} \end{aligned}$$

A variable x is free *in A if $x \in \mathrm{fv}(A)$. If x appears in A but it is not free, we say that it is* bound *in A. A formula A is* closed *if $\mathrm{fv}(A) = \emptyset$, i.e., it has no free variable.*

If Γ is a list of formulas $A_1, \dots, A_n$, we write $\mathrm{fv}(\Gamma) := \bigcup_{i=1}^{n} \mathrm{fv}(A_i)$.

Remark 3.1 *A quantifier $\forall x.A$ or $\exists x.A$ is said to* bind *the occurrences of x in A, so a quantifier is a particular form of* binder. *Binders are informally used in many places in mathematics; for instance, in the expression*

$$\sum_{n=1}^{\infty} \frac{m}{n^2}$$

the variable n is bound by the sum operator, whereas the variable m is free. A closed formula is also called a sentence.

The name of a bound variable does not matter: in the above example, we clearly have

$$\sum_{n=1}^{\infty} \frac{m}{n^2} = \sum_{k=1}^{\infty} \frac{m}{k^2}.$$

In logic, this is often referred to as α-renaming *and is treated as equality: two formulas differing only in the names of bound variables are considered equal. For instance,*

$$\forall n.x^n + y^n = z^n \quad \textit{is the same as} \quad \forall k.x^k + y^k = z^k$$

Definition 3.4 (substitution) *Let t, u be a terms. We define $u[t/x]$ as the term obtained for u by substituting every occurrence of the variable x with t. Formally, the definition is by induction on u:*

$$y[t/x] := \begin{cases} t & \text{if } y = x \\ y & \text{if } y \neq x \end{cases} \qquad f(u_1, \dots, u_n)[t/x] := f(u_1[t/x], \dots, u_n[t/x]).$$

Let t be a term and A a formula. We define the formula $A[t/z]$ (substitution of t for x in A) by induction on A, as follows:

$$P(u_1, \ldots, u_n)[t/x] := P(u_1[t/x], \ldots, u_n[t/x])$$

$$\overline{P}(u_1, \ldots, u_n)[t/x] := \overline{P}(u_1[t/x], \ldots, u_n[t/x])$$

$$\begin{array}{rr} \top[t/x] := \top & \bot[t/x] := \bot \\ (A \wedge B)[t/x] := A[t/x] \wedge B[t/x] & (A \vee B)[t/x] := A[t/x] \vee B[t/x] \\ (\forall y.A)[t/x] := \forall y.A[t/x] & (\exists y.A)[t/x] := \exists y.A[t/x] \end{array}$$

Remark 3.2 *If we see u and t as trees, the substitution $u[t/x]$ is taking each leaf of u labelled by x and plugging into it the root of a copy of the tree t. The substitution $A[t/x]$ does the same, but in all terms appearing in A.*

Note that the substitution $(\forall x.A)[t/x]$ never happens because, by α-renaming, $\forall x.A$ is the same as $\forall y.A[y/x]$, i.e., the bound variable may always be supposed to be different from x.

Definition 3.5 (first-order structure) *Let $\mathcal{L}$ be a first-order language. A* first-order structure *for $\mathcal{L}$, which we denote by $\mathcal{S}$, consists of:*

- *a non-empty collection $\mathcal{D}_{\mathcal{S}}$ of* individuals*, called the* domain *of the structure;*
- *for each function symbol f of $\mathcal{L}$ of arity n, a function $\mathcal{S}(f) : \mathcal{D}_{\mathcal{S}}^n \longrightarrow \mathcal{D}_{\mathcal{S}}$; if f is a constant (i.e., the arity is zero), then $\mathcal{S}(f)$ is an individual;*
- *for each predicate symbol P of $\mathcal{L}$ of arity n, a collection $\mathcal{S}(P) \subseteq \mathcal{D}_{\mathcal{S}}^n$; if P is a propositional constant (i.e., the arity is zero), then $\mathcal{S}(P) \in \{0, 1\}$.*

Definition 3.6 (environment) *Let A be a first-order formula on a language $\mathcal{L}$ and let $\mathcal{S}$ be a structure for $\mathcal{L}$. An* environment *for A in $\mathcal{S}$ is a function $\eta : X \longrightarrow \mathcal{D}_{\mathcal{S}}$ such that X is a finite set of first-order variables containing* $\mathrm{fv}(A)$*. If $x \notin X$, we write $\eta[x \mapsto d]$ for the interpretation mapping $y \in X$ to $\eta(y)$ and x to $d \in \mathcal{D}_{\mathcal{S}}$.*

Definition 3.7 (interpretation of a term) *Let $\mathcal{S}$ be a first-order structure on a language $\mathcal{L}$, let t be a term on $\mathcal{L}$ and let η be an environment defined on all the first-order variables appearing in t. The* interpretation *of t in $\mathcal{S}$ under η, denoted by $[\![t]\!]_\eta^{\mathcal{S}}$, is defined by induction on t:*

$$[\![x]\!]_\eta^{\mathcal{S}} := \eta(x),$$
$$[\![f(t_1, \ldots, t_n)]\!]_\eta^{\mathcal{S}} := \mathcal{S}(f)([\![t_1]\!]_\eta^{\mathcal{S}}, \ldots, [\![t_n]\!]_\eta^{\mathcal{S}}).$$

If t is closed (i.e., it contains no variable), then η is irrelevant and we just write $[\![t]\!]^{\mathcal{S}}$.

Example 3.2 (standard structure for arithmetic) *The* standard structure *for $\mathcal{L}_{\mathrm{arith}}$, denoted by Std, is defined by taking $\mathcal{D}_{\mathsf{Std}} := \mathbb{N}$ (the natural numbers, i.e., the non-negative integers) and by interpreting each function and predicate symbol in the obvious way: $\mathsf{0}$ is the natural number zero, S, $+$ and $\times$*

$$\frac{}{\mathcal{S} \models_\eta P(t_1, \ldots, t_n)} \; ([\![t_1]\!]^{\mathcal{S}}_\eta, \ldots, [\![t_n]\!]^{\mathcal{S}}_\eta) \in \mathcal{S}(P)$$

$$\frac{}{\mathcal{S} \models_\eta \overline{P}(t_1, \ldots, t_n)} \; ([\![t_1]\!]^{\mathcal{S}}_\eta, \ldots, [\![t_n]\!]^{\mathcal{S}}_\eta) \notin \mathcal{S}(P)$$

$$\frac{}{\mathcal{S} \models_\eta \top} \qquad \frac{\mathcal{S} \models_\eta A \quad \mathcal{S} \models_\eta B}{\mathcal{S} \models_\eta A \wedge B} \qquad \frac{\mathcal{S} \models_\eta A}{\mathcal{S} \models_\eta A \vee B} \qquad \frac{\mathcal{S} \models_\eta B}{\mathcal{S} \models_\eta A \vee B}$$

$$\frac{\mathcal{S} \models_{\eta[x \mapsto d]} A \quad \text{for all } d \in \mathcal{D}_{\mathcal{S}}}{\mathcal{S} \models_\eta \forall x.A} \qquad \frac{\mathcal{S} \models_{\eta[x \mapsto d]} A}{\mathcal{S} \models_\eta \exists x.A}$$

Figure 3.1: Truth semantics of first-order classical logic

are successor, addition and multiplication on natural numbers, respectively, and $=$ *and* $\leq$ *are just the equality and the usual order on* $\mathbb{N}$*. Observe that* $[\![\overline{n}]\!]^{\mathsf{Std}} = n$*, i.e., each numeral is interpreted by the corresponding integer.*

This is usually referred to as the standard model *(not structure) because it is in fact a model of Peano arithmetic, a logical theory which we will define later.*

Definition 3.8 (model) *Let* $\mathcal{L}$ *be a first-order language, let* A *be a formula on* $\mathcal{L}$*, let* $\mathcal{S}$ *be a first-order structure for* $\mathcal{L}$ *and let* η *be an environment for* A *in* $\mathcal{S}$*. We define the relation* $\mathcal{S} \models_\eta A$ *as in Fig. 3.1. The relation* $\not\models$ *is defined as the complement of* $\models$*.*

If A *is a sentence (i.e., it is closed), then the environment* η *is irrelevant and we write directly* $\mathcal{S} \models A$*. If this holds, we say that* $\mathcal{S}$ *is a* model *of* A*. Otherwise, i.e., when* $\mathcal{S} \not\models A$*, we say that* $\mathcal{S}$ *is a* countermodel *of* A*.*

A sentence A *is* satisfiable *if it has a model and* unsatisfiable *otherwise. It is* valid *if it has no countermodel, in which case we write* $\models A$*.*

Remark 3.3 *One may wonder why we introduced the environment* η *if in the end we are interested only in the truth of closed formulas (i.e., sentences), which do not need it. The answer is simple: the truth of a sentence* A *in* $\mathcal{S}$ *(the relation* $\mathcal{S} \models A$*) is defined by induction, which means that it is defined in terms of the truth of its subformulas, which may be open and therefore need* η*. This is an example of* induction loading*: in order for a definition or proof by induction to carry over, we often need to define or prove something stronger than what we intended originally.*

Remark 3.4 *Propositional logic may be seen as first-order logic on a language in which there are only predicate constants (i.e., predicate symbols of arity zero). In that case, a first-order structure collapses to a valuation (Definition 1.2) and environments are useless (formulas can never contain first-order variables), so one recovers the definition of propositional model/satisfiablity/validity/etc.*

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, \forall x.A}\ \forall \quad x \notin \mathrm{fv}(\Gamma) \qquad\qquad \frac{\vdash \Gamma, A[t/x]}{\vdash \Gamma, \exists x.A}\ \exists$$

Figure 3.2: Rules for first-order quantifiers, one-sided formulation

3.2 Sequent calculus

Sequent calculus for first-order classical logic, or $\mathbf{LK}^1$, is obtained by adding to the rules of Fig. 2.2 the two rules given in Fig. 3.2. Let us discuss them:

universal quantifier: we may read $\vdash \Gamma, A$ as $\neg\Gamma \vdash A$, which intuitively means that A is provable from some hypotheses (the negation of the formulas in Γ). Now, to assert the provability of $\forall x.A$, we must make sure that x is not used by the hypotheses, *i.e.*, that no formula in $\neg\Gamma$ is making any assumption about x. In other words, x must be *generic*. The side condition of the rule ($x \notin \mathrm{fv}(\Gamma)$) has precisely this purpose.

existential quantifier: let us consider again $\neg\Gamma \vdash A[t/x]$, which intuitively means that $A[t/x]$ is provable from some hypotheses. Now, $A[t/x]$ is precisely the statement "the property A holds for t", from which we obviously obtain the provability of $\exists x.A$ from the same hypotheses. The term t in the rule is said to be the *witness* of the existential quantification.

Remark 3.5 *Like we said above, the side condition $x \notin \mathrm{fv}(\Gamma)$ ensures the* genericity *of x, which is clearly needed for universal quantification: we must make sure that $A[x]$ holds for all x, not just for those satisfying some hypotheses. This requirement is not only unnecessary for existential quantification but it does not even make sense: by α-renaming, the variable x is* never *present, neither in Γ nor in $A[t/x]$.*

Indeed, to make the two rules more symmetric, the universal quantifier rule should be formulated as follows:

$$\frac{\vdash \Gamma, A[z/x]}{\vdash \Gamma, \forall x.A}\ z \notin \mathrm{fv}(\Gamma)$$

It is then apparent how both rules consider a property A of x, where x is just a placeholder, i.e., we may write A as $A[x]$, just like we write $f(x)$ for a function of a real variable x. We might as well use any other variable, e.g. $f(y)$, the function does not change (another example of α-renaming). In the existential rule, the premise states the provability of $A[t]$ for some witness t, which is an arbitrary term, not necessarily a variable. In the universal rule, the premise states the provability of $A[z]$ where z is necessarily a variable*, which must furthermore be generic. In neither case does x actually appear anywhere.*

Please observe also that, in both rules, x may not even appear in A. In that case, we are just adding a "dummy" quantifier: A is not speaking of x, so we vacuously have both that A holds for all x and that there exists x satisfying A.

Sequent calculus for first-order logic may of course be given also in the two-sided formulation. In that case, negation is defined as a primitive connective and the calculus is obtained by adjoining the rules of Fig. 3.3 to those of Fig. 2.1.

$$\frac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x.A \vdash \Delta}\ \forall\vdash \qquad \frac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, \forall x.A}\ \vdash\forall\ x\notin \mathrm{fv}(\Gamma,\Delta)$$

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, \exists x.A \vdash \Delta}\ \exists\vdash\ x\notin \mathrm{fv}(\Gamma,\Delta) \qquad \frac{\Gamma \vdash \Delta, A[t/x]}{\Gamma \vdash \Delta, \exists x.A}\ \vdash\exists$$

Figure 3.3: Rules for first-order quantifiers, two-sided formulation

3.3 Ultrafilters

In the rest of the section, we fix a first-order language $\mathcal{L}$.

Definition 3.9 (elementary equivalence) *Let $\mathcal{S}$ be a first-order structure on $\mathcal{L}$. We set*

$$\mathfrak{U}_{\mathcal{S}} := \{A \text{ sentence on } \mathcal{L} \mid \mathcal{S} \models A\}.$$

Two first-order structures $\mathcal{S}, \mathcal{S}'$ on $\mathcal{L}$ are elementarily equivalent *if $\mathfrak{U}_{\mathcal{S}} = \mathfrak{U}_{\mathcal{S}'}$, i.e., if they validate the same sentences, in which case we write $\mathcal{S} \equiv \mathcal{S}'$.*

From the point of view of proof theory, the most interesting aspect of a first-order structure $\mathcal{S}$ is the sentences it validates, not how it validates them. For our purposes, two elementarily equivalent structures should really be considered the same, so the set $\mathfrak{U}_{\mathcal{S}}$ provides us with a useful abstraction, allowing to disregard the details of $\mathcal{S}$ (its domain and how it interprets the symbols of $\mathcal{L}$).

We will now see that we may characterize the sets of the form $\mathfrak{U}_{\mathcal{S}}$, so that we will be able to use such sets instead of first-order structures, obtaining a sort of proof-theoretic version of truth semantics.

Definition 3.10 (ultrafilter) *An* ultrafilter of sentences *is a set $\mathfrak{U}$ of sentences which is:*

- *closed under logical consequence: if $A \in \mathfrak{U}$ and $\vdash \neg A, B$ is derivable in $\mathbf{LK}^1$, then $B \in \mathfrak{U}$;*
- *closed under conjunction: if $A, B \in \mathfrak{U}$, then $A \wedge B \in \mathfrak{U}$;*
- *proper and maximal: for every sentence A, exactly one of $A \in \mathfrak{U}$ or $\neg A \in \mathfrak{U}$ holds.*

The ultrafilter $\mathfrak{U}$ is complete *if, in addition, $A[t/x] \in \mathfrak{U}$ for every closed term t implies that $\forall x.A \in \mathfrak{U}$.*

Remark 3.6 *The notion of ultrafilter is more general: it may be applied to any preordered set. Here, we are giving directly the notion specialized to sentences preordered by $A \preceq B$ if $\vdash \neg A, B$ is provable in $\mathbf{LK}^1$.*

Lemma 3.1 *For every first-order structure $\mathcal{S}$, $\mathfrak{U}_{\mathcal{S}}$ is an ultrafilter.*

Proof. Closure under logical consequence follows from the validity of $\mathbf{LK}^1$ (Proposition 4.2). Closure under conjunction and the fact that $\mathfrak{U}_{\mathcal{S}}$ is proper and maximal are an immediate consequence of the definition of truth. □

So, every first-order structure induces an ultrafilter. In fact, the converse is also true, and this will be a fundamental point in the proof of the completeness theorem. However, for this latter we only need a weaker statement, namely that a *complete* ultrafilter $\mathfrak{U}$ induces a first-order structure whose set of true sentences is exactly $\mathfrak{U}$. The more general statement requires the notion of *Henkin completion*, which we will not cover here.

Let $\hat{\mathcal{L}}$ be the language $\mathcal{L}$ augmented with a constant $\hat{x}$ for each first-order variable x (this technicality will allow us to treat open terms as if they were closed).

Lemma 3.2 *Let $\mathfrak{U}$ be a complete ultrafilter on $\hat{\mathcal{L}}$. Then, there exists a first-order structure $\mathcal{S}_\mathfrak{U}$ such that, for every sentence A,*

$$A \in \mathfrak{U} \qquad \textit{iff} \qquad \mathcal{S}_\mathfrak{U} \models A.$$

Proof. The structure $\mathcal{S}_\mathfrak{U}$ has as domain the set of all closed terms $\mathcal{D}$. The interpretation of function symbols is immediate: if f is of arity n and $t_1, \ldots, t_n \in \mathcal{D}$, we set

$$\mathcal{S}_\mathfrak{U}(f)(t_1, \ldots, t_n) := f(t_1, \ldots, t_n).$$

Let P (or $\overline{P}$) be a propositional constant. We set $\mathcal{S}_\mathfrak{U}(P) = 1$ iff $P \in \mathfrak{U}$, and $\mathcal{S}_\mathfrak{U}(\overline{P}) = 1$ iff $P \notin \mathfrak{U}$. Let P (or $\overline{P}$) be a predicate symbol of arity $n > 0$. We set

$$\mathcal{S}_\mathfrak{U}(P) := \{(t_1, \ldots, t_n) \in \mathcal{D}^n \mid P(t_1, \ldots, t_n) \in \mathfrak{U}\}$$

and

$$\mathcal{S}_\mathfrak{U}(\overline{P}) := \{(t_1, \ldots, t_n) \in \mathcal{D}^n \mid P(t_1, \ldots, t_n) \notin \mathfrak{U}\}$$

We now check the statement of the lemma, by induction on A. The atomic cases are immediate. Let $A = B \wedge C$. For the forward implication, suppose that $B \wedge C \in \mathfrak{U}$. Since $\vdash \neg B \vee \neg C, B$ and $\vdash \neg B \vee \neg C, C$ are both provable, we have $B, C \in \mathfrak{U}$ as well, so we conclude by the induction hypothesis and the definition of truth of a conjunction. For the converse, we consider the contrapositive and let $\mathcal{S}_\mathfrak{U} \not\models B \wedge C$, which means without loss of generality that $\mathcal{S}_\mathfrak{U} \not\models B$. The induction hypothesis gives us $B \notin \mathfrak{U}$, which implies $B \wedge C \notin \mathfrak{U}$ by the above.

Let $A = B \vee C$. For the forward implication, $B \vee C \in \mathfrak{U}$ implies that at least one of $\neg B$, $\neg C$ is not in $\mathfrak{U}$, otherwise $\neg B \wedge \neg C \in \mathfrak{U}$ contradicting the fact that $\mathfrak{U}$ is proper. Suppose without loss of generality that $\neg B \notin \mathfrak{U}$, which by maximality gives $B \in \mathfrak{U}$, which by induction hypothesis gives $\mathcal{S}_\mathfrak{U} \models B$, which allows us to conclude. For the converse, $\mathcal{S}_\mathfrak{U} \not\models B \vee C$ means that $\mathcal{S}_\mathfrak{U} \not\models B$ and $\mathcal{S}_\mathfrak{U} \not\models C$, which by induction hypothesis gives us that neither B nor C is in $\mathfrak{U}$, so by maximality $\neg B, \neg C \in \mathfrak{U}$, which by closure under conjunction implies $\neg B \wedge \neg C \in \mathfrak{U}$ and therefore $B \vee C \notin \mathfrak{U}$ because $\mathfrak{U}$ is proper.

For the quantifier cases we will need an auxiliary result.

Claim. Let η be an environment. Then, $\mathcal{S}_\mathfrak{U} \models_{\eta[x \mapsto t]} C$ iff $\mathcal{S}_\mathfrak{U} \models_\eta C[t/x]$.

Proof. An easy induction on C. □

The most important consequence of the above Claim, which we will use in the sequel, is that $\mathcal{S}_\mathfrak{U} \models \forall x.B$ iff $\mathcal{S}_\mathfrak{U} \models B[t/x]$ for all $t \in \mathcal{D}$.

Let now $A = \forall x.B$. For the forward implication, let $\forall x.B \in \mathfrak{U}$. Now, the sequent $\vdash \exists x.\neg B, B[t/x]$ is provable for all $t \in \mathcal{D}$, hence $B[t/x] \in \mathfrak{U}$ for all $t \in \mathcal{D}$. The induction hypothesis gives us that $\mathcal{S}_\mathfrak{U} \models B[t/x]$ for all $t \in \mathcal{D}$, so we

conclude by the above Claim. For the converse, suppose that $\mathcal{S}_{\mathfrak{U}} \not\models \forall x.B$. By the above Claim, there exists $t \in \mathcal{D}$ such that $\mathcal{S}_{\mathfrak{U}} \not\models B[t/x]$, so by the induction hypothesis we have $B[t/x] \notin \mathfrak{U}$, which implies $\forall x.B \notin \mathfrak{U}$, otherwise we would contradict closure under logical consequence.

Let now $A = \exists x.B$. For the forward implication, assume $\exists x.B \in \mathfrak{U}$. There must exist $t \in \mathcal{D}$ such that $\neg B[t/x] \notin \mathfrak{U}$, otherwise, by completeness, we would have $\forall x.\neg B \in \mathfrak{U}$ and contradict the fact that $\mathfrak{U}$ is proper. Then, maximality gives us $B[t/x] \in \mathfrak{U}$, which implies $\mathcal{S}_{\mathfrak{U}} \models B[t/x]$ by the induction hypothesis, which is enough to conclude $\mathcal{S}_{\mathfrak{U}} \models \exists x.B$ by the above Claim. For the converse, suppose that $\mathcal{S}_{\mathfrak{U}} \not\models \exists x.B$. Then, by the above Claim, $\mathcal{S}_{\mathfrak{U}} \not\models B[t/x]$ for all $t \in \mathcal{D}$. But then, by the induction hypothesis, $B[t/x] \notin \mathfrak{U}$ for all $t \in \mathcal{D}$, which by maximality implies $\neg B[t/x] \in \mathfrak{U}$ for all $t \in \mathcal{D}$, which in turn implies $\forall x.\neg B \in \mathfrak{U}$ by completeness, and we conclude $\exists x.B \notin \mathfrak{U}$ by the fact that $\mathfrak{U}$ is proper. $\square$

Remark 3.7 *As noted above, the addition of the constants $\hat{x}$ is purely technical and will play a role in the proof of the completeness theorem. Here we want to observe that the structure $\mathcal{S}_{\mathfrak{U}}$ is also a structure for the original language $\mathcal{L}$, because $\mathcal{L}$ is obviously a sublanguage of $\hat{\mathcal{L}}$, and this is what matters.*

Chapter 4

Completeness

In this chapter, we fix an arbitrary first-order language $\mathcal{L}$. All formulas are assumed to be on $\mathcal{L}$ and their validity refers to first-order structures for $\mathcal{L}$.

Our aim is to show the following result:

Theorem 4.1 (validity and completeness of $\mathbf{LK}^1$) *For every sentence A,*

$$\models A \quad \textit{iff} \quad \vdash A \textit{ is provable in } \mathbf{LK}^1.$$

The backward direction is called *validity*, because it asserts that $\mathbf{LK}^1$ may only prove valid sentences. The forward direction is called *completeness*, because it asserts that $\mathbf{LK}^1$ proves *all* valid sentences, *i.e.*, it does not "miss" any valid sentence.

We start with validity, which is easy.

Proposition 4.2 (validity) *For every sentence A, $\vdash A$ provable implies $\models A$.*

Proof. We prove a more general statement: $\vdash \Gamma$ provable implies $\models \bigvee \Gamma$, where $\bigvee \Gamma$ is the disjunction of all formulas in Γ. The lemma is obviously a special case. The general statement is shown by a straightforward induction on the proof of $\vdash \Gamma$: each nullary rule (ax and $\top$) introduces a valid sequent and each other rule is easily seen to preserve validity. □

Remark 4.1 *The reason why validity is so simple to prove is that the rules of $\mathbf{LK}^1$ were built with the interpretation "$\vdash \Gamma$ means $\models \bigvee \Gamma$" in mind. Indeed, the discussion we made of the rules after introducing them (Sect. 2.1 and Sect. 3.2) is essentially a proof of their validity.*

Observe that validity implies the *consistency* of $\mathbf{LK}^1$, *i.e.*, the fact that the empty sequent $\vdash$ is not derivable. Indeed, if $\vdash$ were derivable, then $\vdash \bot$ would be derivable, which is forbidden by validity ($\bot$ is not valid).

Completeness is much harder. It was proved for the first time by Gödel in 1929. The proof we give here is due to Schütte. It is a proof by contraposition, *i.e.*, it actually shows the statement "$\nvdash A$ implies $\not\models A$", and it is based on the following idea:

- we define a proof search procedure for attempting to build a proof of A in $\mathbf{LK}^1$;

$$\frac{}{\vdash \Theta, \neg A, \Theta', A, \Theta''; \Psi} \qquad \frac{}{\vdash \Theta, \top, \Theta'; \Psi}$$

$$\frac{\vdash [A \wedge B], \Theta; \Psi, A \quad \vdash [A \wedge B], \Theta; \Psi, B}{\vdash \Theta; \Psi, A \wedge B} \qquad \frac{\vdash [A \vee B], \Theta; \Psi, A, B}{\vdash \Theta; \Psi, A \vee B}$$

$$\frac{\vdash [\forall x.A], \Theta; \Psi, A[z_n/x]}{\vdash \Theta; \Psi, \forall x.A}\ * \qquad \frac{\vdash [\exists x.A], \Theta; \Psi, A[t_n/x]}{\vdash \Theta; \Psi, \exists x.A}\ \dagger$$

$$\frac{\vdash A, \neg C_n, \Theta; A \quad \vdash A, C_n, \Theta; A}{\vdash \Theta, A;}\ \ddagger \qquad \frac{\vdash [A], \Theta; \Psi}{\vdash \Theta; \Psi, A}\ A \text{ atomic}$$

Figure 4.1: The exhaustive search calculus $\mathbf{esLK}^1$.

- under the assumption that $\nvdash A$, the procedure will not find a proof but will instead continue forever;
- using a fundamental result called König's lemma, we show that, if the procedure continues forever, then it generates an infinite set of formulas which is in fact a complete ultrafilter containing $\neg A$;
- as discussed in Sect. 3.3, this implies the existence of a countermodel of A, thus showing $\not\models A$.

4.1 Exhaustive search

First of all, we fix three enumerations

- of first-order variables $z_0, z_1, z_2, \ldots$;
- of terms $t_0, t_1, t_2, \ldots$;
- and of pairs of dual formulas $(C_0, \neg C_0), (C_1, \neg C_1), (C_2, \neg C_2), \ldots$.

We will be thus able to speak of "the n-th term", "the n-th formula", etc.

The proof search will actually be carried on in a variant of $\mathbf{LK}^1$, which we call $\mathbf{esLK}^1$ (*exhaustive search* $\mathbf{LK}^1$) and which is defined by the rules of Fig. 4.1. The calculus $\mathbf{esLK}^1$ manipulates sequents of the form $\vdash \Theta; \Psi$ where Θ and Ψ are lists of formulas and, furthermore, Θ is without repetitions. Given a repetition-free list Θ, we use the notation $[A], \Theta$ to denote the list A, Θ if Θ does not already contain A, or the list Θ itself otherwise. There are four side conditions:

($*$) in the universal rule, n is the smallest integer such that z_n does not appear free in Θ, Ψ;

($\dagger$) in the existential rule, n is the smallest integer such that $A[t_n/x]$ does not appear in Θ;

($\ddagger$) in the cut rule, n is the smallest integer such that neither C_n nor $\neg C_n$ appear in Θ, A;

(•) a non-initial (*i.e.*, unary or binary) rule cannot be applied if its conclusion may also be the conclusion of an initial (*i.e.*, nullary) rule; also, a $\top$ rule cannot be applied if its conclusion may also be the conclusion of an identity rule.

Since $\mathbf{esLK}^1$ is a calculus tailored for proof search, it is best understood by reading its rules bottom-up, from conclusion to premise(s). The search for a proof of a sentence A starts with the sequent $\vdash; A$ and proceeds bottom-up, generating sequents of the form $\vdash \Theta; \Psi$. Intuitively, Ψ is the *workspace* and Θ is the *memory*. The workspace is where formulas are decomposed into their subformulas. The first thing we check is whether $\vdash \Theta; \Psi$ is trivially provable (*i.e.*, whether the memory contains a pair of dual formulas or $\top$), in which case we stop. Otherwise, we decompose the first formula in the workspace and we memorize the fact that such a formula has been processed. In case the formula is a conjunction, two parallel search threads are initiated. When all possible decompositions have been done (*i.e.*, atomic formulas are obtained) without having reached a trivially provable sequent, the workspace becomes empty and a formula is drawn into the workspace from memory, so that the process starts anew. However, if a formula is in memory it means that it has already been processed before, so this time we try to find a proof by adding a new hypothesis, either C_n or $\neg C_n$ (the smallest dual pair not already in memory), thus starting two parallel threads.

The reason why repetitions are not allowed in Θ is clear: it represents a memory, what matters is that a formula is present, not how many times. On the other hand, it is convenient to allow repetitions in the workspace because they arise from the decomposition of formulas like $B \vee B$. The side conditions ($*$, †, ‡ and •) and the fact that we use lists guarantee the determinism of the proof search procedure:

Lemma 4.3 (determinism) *Every sequent $\vdash \Theta; \Psi$ is conclusion of exactly one rule.*

Proof. Suppose first that Ψ is empty. By inspecting Fig. 4.1, we see that the sequent may be the conclusion only of an identity, $\top$ or cut rule. These four cases are mutually exclusive: the first applies if Θ contains a pair of dual formulas; the second applies if Θ does not contain a pair of dual formulas but contains $\top$; the third applies in all other cases.

Suppose now that Ψ is not empty. Then, it contains a rightmost formula A, which completely determines the rule of which $\vdash \Theta; \Psi$ may be the conclusion, depending on the main connective of A or whether it is atomic. □

Of course, a sequent $\vdash \Theta; \Psi$ of $\mathbf{esLK}^1$ immediately corresponds to a sequent of $\mathbf{LK}^1$: simply turn the semicolon into a comma, obtaining $\vdash \Theta, \Psi$.

Proposition 4.4 (soundness of $\mathbf{esLK}^1$ with respect to $\mathbf{LK}^1$) *If $\vdash \Theta; \Psi$ is provable in $\mathbf{esLK}^1$, then $\vdash \Theta, \Psi$ is provable in $\mathbf{LK}^1$.*

Proof. The rules of $\mathbf{esLK}^1$ are all derivable in $\mathbf{LK}^1$. □

Proposition 4.4 basically says that if we start the proof search procedure with $\vdash; A$ and the procedure terminates, then $\vdash A$ is indeed provable in $\mathbf{LK}^1$. The aim of the rest of the section is to prove a converse of Proposition 4.4, *i.e.*,

that if $\vdash A$ is provable in $\mathbf{LK}^1$, then the procedure starting with $\vdash; A$ does find a proof.

Lemma 4.5 (memory weakening) *Let A be a formula not appearing in Θ, Θ'. Then, the weakening rule*

$$\frac{\vdash \Theta, \Theta'; \Psi}{\vdash \Theta, A, \Theta'; \Psi}$$

is admissible in $\mathbf{esLK}^1$.

Proof. Let π be a proof of $\vdash \Theta, \Theta'; \Psi$ in $\mathbf{esLK}^1$. We reason by induction on the last rule of π. If it is an initial rule, we conclude immediately. Suppose now that $\Psi = \Psi', B \wedge C$ and that π terminates with

$$\frac{\vdash [B \wedge C], \Theta, \Theta'; \Psi', B \quad \vdash [B \wedge C], \Theta, \Theta'; \Psi', C}{\vdash \Theta, \Theta'; \Psi', B \wedge C}$$

We apply twice the induction hypothesis to obtain the provability of $\vdash [B \wedge C], \Theta, A, \Theta'; \Psi', C$ and $\vdash [B \wedge C], \Theta, A, \Theta'; \Psi', C$ (observe that, if $A = B \wedge C$, then these two sequents are $\vdash \Theta, B \wedge C, \Theta'; \Psi', C$ and $\vdash \Theta, B \wedge C, \Theta'; \Psi', C$, respectively), from which we conclude by applying a conjunction rule. The other cases are similar. □

Lemma 4.6 (generalized memorization) *The rule*

$$\frac{\vdash [A], \Theta; \Psi}{\vdash \Theta; \Psi, A}$$

where A is not necessarily atomic, is admissible in $\mathbf{esLK}^1$.

Proof. We reason by induction on A. If it is atomic, the rule is already part of $\mathbf{esLK}^1$. Let $A = B \wedge C$. Using weakening (Lemma 4.5) if necessary, we know that $\vdash [B], [B \wedge C], \Theta; \Psi$ and $\vdash [C], [B \wedge C], \Theta; \Psi$ are provable in $\mathbf{esLK}^1$. We apply twice the induction hypothesis and obtain proofs of $\vdash [B \wedge C], \Theta; \Psi, B$ and $\vdash [B \wedge C], \Theta; \Psi, C$, from which we conclude by applying a conjunction rule. The other cases are similar. □

Lemma 4.7 (workspace weakening) *The weakening rule*

$$\frac{\vdash \Theta; \Psi}{\vdash \Theta; \Psi, A}$$

is admissible in $\mathbf{esLK}^1$.

Proof. An immediate consequence of Lemma 4.5 and Lemma 4.6: from a proof of $\vdash \Theta; \Psi$, the former gives us a proof of $\vdash [A], \Theta; \Psi$, from which the latter gives us a proof of $\vdash \Theta; \Psi, A$. □

Lemma 4.8 (cyclic permutation) *If $\vdash \Theta, \Theta';$ is provable in* $\mathbf{esLK}^1$*, then also $\vdash \Theta', \Theta;$ is provable.*

PROOF. By induction on the length of Θ. If Θ is empty, the result vacuously holds. Let $\Theta = A, \Theta''$ and let n be the smallest integer such that neither C_n nor $\neg C_n$ appear in Θ, Θ'. By weakening (Lemma 4.5 and Lemma 4.7), we have a proof of $\vdash A, \neg C_n, \Theta'', \Theta'; A$ and a proof of $\vdash A, \neg C_n, \Theta'', \Theta'; A$, from which a cut rule gives us a proof of $\vdash \Theta'', \Theta', A;$, to which we apply the induction hypothesis. □

Lemma 4.9 (contraction) *The rule*

$$\frac{\vdash A, \Theta; A}{\vdash \Theta, A;}$$

is admissible in $\mathbf{esLK}^1$.

PROOF. From a proof of $\vdash A, \Theta; A$, by weakening (Lemma 4.5) we get a proof of $\vdash A, \neg C_n, \Theta; A$ and a proof of $\vdash A, \neg C_n, \Theta; A$, where n is the smallest integer such that neither C_n nor $\neg C_n$ appear in A, Θ. From these, a cut rule allows us to conclude. □

Lemma 4.10 (cut) *Let m be the smallest integer such that neither C_m nor $\neg C_m$ is in A, Θ, let $n \geq m$ and suppose that $\vdash A, \diamond_n C_n, \ldots, \diamond_m C_m, \Theta;$ is provable in* $\mathbf{esLK}^1$ *for all possible combinations of $\diamond_i$ being either $\neg$ or nothing. Then, $\vdash A, \Theta;$ is also provable in* $\mathbf{esLK}^1$.

PROOF. The proof is by induction on the integer $n-m$. The base case is $n = m$, in which we start with a proof of $\vdash A, \neg C_m, \Theta;$ and a proof of $\vdash A, C_m, \Theta;$. By weakening (Lemma 4.7), we have proofs of $\vdash A, \neg C_m, \Theta; A$ and $\vdash A, C_m, \Theta; A$, so a cut (which we may apply because by hypothesis) yields $\vdash \Theta, A;$, from which the result follows by applying Lemma 4.8.

In case $n - m > 0$, we start with 2^{m-n} proofs of

$$\vdash A, \neg C_n, \diamond_{n-1} C_{n-1}, \ldots, \diamond_m C_m, \Theta;$$

and 2^{m-n} proofs of

$$\vdash A, C_n, \diamond_{n-1} C_{n-1}, \ldots, \diamond_m C_m, \Theta;.$$

After applying weakening (Lemma 4.7) to each of them and combining them pairwise with a cut, we obtain 2^{m-n} proofs of

$$\vdash \diamond_{n-1} C_{n-1}, \ldots, \diamond_m C_m, \Theta, A;$$

from which we conclude by induction hypothesis, after applying Lemma 4.8. □

Lemma 4.11 (existential) *Let m be the smallest integer such that $A[t_m/x]$ is not in Θ and let $n \geq m$. Then, the rule*

$$\frac{\vdash A[t_m/x], \ldots, A[t_n/x], \exists x.A, \Theta;}{\vdash \exists x.A, \Theta}$$

is admissible in $\mathbf{esLK}^1$.

PROOF. The proof is by induction on the integer $n-m$. The base case is $n = m$, in which we have a proof of $\vdash A[t_m/x], \exists x.A, \Theta;$, from which Lemma 4.6 gives us a proof of $\vdash \exists x.A, \Theta; A[t_m/x]$. Applying an existential rule (which is possible because by hypothesis m is the smallest such that $A[t_m/x]$ is not in Θ) gives us a proof of $\vdash \exists x.A, \Theta; \exists x.A$, from which we conclude by Lemma 4.9.

If $n-m > 0$, we have a proof of $\vdash A[t_m/x], \ldots, A[t_{n-1}/x], A[t_n/x], \exists x.A, \Theta;$, from which Lemma 4.8 gives us a proof of

$$\vdash A[t_n/x], \exists x.A, \Theta, A[t_m/x], \ldots, A[t_{n-1}/x];.$$

We now apply Lemma 4.6 and obtain $\vdash \exists x.A, \Theta, A[t_m/x], \ldots, A[t_{n-1}/x]; A[t_n/x]$. An existential rule (which, again, is allowed) yields a proof of

$$\vdash \exists x.A, \Theta, A[t_m/x], \ldots, A[t_{n-1}/x]; \exists x.A,$$

from which Lemma 4.9 gives us a proof of $\vdash \Theta, A[t_m/x], \ldots, A[t_{n-1}/x], \exists x.A;$, and we conclude by applying the induction hypothesis. □

Proposition 4.12 (completeness of esLK^1 with respect to LK^1) *If $\vdash \Gamma$ is provable in* **LK**^{1}*, then $\vdash \Theta; \Psi$ is provable in* **esLK**^{1}*, where Θ is any sequence containing the formulas of Γ and Ψ is arbitrary.*

PROOF. We show that $\vdash \Theta;$ is provable; the provability of $\vdash \Theta; \Psi$ follows from weakening (Lemma 4.7). We proceed by induction on the last rule of the proof of $\vdash \Gamma$. If it is an id, the result is immediate. Suppose that the last rule is a cut

$$\frac{\vdash \Delta, \neg C_n \quad \vdash \Sigma, C_n}{\vdash \Delta, \Sigma}$$

By consistency, at least one of Δ, Σ is non-empty, so that $\Theta = \Theta', A, \Theta''$. Let now m be the least integer such that neither C_m nor $\neg C_m$ appear in Θ, so that $n \geq m$. The induction hypothesis gives us 2^{n-m+1} proofs of $\vdash A, \diamond_n C_n, \ldots, \diamond_m C_m, \Theta'', \Theta';$, precisely as in the hypothesis of Lemma 4.10, which we use to infer the provability of $\vdash A, \Theta'', \Theta';$, from which we obtain $\vdash \Theta;$ by an application of Lemma 4.8.

Suppose now that the last rule is structural. Weakening is addressed by Lemma 4.5. Contraction is trivial, because the multiplicity of formulas in Γ is irrelevant.

Let us consider the logical rules. If the last rule is $\top$, the result is immediate. The rule $\bot$ is treated using Lemma 4.5. Let now the last rule be

$$\frac{\vdash \Delta, A \quad \vdash \Sigma, B}{\vdash \Delta, \Sigma, A \wedge B}$$

and let $\Theta = \Theta', A \wedge B, \Theta''$. Then the induction hypothesis gives us the provability in **esLK**^{1} of $\vdash [A], A \wedge B, \Theta'', \Theta';$ and $\vdash [B], A \wedge B, \Theta'', \Theta';$, from which we obtain the provability of $\vdash A \wedge B, \Theta'', \Theta'; A$ and $\vdash A \wedge B, \Theta'', \Theta'; B$ by applying Lemma 4.6 twice. A conjunction rule then gives us a proof of $\vdash A \wedge B, \Theta'', \Theta'; A \wedge B$, from which Lemma 4.9 gives us a proof of $\vdash \Theta'', \Theta', A \wedge B;$, and we conclude by applying Lemma 4.8.

The cases of $\vee$ and $\forall$ are similar. Let us treat the case of $\exists$, which is more delicate. Suppose that the $\mathbf{LK}^1$ proof ends with

$$\frac{\vdash \Delta, A[t_n/x]}{\vdash \Delta, \exists x.A}$$

Let $\Theta = \Theta', \exists x.A, \Theta''$ and let m be the smallest integer such that $A[t_m/x]$ is not in Θ, so that $n \geq m$. The induction hypothesis gives an $\mathbf{esLK}^1$ proof of

$$\vdash A[t_m/x], \ldots, A[t_n/x], \exists x.A, \Theta'', \Theta';$$

from which we obtain a proof of $\vdash \Theta'', \Theta', \exists x.A$ by Lemma 4.11, and we conclude again by applying Lemma 4.8. □

Note that Proposition 4.12 has as corollary the property we wanted: if $\vdash A$ is provable in $\mathbf{LK}^1$, then the search procedure starting with $\vdash; A$ in $\mathbf{esLK}^1$ will find a proof. Indeed, suppose $\vdash A$ provable in $\mathbf{LK}^1$. There are two cases:

- A is atomic: in that case, the procedure starts with

  $$\frac{\vdash A;}{\vdash; A}$$

  and Proposition 4.12 guarantees us that $\vdash A;$ is provable;
- A is not atomic: in that case, the procedure starts with

  $$\frac{\vdash A; \Psi_1 \quad \ldots \quad \vdash A; \Psi_n}{\vdash; A}$$

  where $\Psi_1, \ldots, \Psi_n$ contain the subformulas of A (in fact, $1 \leq n \leq 2$); again, Proposition 4.12 guarantees us that all the sequents $\vdash A; \Psi_i$ are provable.

So, in both cases, the proof search terminates.

4.2 The completeness proof

We will now consider the rules of Fig. 4.1 coinductively, *i.e.*, we allow possibly infinite trees generated by the rules of $\mathbf{esLK}^1$. Such possibly infinite trees are called *pseudo-proofs*. A proof of course is a pseudo-proof which happens to be finite.

Lemma 4.13 (uniqueness) *For every sequent $\vdash \Theta; \Psi$, there is exactly one pseudo-proof of $\mathbf{esLK}^1$ which ends with that sequent.*

Proof. By Lemma 4.3. □

Lemma 4.14 (König) *Every infinite pseudo-proof contains an infinite branch.*

Proof. Infinite pseud-proofs are finitely branching infinite trees. König's lemma says that such trees necessarily have a finite branch: start from the root; by hypothesis, an infinite number of nodes is reachable from it, and yet

there are only finitely many children nodes, so at least one of those children must be the root of an infinite tree, and so on. □

In the following, given a sequent $\vdash \Theta; \Psi$ of $\mathbf{esLK}^1$, we disregard the ordering of Θ and see it as a set.

Lemma 4.15 (monotonicity) *Let*

$$\vdash \Theta_0; \Psi_0, \vdash \Theta_1; \Psi_1, \vdash \Theta_2; \Psi_2, \ldots, \vdash \Theta_i; \Psi_i, \ldots$$

be the sequents labelling an infinite branch of an infinite pseudo-proof of conclusion $\vdash \Theta_0; \Psi_0$*. Then, for all* $i \in \mathbb{N}$*,* $\Theta_i \subseteq \Theta_{i+1}$*.*

Proof. A simple inspection of Fig. 4.1. □

Suppose that $\vdash A$ is not provable in $\mathbf{LK}^1$, for A a given sentence. By Proposition 4.4, it is not provable in $\mathbf{esLK}^1$ either. Then, the unique (by Lemma 4.13) pseudo-proof π of $\vdash; A$ in $\mathbf{esLK}^1$ must be infinite. By Lemma 4.14, π contains an infinite branch, labelled by the sequents

$$\vdash \Theta_0; \Psi_0, \vdash \Theta_1; \Psi_1, \vdash \Theta_2; \Psi_2, \ldots, \vdash \Theta_i; \Psi_i, \ldots$$

with Θ_0 empty and $\Psi_0 = A$. If C is a formula, we denote by $\hat{C}$ the sentence of the extended language $\hat{\mathcal{L}}$ (see Sect. 3.3) obtained from C by replacing every free occurrence of variable x with $\hat{x}$. We then define, for all $i \in \mathbb{N}$,

$$\hat{\Theta}_i := \{\hat{C} \mid C \in \Theta_i\} \qquad \mathfrak{I} := \bigcup_{i=0}^{\infty} \hat{\Theta}_i, \qquad \mathfrak{U} := \{\neg C \mid C \in \mathfrak{I}\}.$$

We will now prove that the set $\mathfrak{U}$ is a complete ultrafilter containing $\neg A$, which will allow us to conclude $\not\models A$ by Lemma 3.2.

First of all, whether A is atomic or not, the first rule (going bottom-up) of π puts A in Θ_1, and $\hat{A} = A$ because A is closed, so $\neg A \in \mathfrak{U}$.

Before we proceed further, we need an auxiliary lemma:

Lemma 4.16 *For every* $\hat{C} \in \mathfrak{I}$ *and for every* $n \in \mathbb{N}$*, there exists* $k \geq n$ *such that* $\Theta_k = \Theta', C$ *and* Ψ *is empty.*

Proof. Let p be the smallest such that $C \in \Theta_p$, let $m = \max(p, n)$ and, given $i \in \mathbb{N}$, let j_i^C be the the position of C in Θ_{m+i}, counting from the right, starting from 0 (C is guaranteed to be in Θ_{m+i} for all i by Lemma 4.15). It is enough to show that $j_l^C = 0$ for some l such that Ψ_{m+l} is empty, and then we may take $k := m + l$. Now, by inspecting Fig. 4.1, we see that $j_{i+1}^C = j_i^C$ for every rule except cut, for which we have instead $j_{i+1}^C = j_i^C - 1$ (assuming j_i^C is not already null) and in which Ψ_{m+i} is empty. Observe now, again by inspecting Fig. 4.1, that each rule except cut, when read bottom-up, makes the workspace either shrink or be populated by formulas of strictly smaller logical size. This implies that the workspace becomes empty after a finite number of rules (bottom-up), and the claim is proved. □

In other words, Lemma 4.16 guarantees that each formula in memory is pulled out into the workspace infinitely often. In particular, an infinite number of sequents in $(\vdash \Theta_i; \Psi_i)_{i \in \mathbb{N}}$ is the conclusion of a cut rule.

We now prove that $\mathfrak{U}$ is proper and maximal. Since the memory does not shrink (Lemma 4.15), each of the infinitely many cut rules given by Lemma 4.16 introduces a pair of formulas $(C_n, \neg C_n)$ with larger and larger n. Therefore, for all $n \in \mathbb{N}$, there exists $i \in \mathbb{N}$ such that either C_n or $\neg C_n$ is present in Θ_i, proving that $\mathfrak{U}$ is proper. Suppose now that there exists $i \in \mathbb{N}$ such that Θ_i contains both $\neg A$ and A for some A. Then $\vdash \Theta_i; \Psi_i$ would be the conclusion of an identity rule, contradicting the hypothesis that the branch is infinite. This proves that $\mathfrak{U}$ is maximal.

Let us prove closure under logical consequence. Let $A \in \mathfrak{U}$ and suppose that $\vdash \neg A, B$ is provable in $\mathbf{LK}^1$. By definition, $\neg A \in \Theta_i$ for some i. Suppose now, for the sake of contradiction, that $B \notin \mathfrak{U}$, which means $B \in \Theta_j$ for some j. By taking $k := \max(i, j)$, we have that Θ_k contains both $\neg A$ and B (Lemma 4.15). But, by Proposition 4.12, the provability of $\vdash \neg A, B$ in $\mathbf{LK}^1$ implies the provability of $\vdash \Theta_k; \Psi_k$, against the assumption that the only (by Lemma 4.13) pseudo-proof of $\vdash \Theta_k; \Psi_k$ is infinite.

For what concerns closure under conjunction, let $A, B \in \mathfrak{U}$, which means $\neg A, \neg B \in \Theta_i$ for some i. If, for the sake of absurdity, $A \wedge B \notin \mathfrak{U}$, we would have $A \wedge B \in \Theta_j$, hence $\neg A, \neg B, A \wedge B \in \Theta_k$ for $k := \max(i, j)$ (Lemma 4.15). But sooner or later (Lemma 4.16) $A \wedge B$ would be extracted from the memory by means of a cut rule; this would be immediately followed by a conjunction rule and a rule treating A in the workspace, which would bring A in Θ_p for some $p > k$. Again by Lemma 4.15, Θ_p would contain both A and $\neg A$, contradiction.

So $\mathfrak{U}$ is an ultrafilter. Let us prove its completeness. Let us assume that $A[t/x] \in \mathfrak{U}$ for all closed terms t and suppose, for the sake of contradiction, that $\forall x.A \notin \mathfrak{U}$. This implies that $\forall x.A \in \Theta_i$ for some i. Again, by Lemma 4.16, at some point such a formula will be put in the workspace, implying the presence of $A[z_n/x]$ in Θ_k for some z_n and $k > i$. This in turn implies $\neg A[\hat{z}_n/x] \in \mathfrak{U}$. But we started with the assumption that $A[\hat{z}_n/x] \in \mathfrak{U}$, which means that for some $p \geq i$ we have both $\neg A[z_n/x]$ and $A[z_n/x]$ in Θ_p, contradicting the fact that the pseudo-proof is infinite.

Chapter 5

Undecidability and Incompleteness

5.1 Informal computability

Gödel's incompleteness theorems rely crucially on a notion of "mechanizable procedure" or, more technically, of "computable function". Mathematicians throughout history had been aware that certain mathematical constructions are *effective*, *i.e.*, realizable in the physical world (at least theoretically, *i.e.*, ignoring resource limits, or what we would call today *computational complexity*), whereas others are, somehow, purely abstract. However, negative results such as Gödel's required to put this distinction on more formal grounds.

In the early 20th century, several people independently gave formal definitions of the intuitive concept of computable function. Perhaps the three most famous are (in order of appearance):

recursive functions: this is the original concept of computability used by Gödel in his proof of the incompleteness theorem, and was developed and expanded later mostly by Kleene. The idea is the following: we fix a set of functions, the *basic functions*, which are so simple that we "know" they are computable, then we give ways of building new computable functions starting from functions which we have already shown to be computable. Essentially, this amounts to defining a very rudimentary programming language, in which the only data is of type `nat` (the non-negative integers) and in which the only control structure are `while` loops.

λ-calculus: this model of computation was introduced by Church and may be seen as the paradigmatic functional programming language. It turned out to have a strikingly deep connection with a proof theoretic formalism independently developed by Gentzen (natural deduction): the so-called *Curry-Howard correspondence*, which we will see later.

Turing machines: this is perhaps the most famous model of computation, and it is still widely used today as the basis, for example, of computational complexity theory. Turing devised his machines by reflecting on what a mathematician does when he/she performs a computation: he/she has a

sheet of paper (which we may suppose to be extendable at will) on which he/she writes and erases a finite set of symbols according to a fixed, finite set of rules. A Turing machine imitates this behavior, which justifies the intuitive feeling that Turing machines compute exactly what a human can compute.

Each one of the above models yields a notion of computable function on natural numbers: it is a function $f : \mathbb{N} \to \mathbb{N}$ such that f may be computed (according to a formal definition which depends on the model) by the programs/machines/λ-terms considered in the model. These formalized versions of the notion of computable function are usually called *recursive*, *Turing-computable* and λ-*definable*, respectively.

Wonder of wonders, they all turn out to coincide. Furthermore, any other "reasonable" model of computation introduced in the sequel also turned out to yield exactly the same class of computable functions. This is why people believe in the so-called *Church-Turing thesis*: there is only one formal notion of computable function on the natural numbers, which is given by any reasonable model of computation as expressive as recursive functions or Turing machines or λ-calculus, and which exactly captures our intuitive notion of "computable function".

Here, we will not introduce any formal model of computation but rather speak informally of "programs" as we intuitively understand them, *i.e.*, finite sets of instructions executable on some kind of machine which are powerful enough to compute any computable function. We will appeal to the following intuitive notions, which we hope are acquired and understood by any graduate student in computer science:

- programs may manipulate any datatype whose elements are finite objects, such as Booleans, natural numbers, formulas, sequents, sequent calculus proofs, programs themselves, etc.

- Programs may call other programs, start parallel threads executing other programs, communicate one with the other, etc.

- A set X of elements of type `data` (such as a set of natural numbers or formulas) is called *decidable* if there exists a program P : `data` $\to$ `bool` which, given in input e : `data`, outputs `true` if $e \in X$ and outputs `false` if $e \notin X$. We then say that P *decides* X. If no such program exists, X is said to be *undecidable*.

- A set X as above is called *recursively enumerable* (or *semi-decidable*) if there exists a program P : `data` $\to$ `bool` which, given in input e : `data`, outputs `true` if $e \in X$ and may output `false` or run forever if $e \notin X$. In that case, we say that P *semi-decides* X.

- Given a set X as above, we denote by $\overline{X}$ its complement, *i.e.*, the set of elements of type `data` which are not in X. Then, X decidable implies $\overline{X}$ decidable (simply take the program deciding X and add a negation to it).

- On the other hand, the complement of a recursively enumerable set is called *co-recursively enumerable* and is in general *not* recursively enumerable. In fact, a set X is both recursively enumerable and co-recursively

enumerable exactly when it is decidable (for the forward direction, a program deciding X may be constructed by launching two parallel threads, one running the program semi-deciding X and the other running the program semi-deciding $\overline{X}$; the first thread that stops gives the answer. The converse is obvious).

- There exist undecidable (but semi-decidable) sets, the mother of all of them being the set of programs that terminate when given some fixed input (of the suitable datatype).

5.2 Incompleteness: a road map

The job of a mathematician is to prove theorems. But what is a theorem? And what does it take to prove one? Let us attempt to give informal definitions. A theorem is, in a first approximation, a statement about some kind of mathematical world, which may or may not have direct connections to the physical world. The objects populating this world (geometrical shapes, integer numbers, complex numbers, differential equations, probability distributions, presheaves, toposes...) exist somewhere, exactly where is a fascinating question which we will leave the reader to ponder upon, here let us just say that it is in the collective imagination of mathematicians.

Now, more precisely, a theorem is a *true* statement about the mathematical world. Truth too, however, is an intuitive notion: for instance, we feel that $0 = 1$ is false, while $3 + 2 = 5$ is true. These are extremely simple statements, whose truth or falsity is obvious, but it is enough to rise the complexity by a few order of magnitudes to make ourselves feel less sure about "truth" (consider, for example, the statement "982 451 653 is prime"). So, how do we make truth more precise? How can we be sure that a statement is a theorem?

The most widely accepted way is to resort to the so-called *axiomatic method*, introduced by Euclid in the 3rd century BC. We start from a set of basic statements, called *axioms* (or *postulates*), whose truth is supposedly self-evident and incontrovertible. Then, we devise a set of rules that allow us to derive true statements from true statements, *i.e.*, new theorems from old theorems (or axioms). The validity of these rules too must be self-evident, as is the fact that applying any number of them to a true statement still yields a true statement. This, and nothing else, is how mathematicians prove theorems.[1]

What we have done in the previous chapters may be seen as providing ourselves with formidably precise tools to implement the axiomatic method. We fix

[1] Two remarks are in order here. First, observe that the "liberalism" of the mathematical world with respect to the physical world is what allows mathematics to be an *exact* science, *i.e.*, a science in which something may indeed be "proved". In physical sciences, nothing can ever be "proved": there is no guarantee that our current laws of physics, which seem to predict very well the behavior of the reality around us, will stop working tomorrow. In some sense, mathematics can be exact because it establishes its own rules.

Second, it is important to stress here the ultimate social nature of mathematics: both the axioms and the rules are established by consensus within the community of mathematicians, they do not "descend from the sky". They may be debated, they may evolve, usually with the intent of making them better suited to the needs of the community, just like any other social construct. The axiomatic method itself is too, in fact, a social construct: although nowadays it is "the" way of doing mathematics, mathematicians existed before Euclid and developed relatively sophisticated mathematics without using the axiomatic method.

a language, which is the "language of mathematics", *i.e.*, the language in which we feel we may express all desirable statements about the mathematical world. Then, we fix our set of axioms, which will formally be called a *theory*. Then, sequent calculus allows us to prove theorems from those axioms under the form of proofs of sequents $\Gamma \vdash A$, where A is the theorem itself and Γ is a finite set of axioms of our theory (those that happen to be necessary to establish the truth of A—we stressed that they are finite because a theory may as well contain infinitely many axioms). We then say that "A is a theorem of the theory T", or "T proves A", and write $T \vdash A$.

The notion of model, which we briefly introduced, formalizes the idea of "mathematical world" and gives us the bridge with truth: a statement A is a *logical consequence* of the theory T, which we write $T \vDash A$, if is is true in any model of T, *i.e.*, if A is true in any mathematical world in which all the axioms of T are true. The completeness theorem (plus a corollary of it, the so-called *compactness theorem*) establishes that there is a perfect harmony between the theorems we may prove and the truth of such theorems:

$$T \vDash A \qquad \text{iff} \qquad T \vdash A.$$

It must be pointed out here that our implementation of the axiomatic method does suffer one restriction: we are limited to mathematical worlds describable by first-order sentences/first-order models. This, however, does not turn out to be a heavy limitation: in the early 20th century logicians devised first-order languages and theories which allow to speak, by means of encodings, of basically any statement whatsoever that mathematicians ever wish to consider (such encodings are admittedly rather artificial, but they technically suffice to exhibit the power of first-order logic). In fact, by reducing geometry to analysis, and analysis to algebra plus topology, and algebra to arithmetic (or number theory), it was clear already by the turn of the 20th century that it was enough to put first-order arithmetic on solid grounds to give solid foundations to the whole mathematical building.

Now, every theory usually has a preferred world which it is inspired upon, which is referred to as the *intended model* of the theory. For instance, in the case of arithmetic this is the set $\mathbb{N} := \{0, 1, 2, 3, \ldots\}$ or, in technical terms, the *standard model* Std introduced in Example 3.2, which uses the language of basic arithmetic introduced in Example 3.1. So the standard model corresponds to the mathematician's intuition of the world of natural numbers and, in the original, informal sense of the word "theorem" we gave above, a theorem of arithmetic is a sentence which is true in Std: these are the theorems number theorists are interested in. Our goal is therefore to devise a theory (*i.e.*, a set of axioms) T in the language of arithmetic such that

$$T \vdash A \qquad \text{iff} \qquad \mathsf{Std} \models A.$$

The forward implication, called *soundness*, is a minimum requirement: why would we be interested in an axiomatic system proving statements which are false in the intended model? On the other hand, a theory verifying the backward implication is called *complete*, because it does not "miss" any theorem.

Gödel's first incompleteness theorem states the impossibility of realizing the above equivalence: a sound theory T which is strong enough to prove certain facts about Std (basically any theory strong enough to attempt to capture all

theorems) is necessarily incomplete, unless its axioms are uncomputable and, therefore, useless from a practical point of view.[2]

What does this mean? From the proof theoretic point of view, it says that no matter how we try to axiomatize arithmetic, provided we do it in a computable (more precisely, recursively enumerable) way, there will always be theorems which our axiomatization will miss.

From the point of view of models, it is useful to realize that a complete theory is a theory which has essentially one model: it may have infinitely many models of all different sorts but these are all elementarily equivalent (Definition 3.9), *i.e.*, given an arithmetic statement, there is only one possible notion of truth associated with it, which is the truth in the standard model. So what Gödel's incompleteness says is that the intended model, the intuitive mathematical world of natural numbers cannot be captured by a recursively enumerable axiomatization. Any such axiomatization is always speaking, along with the intended world of the standard model, of some other weird, non-standard worlds, where there are "integers" satisfying properties we would not have thought.

In both cases, Gödel's theorem says that *the axiomatic method does not work*, because it cannot capture the mathematical world that exists in our intuitions. But the axiomatic method is the only method we have for doing mathematics, which immediately conveys the importance Gödel's result. In practice, of course, this does not mean that we should stop doing mathematics. It simply means that our search for mathematical truth cannot be narrowed to a single, all-encompassing formal system, but it is rather an open-ended activity, which will always need new ideas and new insights (at least until something replacing the axiomatic method is found...). So, after all, Gödel's incompleteness is good news!

We have spoken about the *first* incompleteness theorem, which obviously implies that there is a second theorem. This latter basically gives an example of a true but unprovable sentence of T: the consistency of T itself. In other words, as soon as a sound theory of arithmetic is powerful enough, it cannot prove its own consistency. This result was important at the time of Gödel, because Hilbert's program was essentially a quest for a self-justifying (*i.e.*, self-consistent) elementary theory of mathematics. Today, it is used primarily as the motivation for studying the *relative consistency* of theories: if T cannot prove its own consistency, maybe it can prove the consistency of T', which is therefore strictly weaker, and one may thus study the relationship of theories in terms of so-called consistency strength, which is still an actively developed field of research.

5.3 Logical theories

Let us fix a language $\mathcal{L}$.

[2] There obviously is a sound and complete theory of arithmetic: just take $T := \{A \text{ sentence} \mid \mathsf{Std} \models A\}$. The first incompleteness theorem says that such a T is not even recursively enumerable. We say that this is useless because, in that case, when given an $\mathbf{LK}^1$ proof of $\Gamma \vdash A$, we are unable to determine whether this is indeed a proof that $T \vdash A$, because the statement $\Gamma \subseteq T$ is not even semi-decidable. In other words, there is no way to tell whether a purported proof is indeed a proof...

Definition 5.1 (theory) *A* theory *T on $\mathcal{L}$ is a set of sentences on $\mathcal{L}$, which are referred to as the* axioms *of T. A theory T' on a language $\mathcal{L}'$ is an* extension *of T if $\mathcal{L} \subseteq \mathcal{L}'$ and $T \subseteq T'$. A theory T' is a* subtheory *of T if the latter is an extension of T'.*

A model *of T is a structure $\mathcal{M}$ for $\mathcal{L}$ which verifies $\mathcal{M} \models C$ for every $C \in T$.*

A sentence A is a logical consequence *of T, which we write $T \vDash A$, if $\mathcal{M} \models A$ for every model $\mathcal{M}$ of T. A* conservative extension *of T is an extension T' of T on a language $\mathcal{L}'$ such that, whenever A is sentence of $\mathcal{L}$, $T' \vDash A$ implies $T \vDash A$ (i.e., every formula on the smaller language that is a logical consequence of the bigger theory is already a logical consequence of the smaller theory).*

A sentence A is provable *in T, which we write $T \vdash A$, if there exists a finite subset $\Gamma \subseteq T$ such that the sequent $\Gamma \vdash A$ is provable in $\mathbf{LK}^1$.*

Definition 5.2 *A theory T is:*

consistent *if $T \nvdash \bot$;*

complete *if, for every sentence A, either $T \vDash A$ or $T \vDash \neg A$;*

decidable (or recursively enumerable) *if the set of logical consequences of T is decidable (or recursively enumerable).*

Proposition 5.1 *A theory T is*

1. *consistent iff $T \nvdash A$ for some sentence A iff it has a model;*
2. *complete iff it has only one model model up to $\equiv$ (elementary equivalence,* cf. *Definition 3.9).*
3. *recursively enumerable iff the set T alone is recursively enumerable.*

Proof. For point 1, we reason cotrapositively, *i.e.*, we show that (i) T is inconsistent iff (ii) $T \vDash A$ for every A iff (iii) T has no model. We start with (i) implies (iii): no structure $\mathcal{S}$ verifies $\mathcal{S} \models \bot$, so the only way for $T \vDash \bot$ to hold is that T has no models. For what concerns (iii) implies (ii), call M_T and M_A the class of models of T and of A, respectively, and observe that, by definition, $T \vDash A$ iff $M_T \subseteq M_A$; but $M_T = \emptyset$, so $M_T \subseteq M_A$ regardless of M_A. Finally, (ii) implies (i) is trivial (the latter is a particular case of the former).

For point 2, assume that T is complete and let $\mathcal{M}_1, \mathcal{M}_2$ be two models of T. Now, for every sentence A, either $T \vDash A$ or $T \vDash \neg A$, which by definition means that either $\mathcal{M}_1, \mathcal{M}_2 \models A$ or $\mathcal{M}_1, \mathcal{M}_2 \models \neg A$, hence, by genericity of A, $\mathcal{M}_1 \equiv \mathcal{M}_2$. For the converse, suppose that all models of T validate the same sentences. Now, either T has no model, in which case we conclude by point 1, or it has a model $\mathcal{M}$. Then, given a sentence A, we have that either $\mathcal{M} \models A$, in which case $T \vDash A$ (because every model behaves like $\mathcal{M}$), or $\mathcal{M} \models \neg A$, in which case $T \vDash \neg A$ (for the same reason).

For point 3, the forward implication is trivial. For the backward implication, the fact that T is recursively enumerable allows us to write a program check_T which semi-decides whether a proof of $\Gamma \vdash A$ in $\mathbf{LK}^1$ is indeed a proof of $T \vdash A$: check_T simply starts one parallel thread for each distinct formula C in Γ, in which the program semi-deciding whether $C \in T$ is executed. If all such threads terminate, we may conclude that we have indeed a proof of $T \vdash A$.

Therefore, to semi-decide whether $T \vdash A$, we perform a parallel search for a proof of all possible sequents $\Gamma \vdash A$ with Γ a finite subset of T. In case we do have $T \vdash A$, such a search will terminate, and we may answer positively; otherwise, the program will run forever. $\square$

Remark 5.1 *In the sequel, when we say that a theory is recursively enumerable, we will intend that its set of axioms is recursively enumerable. Thanks to Proposition 5.1.3, this is actually the same as Definition 5.2.*

On the other hand, observe that Proposition 5.1.3 is false when applied to decidable theories: the logical consequences of a decidable set of axioms may very well form an undecidable set (albeit this will certainly be recursively enumerable). Robinson's arithmetic and Peano arithmetic (which we will introduce below) are two examples.

The following is a central result of model theory, which we will not prove here. It may actually be deduced from the completeness theorem (Theorem 4.1).

Theorem 5.2 (compactness) *Let T be a theory such that, for every finite subset $\Gamma \subseteq T$, there exists $\mathcal{M}_\Gamma$ such that $\mathcal{M}_\Gamma \models \Gamma$. Then, there exists $\mathcal{M}$ such that $\mathcal{M} \models T$.*

Remark 5.2 *The converse of Theorem 5.2 is obvious. Hence, in light of Proposition 5.1.1 we may summarize compactness by saying that a theory T is consistent iff every finite subtheory of T is consistent.*

Corollary 5.3 (of Theorem 4.1) *For every theory T and sentence A, we have*

$$T \vDash A \qquad \textit{iff} \qquad T \vdash A$$

Proof. The implication from left to right uses completeness and compactness (Theorem 5.2), which is itself a consequence of completeness. We consider two cases:

- T is inconsistent: in this case, we need to prove that $T \vdash A$ for every sentence A. By compactness, there exists a finite subtheory $\Gamma \subseteq T$ which is also inconsistent, which by completeness means that $\Gamma \vdash \bot$ is provable, from which we obtain the provability of $\Gamma \vdash A$ for every A.
- T is consistent: we reason contrapositively and suppose $T \nvdash A$, which means, by definition, that $\Gamma \vdash A$ is not provable for any finite $\Gamma \subseteq T$. Completeness then gives us a countermodel of $\bigwedge \Gamma \Rightarrow A$, which is a model of the finite theory $\Gamma \cup \{\neg A\}$. Now, since T is consistent, we also trivially have a model of each Γ itself, so we may conclude by compactness that the theory $T \cup \{\neg A\}$ has a model, hence $T \nvDash A$.

The implication from right to left is an immediate consequence of validity: $\Gamma \vdash A$ provable (with Γ a finite subset of T) implies $\bigwedge \Gamma \Rightarrow A$ valid, which in turn is easily seen to imply that $\Gamma \vDash A$, which implies $T \vDash A$ because $\Gamma \subseteq T$. $\square$

Remark 5.3 *By Corollary 5.3, we may replace (where it makes sense) every symbol $\vDash$ with the symbol $\vdash$. This is especially true of Definition 5.2 and Proposition 5.1.*

We conclude this section with the following result, which will be essential in the proof of the first incompleteness theorem.

Proposition 5.4 *A recursively enumerable theory T which is both consistent and complete is decidable.*

PROOF. Observe that, when coupled together, consistency and completeness imply that, for all A, exactly one of $T \vdash A$ or $T \vdash \neg A$ holds: completeness says "at least one of the two", consistency says "at most one of the two". So we may construct a program decide_T deciding whether $T \vdash A$ as follows: we generate, by brute force, all possible $\mathbf{LK}^1$ proofs; whenever we find a proof π of the sequent $\Gamma \vdash A$ or $\Gamma \vdash \neg A$, we open a parallel thread which executes check_T (the program given in the proof of Proposition 5.1.3) on π, and we keep going; by the above property, one of these threads is guaranteed to terminate; if it is a thread checking a proof of a sequent of the form $\Gamma \vdash A$, decide_T terminates answering "provable"; if it is a thread checking a proof of a sequent of the form $\Gamma \vdash \neg A$, decide_T terminates answering "not provable". □

5.4 Arithmetical theories

We already introduced the language $\mathcal{L}_{\text{arith}}$ of basic arithmetic (Example 3.1). It contains:

- a first-order constant $\mathsf{0}$, representing the natural number 0;
- a unary function symbol S, representing the successor function;
- two binary function symbols $+$ and $\times$, written in infix notation and representing addition and multiplication, respectively;
- two binary predicate symbols $=$ and $<$, also written in infix notation and whose negations are written $\neq$ and $\geq$, which stand for the equality and strictly-less-than relations.

We also defined the *numerals* to be the closed terms of the form $\overline{n} := \overbrace{\mathsf{S}\cdots\mathsf{S}}^{n}\mathsf{0}$, for each $n \in \mathbb{N}$, which are supposed to represent the non-negative integers.

Theories on the language $\mathcal{L}_{\text{arith}}$ are known as *arithmetical theories*. The intended model of such theories is the *standard model* Std introduced in Example 3.2, which is the set of natural numbers $\mathbb{N}$ with the usual operations and relations. Therefore, a reasonable desiderata for an arithmetical theory T is that it does not prove sentences which are false in the standard model.

Definition 5.3 (sound arithmetical theory) *An arithmetical theory T is* sound *if, for every sentence A, $T \vdash A$ implies $\mathsf{Std} \models A$.*

Soundness is a fairly strong requirement:

Lemma 5.5 *A sound theory T is consistent.*

PROOF. Observe that $\mathsf{Std} \not\models \overline{0} = \overline{1}$, which by soundness and Corollary 5.3 implies $T \not\vdash \overline{0} = \overline{1}$, so we conclude by Proposition 5.1.1. □

The most famous arithmetical theory is probably (first-order) *Peano arithmetic*. An interesting subtheory of Peano arithmetic is *Robinson's arithmetic*, which we introduce next. First, we start with formulas asserting the basic laws of equality: reflexivity, symmetry, transitivity and contextuality. Any theory with equality would contain similar axioms, adapted to the function and predicate symbols of its language.

$$
\begin{aligned}
E_1 &:= \forall x.x = x \\
E_2 &:= \forall xx'.x = x' \Rightarrow \mathsf{S}x = \mathsf{S}x' \\
E_3 &:= \forall xx'yy'.x = x' \wedge y = y' \Rightarrow x + y = x' + y' \\
E_4 &:= \forall xx'yy'.x = x' \wedge y = y' \Rightarrow x \times y = x' \times y' \\
E_5 &:= \forall xx'yy'.x = x' \wedge y = y' \Rightarrow (x = y \Rightarrow x' = y') \\
E_6 &:= \forall xx'yy'.x = x' \wedge y = y' \Rightarrow (x < y \Rightarrow x' < y')
\end{aligned}
$$

We invite the reader to check that symmetry and transitivity of equality are derivable from the above axioms (in particular, from E_1 and E_5).

Then there are the axioms proper to Robison's arithmetic:

$R_1 := \forall x.\mathsf{S}x \neq \mathsf{0}$	(zero is not the successor of any number)
$R_2 := \forall x.\forall y.\mathsf{S}x = \mathsf{S}y \Rightarrow x = y$	(the successor function is injective)
$R_3 := \forall x.x = \mathsf{0} \vee (\exists y.x = \mathsf{S}y)$	(every number is either zero or a successor)
$R_4 := \forall x.x + \mathsf{0} = x$	(the recursive definition...
$R_5 := \forall x.\forall y.x + \mathsf{S}y = \mathsf{S}(x + y)$	...of addition)
$R_6 := \forall x.x \times \mathsf{0} = \mathsf{0}$	(the recursive definition...
$R_7 := \forall x.\forall y.x \times \mathsf{S}y = (x \times y) + x$	...of multiplication)
$R_8 := \forall x.x \geq \mathsf{0}$	(every number is non-negative)
$R_9 := \forall x.\forall y.x < y \vee x = y \vee y < x$	(the order is total)
$R_{10} := \forall x.\forall y.x < \mathsf{S}y \Leftrightarrow (x < y \vee x = y)$	($x < y + 1$ means either $x = y$ or $x < y$)

Robison's arithmetic, which is denoted by **Q**, consists of the set of all the above axioms. First-order Peano arithmetic, denoted by **PA**, is obtained by adding to **Q** the following infinite set of axioms, one for each formula A of $\mathcal{L}_{\text{arith}}$:

$$\mathsf{Ind}_A := A[\mathsf{0}/x] \Rightarrow (\forall y.A[y/x] \Rightarrow A[\mathsf{S}y/x]) \Rightarrow \forall x.A.$$

The formula Ind_A is obviously stating the induction principle applied to the predicate A, so Peano arithmetic is just Robinson's arithmetic plus induction.

The fundamental property of Robinson's arithmetic is that it is able to represent decidable sets of integers.

Definition 5.4 (representability) *Let T be an extension of* **Q**. *A set $X \subseteq \mathbb{N}$ is* representable *in T if there exists a formula A with one free variable x such*

that

$$\begin{array}{lll} n \in X & \textit{implies} & T \vdash A[\overline{n}/x], \\ n \notin X & \textit{implies} & T \vdash \neg A[\overline{n}/x]. \end{array}$$

Lemma 5.6 *Every decidable set $X \subseteq \mathbb{N}$ is representable in* $\mathbf{Q}$.

Proof. The proof of this (crucial) lemma is very technical, we will not even attempt to give it here. Essentially, one may think of it as consisting of three ingredients:

1. the *Gödelization* of programs, *i.e.*, the fact that every program, as well as the configurations generated by the execution of a program, by virtue of being all finite objects, may be encoded as natural numbers;
2. the *arithmetization* of the execution of programs, *i.e.*, the fact that the manipulation of program configurations induced by the step-by-step execution of such programs may be turned into a function on the numbers representing the configurations;
3. the *expressiveness* of Robinson's arithmetic, which is just enough so that all those numerical functions (or, rather, their graphs) may be encoded by formulas of $\mathcal{L}_{\text{arith}}$ which may be proved or disproved using the axioms of $\mathbf{Q}$.

The idea is then the following: if X is decidable, then there is a program P : `nat` $\to$ `bool` deciding it; the sequence of configurations induced by the execution of P on input n, which is finite (P always terminates) may be encoded by an integer m; using all of the above, one finds

- a formula Exec_P with two free variables x, y such that, if m encodes an execution of P on input n, then $\mathbf{Q} \vdash \text{Exec}_P[\overline{n}/x, \overline{m}/y]$, and $\mathbf{Q} \vdash \neg\text{Exec}_P[\overline{n}/x, \overline{m}/y]$ otherwise;
- a formula Accept_P with one free variable y such that, if the execution of P encoded by m terminates with answer `true`, then $\mathbf{Q} \vdash \text{Accept}_P[\overline{m}/y]$, and $\mathbf{Q} \vdash \neg\text{Accept}_P[\overline{m}/y]$ otherwise.

Then, one may set $A := \exists y.\text{Exec}_P \wedge \text{Accept}_P$. □

Remark 5.4 *It is important to observe that the proof of Lemma 5.6 uses just the language and axioms of* $\mathbf{Q}$. *Therefore, any theory extending* $\mathbf{Q}$ *enjoys Lemma 5.6 and, even though the language of T may extend $\mathcal{L}_{\text{arith}}$, the language used for representing predicates is still $\mathcal{L}_{\text{arith}}$. Actually,* $\mathbf{Q}$ *is in some sense the minimum amount of arithmetic needed for Lemma 5.6 to hold.*

Lemma 5.7 *Any consistent extension T of* $\mathbf{Q}$ *is undecidable.*

Proof. Let us assume, for the sake of absurdity, that T is decidable. Consider a computable enumeration $A_0, A_1, A_2, \ldots$ of all formulas of $\mathcal{L}_{\text{arith}}$ with one free variable x. Define the set

$$D := \{n \in \mathbb{N} \mid T \vdash A_n[\overline{n}/x]\}.$$

Since T is decidable, the set D is also decidable, as well as its complement $\overline{D} := \mathbb{N} \setminus D$. But T extends $\mathbf{Q}$, so by Lemma 5.6 $\overline{D}$ is representable in T by some formula of $\mathcal{L}_{\text{arith}}$ with one free variable x, say A_i. Now we have two possibilities:

- $i \in \overline{D}$: then, by representability, $T \vdash A_i[\overline{i}/x]$. But, at the same time, $i \in \overline{D}$ is equivalent to $i \notin D$ and so, by definition of D, $T \nvdash A_i[\overline{i}/x]$, a contradiction.
- $i \notin \overline{D}$: then, by representability, $T \vdash \neg A_i[\overline{i}/x]$. But, at the same time, $i \notin \overline{D}$ is equivalent to $i \in D$, so $T \vdash A_i[\overline{i}/x]$ by definition of D. So T would be inconsistent, against the hypothesis.

In both cases, we obtain a contradiction, so T must be undecidable. □

Remark 5.5 *The set $\overline{D}$ used in the proof is an example of a standard technique in logic and computer science, called* diagonalization*. First introduced by Cantor, it underlies countless arguments for showing negative results, such as Russel's paradox ("the set of all sets not containing themselves") or Turing's proof of the undecidability of the halting problem.*

The name "diagonalization" comes from the fact that a unary predicate is obtained from a binary one by identifying its two parameters: if (x, y) is a general coordinate in a bidimensional space, (x, x) designates a coordinate on the "diagonal" of the space. In the case of D, the two parameters that are identified are the index of the formula and the numeral substituted in it.

Apart from playing a fundamental role in the proof of the first incompleteness theorem, the above result has the following remarkable consequence:

Proposition 5.8 (undecidability of first-order logic) *Provability in first-order logic is undecidable. More specifically, the problem of deciding whether, for a given formula A, the sequent $\vdash A$ is provable in $\mathbf{LK}^1$ is in general undecidable.*

Proof. Observe that $\mathbf{Q}$ has finitely many axioms. Let us call R the conjunction of all of them. Then, given a formula C of $\mathcal{L}_{\text{arith}}$, $\mathbf{Q} \vdash C$ is equivalent to the provability of $R \vdash C$ in $\mathbf{LK}^1$, which is in turn equivalent to the provability of $\vdash R \Rightarrow C$, so we conclude by Lemma 5.7. □

5.5 The incompleteness theorems

Theorem 5.9 (Gödel's first incompleteness theorem) *Let T be a sound, recursively enumerable extension of $\mathbf{Q}$. Then, T is incomplete.*

Proof. Recall that soundness implies consistency (Lemma 5.5). So, as a consistent extension of $\mathbf{Q}$, T is undecidable by Lemma 5.7. Suppose now that T is complete. Then, T would be recursively enumerable, consistent and complete, hence decidable by Proposition 5.4, contradiction. □

Gödel's original proof of Theorem 5.9 is more complex than the one presented here. This higher complexity results in a more accurate result: Gödel explicitly

constructs a formula G_T such that, under the assumptions of Theorem 5.9, neither $T \vdash G_T$ nor $T \vdash \neg G_T$ holds. In a nutshell, this is done as follows:

- one starts by arithmetizing the language $\mathcal{L}_{\text{arith}}$, with the goal of defining formulas which speak of provability in $\mathbf{Q}$.
- In particular, one constructs a formula Thm with two free variables x, y such that $\mathbf{Q} \vdash \text{Thm}[\overline{n}/x, \overline{m}/y]$ precisely when n is the code of an $\mathbf{LK}^1$ proof of a sequent $\Gamma \vdash A$ with $\Gamma \subseteq \mathbf{Q}$ and the code of A is m.
- From the above, we may define $\text{Unprov} := \forall x.\neg\text{Thm}$, so that we have $\mathbf{Q} \vdash \text{Unprov}[\overline{m}/y]$ precisely when the formula whose code is m is unprovable in Robinson's arithmetic.
- One then proves the Fixpoint Lemma, or Diagonalization Lemma: for every formula A with one free variable y, there exists a sentence F (its fixpoint) such that $\mathbf{Q} \vdash F \Leftrightarrow A[\overline{\ulcorner F \urcorner}/y]$, where $\ulcorner F \urcorner$ denotes the code of F. In other words, the sentence F is equivalent to the fact that its own code (which is an integer) enjoys the property A, and such an equivalence is provable in $\mathbf{Q}$.
- At this point, one obtains the famous liar's paradox: if we consider the fixpoint $G_{\mathbf{Q}}$ of Unprov, we have $\mathbf{Q} \vdash G_{\mathbf{Q}} \Leftrightarrow \text{Unprov}[\overline{\ulcorner G_{\mathbf{Q}} \urcorner}/y]$, *i.e.*, $\mathbf{Q}$ proves that $G_{\mathbf{Q}}$ asserts its own unprovability in $\mathbf{Q}$. From this, modulo other technical lemmas, one concludes that neither $\mathbf{Q} \vdash G_{\mathbf{Q}}$ nor $\mathbf{Q} \vdash \neg G_{\mathbf{Q}}$ (there is a subtlety here: while the unprovability of $G_{\mathbf{Q}}$ only requires the consistency of $\mathbf{Q}$, the unprovability of $\neg G_{\mathbf{Q}}$ requires a stronger property, namely its ω-consistency. This mismatch was later fixed by Rosser: by complicating a bit the reasoning, it is possible to prove incompleteness relying on mere consistency. Our simpler proof, although weaker, has the advantage of bypassing this problem entirely).

Moreover, the machinery developed to construct the Gödel formula G_T also allows us, in case the theory is **PA** (or an extension of it), to find a formula $\text{Con}(\mathbf{PA}) := \forall x.\neg\text{Thm}[\overline{\ulcorner \bot \urcorner}/y]$ which expresses the consistency of **PA** itself. After some technical work, it is possible to show that $\mathbf{PA} \vdash \text{Con}(\mathbf{PA}) \Leftrightarrow G_{\mathbf{PA}}$, which leads to the second incompleteness theorem:

Theorem 5.10 (Gödel's second incompleteness theorem) *No sound, recursively enumerable extension of* **PA** *proves its own consistency.*

Let us mention a further benefit of Gödel's proof. So far we proved the existence of a sentence G such that neither G nor $\neg G$ are provable from, say, Peano arithmetic **PA** (our proof of Theorem 5.9 does not even do that: it is purely existential, it shows that such a G must exist without explicitly constructing it). In fact, we can do more: we can show that, between G and $\neg G$, it is the former which is true in the standard model. The explanation of this fact requires a couple of further technical definitions.

Definition 5.5 (bounded quantifiers) *We define* bounded quantifiers *as follows:*

$$\forall x < t.A := \forall x.x < t \Rightarrow A,$$
$$\exists x < t.A := \exists x.x < t \wedge A,$$

where t is an arbitrary term nor containing x.

We denote by Δ_0^0 the set of all formulas of $\mathcal{L}_{\text{arith}}$ whose quantifiers, if present, are all instances of bounded quantifiers. We then define the following sets of formulas:

$$\Sigma_1^0 := \{A \textit{ formula} \mid \mathbf{Q} \vdash A \Leftrightarrow \exists x_1 \dots \exists x_m . B, \textit{ with } B \in \Delta_0^0\};$$
$$\Pi_1^0 := \{A \textit{ formula} \mid \mathbf{Q} \vdash A \Leftrightarrow \forall x_1 \dots \forall x_m . B, \textit{ with } B \in \Delta_0^0\}.$$

Remark 5.6 *Observe that the negation of a Σ_1^0 formula is a Π_1^0 formula, and vice versa.*

The following result tells us that completeness holds for Σ_1^0 formulas:

Lemma 5.11 *If $A \in \Sigma_1^0$ and $\mathsf{Std} \models A$, then $\mathbf{Q} \vdash A$.*

Proof. One first proves that Δ_0^0 formulas are *primitive recursive*, *i.e.*, their truth in the standard model may be verified by means of a program only containing `for` loops. Intuitively, this is because a bounded quantifier only requires a number of verifications which is known in advance. Now, $A \in \Sigma_1^0$ means $\mathbf{Q} \vdash A \Leftrightarrow \exists x.B$ with $B \in \Delta_0^0$ (we have assumed for simplicity that there is only one quantifier, the argument does not change substantially in the general case). The fact that $\mathsf{Std} \models A$ implies that $\mathsf{Std} \models_{[x \mapsto n]} B$ for some $n \in \mathbb{N}$. But, since primitive recursive sets are in particular decidable, we may apply Lemma 5.6 and infer $\mathbf{Q} \vdash B[\overline{n}/x]$, from which we conclude by applying an existential rule in $\mathbf{LK}^1$. □

Now, the additional technical import of Gödel's proof is that Thm is a Δ_0^0 formula. Intuitively, this is because, once both a proof π and a formula A are fixed, verifying whether π proves A does not require more than `for` loops (everything is bounded by the size of π). Therefore, the Gödel formula G is Π_1^0 (it is equivalent to $\forall x.\neg\text{Thm}$). So, when in the incompleteness proof we say that $\mathbf{Q} \nvdash \neg G$, by Lemma 5.11 (applied contrapositively) we are saying that $\mathsf{Std} \not\models \neg G$, hence $\mathsf{Std} \models G$. This is actually a general rule, let us state it:

Lemma 5.12 *If $A \in \Pi_1^0$, then $\mathbf{Q} \nvdash A$ implies $\mathsf{Std} \models A$, i.e., an unprovable Π_1^0 formula must be true in the standard model.*

Lemma 5.12 holds in particular for the sentence stating the consistency of **PA**. Indeed, this is equivalent to $G_{\mathbf{PA}}$ and, more explicitly, may be stated as $\text{Con}(\mathbf{PA}) := \forall x.\neg\text{Thm}[\overline{\ulcorner \bot \urcorner}/y]$. Therefore, **PA** *is* consistent, in the sense that $\text{Con}(\mathbf{PA})$ is a "theorem" in the informal sense we gave in Sect. 5.2 (it is a sentence which is true in the intended model). However, such a theorem escapes **PA**.

Let us conclude by observing that, as we formulated them here, Gödel's incompleteness theorems apply only to arithmetical theories, *i.e.*, theories based on (an extension of) the language $\mathcal{L}_{\text{arith}}$. There are other interesting theories, such as Zermelo-Fraenkel set theory **ZF**, which are based on completely different languages[3] but for which we would like the incompleteness theorems to hold, because they intuitively encompass more mathematics than arithmetic. This is

[3]The language of **ZF** is extremely simple, it consists of only one binary predicate symbol $\in$, together with the usual equality predicate $=$.

possible by showing that **Q** or **PA** may be *encoded* in the desired theory: for instance, all the function and predicate symbols of $\mathcal{L}_{\text{arith}}$ may be encoded in the language of **ZF** in such a way that the (encoding of the) axioms of **Q** or **PA** become provable in **ZF**. At that point, it is clear that the incompleteness theorems apply to such non-arithmetical theories as well, thereby revealing the surprising breadth of Gödel's results: *any* theory which claims to be a "theory of mathematics" will contain enough arithmetic to fall into the incompleteness trap.

Chapter 6

Cut Elimination

The following theorem, which is arguably the central result of proof theory, was proved by Gentzen in 1934. Let us call *cut-free* $\mathbf{LK}^1$ the calculus $\mathbf{LK}^1$ without the cut rule.

Theorem 6.1 (cut-elimination) *The* cut *rule is admissible in cut-free* $\mathbf{LK}^1$.

What is especially interesting about Theorem 6.1 is that its proof is constructive, *i.e.*, it gives a procedure for gradually transforming a proof with cuts into a proof without cuts.

First of all, we need some terminology.

Definition 6.1 (principal formula, context) *By observing Fig. 2.2 and Fig. 3.2, we see that the conclusion of each logical rule is always of the form* $\vdash \Sigma, F$*, where* F *is the occurrence of formula introduced by the rule. We say that* Σ *and* F *are the* context *and the* principal *formula of the rule, respectively.*

In the ax *rule, both* $\neg A$ *and* A *are called principal (and the context is empty). Dually, in a* cut *rule, there is no principal formula.*

We also stipulate that no formula is principal in structural rules.

Definition 6.2 (principal ancestor) *Let* δ *be a derivation in* $\mathbf{LK}^1$ *and let* $\vdash \Gamma, A$ *be a sequent appearing in* δ*, which is the conclusion of an instance of rule* R *such that* F *is not principal (note that, under this hypothesis,* R *cannot be* ax*). Then, we define the* immediate ancestors *of* F *according to the nature of* R*, as follows:*

- *if* R *is logical or* cut*, by inspecting the rules we see that there is exactly one occurrence of formula in the premise(s) of* R *corresponding to* F*; then, we take that occurrence to be the only immediate ancestor of* F*.*
- *If* R *is a weakening, there are two mutually exclusive cases:*
  - F *is the occurrence introduced by the rule; then* F *has no immediate ancestor;*
  - *otherwise, there is a unique occurrence in the premise corresponding to* F*, which is taken to be its only immediate ancestor.*
- *If* R *is a contraction, there are two mutually exclusive cases:*

– *F is the occurrence which is contracted by the rule; then, both occurrences of F in the premise are immediate ancestors of F;*

– *otherwise, there is a unique occurrence in the premise corresponding to F, which is taken to be its only immediate ancestor.*

We say that an occurrence A_n of formula is an ancestor *of an occurrence A_0 of the same formula if there exist $A_1, \ldots, A_{n-1}$ such that, for all $0 \leq i < n$, A_{i+1} is an immediate ancestor of A_i (i.e., the ancestor relation is the reflexive-transitive closure of the immediate ancestor relation). An ancestor is* principal *if it is the principal formula of the rule of which it is conclusion.*

Remark 6.1 *Because of the contraction rule, an occurrence of formula may have more than one principal ancestor. Because of the weakening rule, it may have none.*

Definition 6.3 (taxonomy of cuts) *Consider an instance of* cut *rule*

$$\dfrac{\dfrac{\vdots}{\vdash \Gamma, \neg A}\ \mathsf{R}_1 \qquad \dfrac{\vdots}{\vdash \Delta, A}\ \mathsf{R}_2}{\vdash \Gamma, \Delta}\ \mathsf{cut}$$

We say that A and $\neg A$ are the cut-formulas. *Such an instance of* cut *is*

principal: *if both cut-formulas are principal (for R_1 and R_2);*

simply commutative: *if exactly one of the cut-formulas is principal;*

doubly commutative: *if none of the cut-formulas is principal.*

Simply and doubly commutative cuts are collectively called just commutative.

Recall that a *multiset* on a set X is a function $\mu : X \longrightarrow \mathbb{N}$. Intuitively, it is a "set with reptitions". Indeed, a usual subset $S \subseteq X$ may be seen as a function $S : X \longrightarrow \{0, 1\}$: an element of X either is or is not in S. For a multiset μ, an element may be in μ with a multiplicity higher than 1. We will only use finite multisets, which are such that $\mu(x) = 0$ for all but finitely many $x \in X$.

Definition 6.4 (weight of a proof) *The* logical size *of a formula A, denoted by $|A|$, is defined inductively as follows:*

$$\begin{aligned} |P(t_1, \ldots, t_n)| := |\overline{P}(t_1, \ldots, t_n)| &:= 1 \\ |\top| := |\bot| &:= 1 \\ |A \wedge B| := |A \vee B| &:= |A| + |B| + 1 \\ |\forall x.A| := |\exists x.A| &:= |A| + 1 \end{aligned}$$

Observe that $|A| = |\neg A|$.

The weight *of an instance of* cut *rule whose principal formula is A is the pair $(|A|, i)$, where $i = 0$ if the cut is principal, $i = 1$ if the cut is simply commutative and $i = 2$ if the cut is doubly commutative.*

Let π be a proof of $\mathbf{LK}^1$. The weight *of π, denoted by $\mu(\pi)$, is the* multiset *composed of all weights of all instances of* cut *rule of π.*

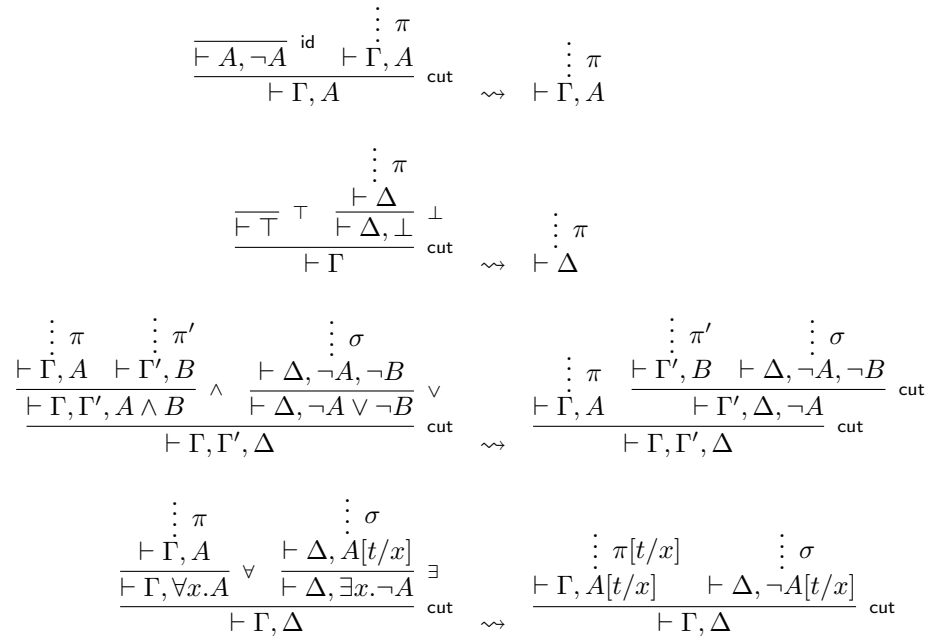


Figure 6.1: Elimination of principal cuts.

Note that the weight of a cut is an element of $\mathbb{N} \times \{0, 1, 2\}$, so the weight of a proof is an element of $\mathbb{N}^{(\mathbb{N}\times\{0,1,2\})}$, *i.e.*, a function $\mathbb{N} \times \{0, 1, 2\} \longrightarrow \mathbb{N}$ which is almost everywhere null. This means that weights may be seen as ordinals strictly smaller than $\omega^{3\cdot\omega} = \omega^\omega$. Formally, the proof of Theorem 6.1 is an induction on that ordinal. Indeed, we will show that, given a proof π of $\vdash \Gamma$, either $\alpha(\pi) = []$ (the empty multiset, or the everywhere-zero function), in which case π is cut-free by definition, or $\alpha(\pi) > 0$, in which case we will show that we may find another proof π' of $\vdash \Gamma$ such that $\alpha(\pi') < \alpha(\pi)$.

So, let π be a proof of $\vdash \Gamma$ containing at least one instance of cut. Then, since π is a finite tree, there is necessarily a *maximal* cut, *i.e.*, a cut of the form

$$\dfrac{\vdots\ \sigma \qquad \vdots\ \delta}{\vdash \Sigma, \neg A \quad \vdash \Delta, A}\ \mathsf{cut}$$

such that σ and δ are cut-free. At this point, we operate according to the nature of such a cut:

principal: if the cut is principal, we locally apply one of the transformations of Fig. 6.1, obtaining a proof π' of $\vdash \Gamma$ (the conclusion is obviously left unchanged by the transformations);

simply commutative: if the cut is simply commutative, we may suppose without loss of generality that A is principal in $\vdash \Delta, A$; then we find all the principal ancestors of $\not{A}$ in σ, let us say there are n of them, belonging to sequents of the form $\vdash \Phi_1, \neg A, \ldots, \vdash \Phi_n, \neg A$; at this point, we

"insert" n copies of δ in σ by means of n instances of cut:

$$\dfrac{\begin{array}{cc}\vdots & \vdots\ \delta \\ \vdash \Phi_i, \neg A & \vdash \Delta, A\end{array}}{\vdash \Phi_i, \Delta}$$

This operation produces a proof of $\vdash \Sigma, \Delta, \ldots, \Delta$, where there are n copies of Δ; from this, we obtain $\vdash \Sigma, \Delta$ by means of structural rules, which gives us a proof π' of $\vdash \Gamma$ (again, the conclusion of the proof is obviously left unchanged by this whole operation);

doubly commutative: if the cut is doubly commutative, we choose arbitrarily between A and $\neg A$ and proceed as above.

It is now easy to check that, in all cases, $\alpha(\pi') < \alpha(\pi)$:

principal: by inspection of Fig. 6.1;

simply commutative: we observe that the new cuts introduced are on the same cut-formulas as the original one, but they are all principal, so we transform 1 cut of weight $(|A|, 1)$ into n cuts of weight $(|A|, 0)$, which is strictly smaller; it is important to observe that no other cut is introduced, thanks to the hypothesis of maximality of the original cut;

doubly commutative: we observe that the new cuts introduced are an the same cut-formulas as the original one, but they are all simply commutative, so we transform 1 cut of weight $(|A|, 2)$ into n cuts of weight $(|A|, 1)$, which is strictly smaller; the same remark on the maximality of the original cut holds here.

Remark 6.2 *It is important to stress the* modularity *of the above proof. As long as a pair of dual connectives has a principal cut-elimination rule which makes the logical size decrease, the proof applies unchanged.*

In fact, the same proof may be applied to the two-sided formulation of **LK**. *In that case, instead of having a principal cut-elimination rule for each pair of dual connectives, we have one rule for each connective: indeed, a two-sided principal cut relates an occurrence of A on the right with an occurrence of A on the left, with both occurrences being introduced by the right and left logical rules for the main connective of A. In Sect. 7.1 we will give a specialized version of such rules for* **LJ**, *the intuitionistic sequent calculus. From those, it is immediate to find the general rules for two-sided* **LK**.

The cut-elimination theorem may appear to be somewhat surprising, especially in view of the central role that the cut rule plays in our proof of the completeness theorem (Chapter 4). In fact, there are completeness proofs which apply directly to cut-free $\mathbf{LK}^1$, so this impression is misleading. However, there is a technical way of saying that cut-elimination does indeed have a highly non-trivial meaning, as follows.

Definition 6.5 (subformula property) *We say that a sequent calculus rule of the form*

$$\dfrac{\Gamma_1 \vdash \Delta_1 \quad \ldots \quad \Gamma_n \vdash \Delta_n}{\Gamma' \vdash \Delta'}$$

satisfies the subformula property *if every formula appearing in* $\Gamma_1, \ldots, \Gamma_n, \Delta_1, \ldots, \Delta_n$ *is a subformula of a formula appearing in* $\Gamma' \vdash \Delta'$.

A derivation satisfies the subformula property if every rule in it satisfies the subformula property.

It is enough to inspect the rules of $\mathbf{LK}^1$ (both in its two-sided and one-sided formulation) to see that all rules satisfy the subformula property except cut. Therefore, the cut-elimination theorem says that every $\mathbf{LK}^1$ proof may be transformed into a proof enjoying the subformula property. Note that such a proof only ever mentions subformulas of its conclusion, *i.e.*, to prove $\vdash A$ we only need to work on the subformulas of A. This is also known as an *analytic proof*.

The existence of analytic proofs is a non-trivial fact: when mathematicians face a difficult problem, *i.e.*, a formula they cannot prove with known methods, it is common practice to seek new methods, often involving objects and constructions which are far removed from those mentioned by the original formula (think of analytic number theory, which uses complex analysis to prove statements about natural numbers). The existence of analytic proofs means that, in principle, such detours are useless: every theorem has a proof using only the concepts mentioned in the statement of the theorem.

Another view of the subformula property is obtained by considering proof search, in which sequent calculus rules are read bottom-up. Indeed, this is the direction in which they are applied by mathematicians: only a god would be able to build a proof top-down starting from exactly the axioms necessary to prove the conclusion. We poor human beings have the conclusion and (desperately, sometimes) look for a premise from which it might come, with the goal of reaching one day an axiom. From this bottom-up perspective, the subformula property assures us that we need not, so to speak, invent anything: the formulas we will deal with at the next step of the search are all subformulas of the formulas we already have. This restricts immensely the search space.

It is fair to say that the subformula property gives us a precise way of saying that *the cut rule is the only intelligent rule of sequent calculus*. It is the only rule that requires creativity: when read bottom-up, the cut rule introduces a (pair of dual) formula(s) which may have nothing to do whatsoever with the formulas we are trying to prove. The only clue that guides us in the choice of such a formula is the intuition that it will help us reach our goal. We may therefore read cut-elimination as saying that "creativity is useless", which is indeed a quite non-trivial fact.

So, is creativity useless? Is it useless to look into sophisticated mathematical theories for finding a proof of an elementary fact? Quantitative considerations give us a clear negative answer: Statman and Orevkov showed that there exist sequences of formulas $F_0, F_1, F_2, \ldots$, such that the logical size of F_n is $\Theta(n)$ and such that each $\vdash F_n$ admits a proof in $\mathbf{LK}^1$ of size $\Theta(n)$, such that the shortest cut-free proof of $\vdash F_n$ has size hyperexponential in n (the hyperexponential function, also known as *tetration*, is defined recursively by $h(0) := 1$, $h(n+1) := 2^{h(n)}$). Already for $n = 5$ the size of such a proof greatly surpasses the number of atoms in the universe.

The above result implies in particular that cut-elimination is, in general, an *unfeasible algorithm*, *i.e.*, its complexity is too big to be of any practical use. Therefore, although *in principle* analytic proofs always exist and we have an

effective procedure for converting an arbitrary proof into an analytic proof, in practice it is much more convenient to use cuts! This, in a way, may be seen as the "revenge of creativity": being stupid gets you as far as being smart, but you'd better be ready to waste a lot of time...

Chapter 7

Intuitionistic Logic

Intuitionistic logic arises from exquisitely proof-theoretic considerations: we are not merely interested in whether a formula is provable (*i.e.*, in its validity) but in *how* it is provable.

The distinguishing characteristic of intuitionistic logic is conferring a stronger meaning to disjunction and existential quantification:

Definition 7.1 (disjunction and existence properties) *We say that a proof system satisfies the*

disjunction property *if the provability of $A \vee B$ implies the provability of A or the provability of B;*

existence property *if the provability of $\exists x.A$ implies the provability of $A[t/x]$ for some term t (called* witness *of A).*

In classical logic, none of the above properties holds. The failure of the disjunction property is blatantly given by the law of excluded middle: $\vdash A \vee \neg A$ is always provable in **LK**, but in case A is a satisfiable formula also admitting a countermodel (such as a propositional constant), by validity of **LK** none of $\vdash A$ and $\vdash \neg A$ will be provable. The failure of the existential property is exemplified by the so-called *drinker's formula*:

$$F := \exists x.(D(x) \Rightarrow \forall y.D(y)).$$

If we interpret $D(z)$ as meaning "z is drinking", then F is saying that, in a bar, there is always someone such that, if he or she is drinking, then everyone is drinking. The formula F is true in all first-order structures (we invite the reader to check this), hence provable in $\mathbf{LK}^1$ (we invite the reader to find such a proof), and yet we have

Lemma 7.1 *There is no term t such that $\vdash D(t) \Rightarrow \forall y.D(y)$ is provable in $\mathbf{LK}^1$.*

Proof. This is an application of cut-elimination and the reversibility of the rules ($\vdash\Rightarrow$) and ($\vdash \forall$), thanks to which we may assume that a proof of the above formula must end as follows:

$$\dfrac{\dfrac{D(t) \vdash D(y)}{D(t) \vdash \forall y.D(y)}}{\vdash D(t) \Rightarrow \forall y.D(y)}$$

which implies that y is not free in t, in particular $t \neq y$, so the top sequent is unprovable. □

Example 7.1 *Another often cited example of classical existential proof giving no explicit witness is the following:*

Fact. *There exists a pair of irrational numbers (x, y) such that $x^y \in \mathbb{Q}$.*

PROOF. *Let $a := \sqrt{2}^{\sqrt{2}}$. We have two possibilities:*

- *$a \notin \mathbb{Q}$, i.e., a is irrational: in that case, the pair $(a, \sqrt{2})$ satisfies the statement (recall that $\sqrt{2}$ is irrational);*
- *$a \in \mathbb{Q}$: in that case, the pair $(\sqrt{2}, \sqrt{2})$ satisfies the statement.*

In both cases, the statement is satisfied, so we may conclude. □

The above proof gives us two possible pairs of irrational numbers satisfying the statement, namely $(\sqrt{2}^{\sqrt{2}}, \sqrt{2})$ and $(\sqrt{2}, \sqrt{2})$, but it gives us no hint whatsoever as to which one of these may be taken as an actual witness of the statement. This is an example of non-constructive *proof.*

The non-constructivity of the above proof is actually quite mild: at least we have two possible candidates which are given explicitly. Many important theorems of standard mathematics (typically, in analysis) are non-constructive in a much more violent way: think of a basis for the vector space of all functions from $\mathbb{R}$ to itself; we know that it must exists, but how in the world would we define it explicitly?

Remark 7.1 *From now on, it will make little sense to appeal to the idea of truth and validity of a formula. Indeed, intuitionistic logic refines classical logic in a way which is not primarily related to truth values (as, for instance, many-valued logics or fuzzy logics do).*

The right way to look at intuitionistic logic is as a proof system forcing us to see differences that classical logic ignores. Typically, it distinguishes between the validity of a formula *A and the* impossibility of *A* being invalid. *Such a distinction is obviously meaningless if we only look at truth values but it makes perfect sense if we look at proofs of A: in the first case, we found a direct proof of $\vdash A$; in the second case, we proved $\vdash A$ from a proof of $\vdash \neg\neg A$.*

More concretely, a constructive proof of the statement in Example 7.1 would satisfy the existence property and, when formalized in sequent calculus, would end with two existential rules

$$\frac{\vdash \mathrm{Irrat}(t) \wedge \mathrm{Irrat}(u) \Rightarrow \mathrm{Rat}(t^u)}{\vdash \exists x.\exists y.\mathrm{Irrat}(x) \wedge \mathrm{Irrat}(y) \Rightarrow \mathrm{Rat}(x^y)}$$

for some explicit terms t, u. Instead, the proof given above uses excluded middle which, as we will see momentarily, is equivalent to the principle $\neg\neg A \Rightarrow A$, so the formalization of that proof does not *end with two existential rules. A similar remark applies to the proof of the drinker's formula.*

Identity rules:

$$\frac{}{A \vdash A}\ \mathsf{id} \qquad\qquad \frac{\Gamma \vdash A \quad \Gamma', A \vdash \Sigma}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut}$$

Structural rules:

$$\frac{\Gamma \vdash \Sigma}{\Gamma, A \vdash \Sigma}\ \mathsf{w}\vdash \qquad\qquad \frac{\Gamma \vdash}{\Gamma \vdash A}\ \vdash\mathsf{w}$$

$$\frac{\Gamma, A, A \vdash \Sigma}{\Gamma, A \vdash \Sigma}\ \mathsf{c}\vdash \qquad\qquad \text{(no contraction to the right)}$$

Logical rules:

$$\frac{\Gamma \vdash A \quad \Gamma', B \vdash \Sigma}{\Gamma, \Gamma', A \Rightarrow B \vdash \Sigma}\ \Rightarrow\vdash \qquad\qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ \vdash\Rightarrow$$

$$\text{(no rule for } \top \text{ to the left)} \qquad\qquad \frac{}{\Gamma \vdash \top}\ \vdash\top$$

$$\frac{\Gamma, A_i \vdash \Sigma}{\Gamma, A_1 \wedge A_2 \vdash \Sigma}\ \wedge\vdash_i,\ i\in\{1,2\} \qquad\qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \vdash\wedge$$

$$\frac{}{\Gamma, \bot \vdash \Sigma}\ \bot\vdash \qquad\qquad \text{(no rule for } \bot \text{ to the right)}$$

$$\frac{\Gamma, A \vdash \Sigma \quad \Gamma, B \vdash \Sigma}{\Gamma, A \vee B \vdash \Sigma}\ \vee\vdash \qquad\qquad \frac{\Gamma \vdash A_i}{\Gamma \vdash A_1 \vee A_2}\ \vdash\vee_i\ i\in\{1,2\}$$

$$\frac{\Gamma, A[t/x] \vdash \Sigma}{\Gamma, \forall x.A \vdash \Sigma}\ \forall\vdash \qquad\qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x.A}\ \vdash\forall\ x\notin\mathrm{fv}(\Gamma)$$

$$\frac{\Gamma, A \vdash \Sigma}{\Gamma, \exists x.A \vdash \Sigma}\ \exists\vdash\ x\notin\mathrm{fv}(\Gamma) \qquad\qquad \frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x.A}\ \vdash\exists$$

Figure 7.1: Intuitionistic sequent calculus $\mathbf{LJ}^1$. The multiset Σ contains at most one occurrence of formula.

7.1 Sequent calculus

Intuitionistic sequent calculus, denoted by **LJ** (or $\mathbf{LJ}^1$ if we want to stress the presence of first-order quantifiers), is defined by restricting the rules of **LK** to operate on a restricted class of sequents:

Definition 7.2 (intuitionistic sequent) *A sequent $\Gamma \vdash \Sigma$ is* intuitionistic *if Σ contains at most one formula. The rules of sequent calculus for intuitionistic logic, denoted by* **LJ**, *are presented in Fig. 7.1.*

Some comments about Fig. 7.1:

contraction: the most important structural modification imposed by the restriction to intuitionistic sequents is that contraction to the right is for-

bidden: indeed, the rule

$$\frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}$$

would be impossible to apply because the sequent in the premise is not intuitionistic, even if Δ is empty.

Implication: in intuitionistic logic, implication is a primitive connective, *i.e.*, it cannot be defined as $A \Rightarrow B := \neg A \vee B$. This is why its rules are included in **LJ**.

Negation: as anticipated above, a prominent difference between intuitionistic and classical logic is the behavior of negation, which (among other things) is no longer involutive. This, in particular, prevents us from using a De Morgan definition of negation (in fact, De Morgan laws too fail in intuitionistic logic). However, we may still make some syntactic economy and, instead of considering negation as a primitive connective, we define it as

$$\neg A \quad := \quad A \Rightarrow \bot.$$

We invite the reader to check that the definition is classically sound, *i.e.*, that the equivalence between the above formulas may be derived in **LK**.

Additive formulation of logical rules: with respect to Fig. 2.1, we used the *additive formulation* of the logical rules. For disjunction, the choice is justified by observing that the rule

$$\frac{\Gamma \vdash \Delta, A, B}{\Gamma \vdash \Delta, A \vee B}$$

would be impossible to apply because the sequent in the premise is not intuitionistic. Therefore, if we want to be able to prove disjunctive formulas in **LJ** (which we obviously do!), we need to shift to the additive formulation, knowing anyway that it is equivalent to the multiplicative formulation used in Fig. 2.1, so it makes no difference classically.

As for conjunction, we could in principle use the multiplicative formulation, which adapts seamlessly to intuitionistic sequents:

$$\frac{\Gamma, A, B \vdash \Sigma}{\Gamma, A \wedge B \vdash \Sigma} \qquad \frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \wedge B}$$

However, making the same choice as for the dual connective makes the formulation of cut-elimination more harmonious (see below).

The same holds for truth and falsehood: their multiplicative formulation poses no problems (we give it below) but, to be more uniform, we prefer to formulate these rules too in the additive style.

$$\frac{\Gamma \vdash \Sigma}{\Gamma, \top \vdash \Sigma} \qquad \frac{}{\vdash \top}$$

$$\frac{}{\bot \vdash} \qquad \frac{\Gamma \vdash}{\Gamma \vdash \bot}$$

Quantifiers: the quantifier rules are unchanged: they are just adaptation to intuitionistic sequents of the classical rules.

Example 7.2 (rules for negation) *The following rules are derivable:*

$$\frac{\Gamma \vdash A}{\Gamma, \neg A \vdash \bot}\ \neg\vdash \qquad\qquad \frac{\Gamma, A \vdash \bot}{\Gamma \vdash \neg A}\ \vdash\neg$$

Indeed, recalling that $\neg A := A \Rightarrow \bot$*, they are special cases of the* $(\Rightarrow\vdash)$ *and* $(\vdash\Rightarrow)$ *rules, respectively.*

Let us move on to the central result of (intuitionistic) proof theory:

Theorem 7.2 *Cut-elimination holds in* **LJ**.

The proof is essentially identical to that of Theorem 6.1. We only need to define the cut-elimination rules for principal cuts, which are as follows:

$$\frac{\dfrac{}{A \vdash A}\ \mathsf{id} \qquad \begin{matrix}\vdots\ \pi\\ \Gamma, A \vdash \Sigma\end{matrix}}{\Gamma, A \vdash \Sigma}\ \mathsf{cut} \quad \rightsquigarrow \quad \begin{matrix}\vdots\ \pi\\ \Gamma, A \vdash \Sigma\end{matrix}$$

$$\frac{\dfrac{\begin{matrix}\vdots\ \pi\\ \Gamma, A \vdash B\end{matrix}}{\Gamma \vdash A \Rightarrow B}\ \vdash\Rightarrow \qquad \dfrac{\begin{matrix}\vdots\ \sigma\\ \Gamma' \vdash A\end{matrix} \qquad \begin{matrix}\vdots\ \tau\\ \Gamma'', B \vdash \Sigma\end{matrix}}{\Gamma', \Gamma'', A \Rightarrow B \vdash \Sigma}\ \Rightarrow\vdash}{\Gamma, \Gamma', \Gamma'' \vdash \Sigma}\ \mathsf{cut} \quad \rightsquigarrow \quad \frac{\dfrac{\begin{matrix}\vdots\ \sigma\\ \Gamma' \vdash A\end{matrix} \qquad \begin{matrix}\vdots\ \pi\\ \Gamma, A \vdash B\end{matrix}}{\Gamma, \Gamma' \vdash B}\ \mathsf{cut} \qquad \begin{matrix}\vdots\ \tau\\ \Gamma'', B \vdash \Sigma\end{matrix}}{\Gamma, \Gamma', \Gamma'' \vdash \Sigma}\ \mathsf{cut}$$

$$\frac{\dfrac{\begin{matrix}\vdots\ \sigma_1\\ \Gamma \vdash A_1\end{matrix} \qquad \begin{matrix}\vdots\ \sigma_2\\ \Gamma \vdash A_2\end{matrix}}{\Gamma \vdash A_1 \wedge A_2}\ \vdash\wedge \qquad \dfrac{\begin{matrix}\vdots\ \pi\\ \Gamma', A_i \vdash \Sigma\end{matrix}}{\Gamma', A_1 \wedge A_2 \vdash \Sigma}\ \wedge\vdash_i}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut} \quad \rightsquigarrow \quad \frac{\begin{matrix}\vdots\ \sigma_i\\ \Gamma \vdash A_i\end{matrix} \qquad \begin{matrix}\vdots\ \pi\\ \Gamma', A_i \vdash \Sigma\end{matrix}}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut}$$

$$\frac{\dfrac{\begin{matrix}\vdots\ \pi\\ \Gamma \vdash A_i\end{matrix}}{\Gamma \vdash A_1 \vee A_2}\ \vdash\vee_i \qquad \dfrac{\begin{matrix}\vdots\ \sigma_1\\ \Gamma', A_1 \vdash \Sigma\end{matrix} \qquad \begin{matrix}\vdots\ \sigma_2\\ \Gamma', A_2 \vdash \Sigma\end{matrix}}{\Gamma', A_1 \vee A_2 \vdash \Sigma}\ \vee\vdash}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut} \quad \rightsquigarrow \quad \frac{\begin{matrix}\vdots\ \pi\\ \Gamma \vdash A_i\end{matrix} \qquad \begin{matrix}\vdots\ \sigma_i\\ \Gamma', A_i \vdash \Sigma\end{matrix}}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut}$$

$$\frac{\dfrac{\begin{matrix}\vdots\ \pi\\ \Gamma \vdash A\end{matrix}}{\Gamma \vdash \forall x.A}\ \vdash\forall \qquad \dfrac{\begin{matrix}\vdots\ \sigma\\ \Gamma', A[t/x] \vdash \Sigma\end{matrix}}{\Gamma', \forall x.A \vdash \Sigma}\ \forall\vdash}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut} \quad \rightsquigarrow \quad \frac{\begin{matrix}\vdots\ \pi[t/x]\\ \Gamma \vdash A[t/x]\end{matrix} \qquad \begin{matrix}\vdots\ \sigma\\ \Gamma', A[t/x] \vdash \Sigma\end{matrix}}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut}$$

$$\frac{\dfrac{\begin{matrix}\vdots\ \sigma\\ \Gamma \vdash A[t/x]\end{matrix}}{\Gamma \vdash \exists x.A}\ \vdash\exists \qquad \dfrac{\begin{matrix}\vdots\ \pi\\ \Gamma', A \vdash \Sigma\end{matrix}}{\Gamma', \exists x.A \vdash \Sigma}\ \exists\vdash}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut} \quad \rightsquigarrow \quad \frac{\begin{matrix}\vdots\ \sigma\\ \Gamma \vdash A[t/x]\end{matrix} \qquad \begin{matrix}\vdots\ \pi[t/x]\\ \Gamma', A[t/x] \vdash \Sigma\end{matrix}}{\Gamma, \Gamma' \vdash \Sigma}\ \mathsf{cut}$$

(Note that, with the additive formulation, there can be no principal cut whose main formula is $\top$ or $\bot$, because each of these constants lacks a rule on one side, the left and right, respectively).

Every result we prove in the rest of the chapter is a consequence of cut-elimination. In particular, we will use cut-elimination to prove that **LJ** does indeed define a constructive logic, *i.e.*, it is a proof system in which both the disjunction and existence properties are verified.

Let us first take a detour into consistency. In the classical case, the consistency of $\mathbf{LK}^1$ is an immediate corollary of its validity: there exist invalid formulas, hence by validity there exist unprovable formulas, hence $\vdash$ is not provable in $\mathbf{LK}^1$ (for otherwise *every* formula would be provable). So, as far as we are concerned, cut-elimination is not strictly needed to show consistency.

In the case of intuitionistic logic, there is a truth semantics in terms of *Kripke models*, with respect to which $\mathbf{LJ}^1$ may be shown to be valid and complete. The description of such a semantics is beyond the scope of these notes. We will content ourselves with inviting the reader to observe that it must be quite different from the usual Boolean truth semantics, because $\neg P \vee P$ is not intuitionistically valid. So, in absence of a truth semantics, cut-elimination becomes our only way to consistency:

Proposition 7.3 (consistency of $\mathbf{LJ}^1$) *The empty sequent $\vdash$ is not provable in $\mathbf{LJ}^1$.*

Proof. Simply observe that $\vdash$ is not cut-free-provable: no rule except cut allows to infer the empty sequent. But then, by cut-elimination, $\vdash$ is not provable at all. □

Let us move on towards the disjunction and existence properties.

Definition 7.3 (strictly positive subformula) *A* strictly positive context *is defined by the following grammar:*

$$S ::= \langle\cdot\rangle \mid \top \mid S \wedge A \mid A \wedge S \mid \bot \mid S \vee A \mid A \vee S \mid A \Rightarrow S \mid \forall x.S \mid \exists x.S$$

The special constant $\langle\cdot\rangle$ is called the hole *of the context. Note that every context contains at most one occurrence of the hole. We denote by $S\langle A\rangle$ the formula obtained by substituting the formula A for the hole in S.*

A subformula occurrence B of A is strictly positive *if $A = S\langle B\rangle$ for a strictly positive context S.*

Theorem 7.4 *1. If Γ contains no strictly positive disjunctive subformulas, then $\Gamma \vdash A \vee B$ provable in **LJ** implies $\Gamma \vdash A$ or $\Gamma \vdash B$ provable in **LJ**;*

*2. if Γ contains no strictly positive existential subformulas, then $\Gamma \vdash \exists x.A$ provable in **LJ** implies $\Gamma \vdash A[t_1/x] \vee \cdots \vee A[t_n/x]$ provable in **LJ**, for some terms $t_1, \ldots, t_n$;*

*3. if Γ contains no strictly positive existential or disjunctive subformula, then $\Gamma \vdash \exists x.A$ provable in **LJ** implies $\Gamma \vdash A[t/x]$ provable in **LJ** for some term t.*

Proof. Let us start with point 1. We proceed by induction on the last rule of the proof of $\Gamma \vdash A \vee B$. By cut-elimination, we may suppose that such a rule is never a cut. It cannot be an identity rule or a $(\vee \vdash)$ rule either, for otherwise a disjunction would appear strictly positively in Γ. It cannot be a right logical rule

other than ($\vdash \vee_i$), for obvious reasons (the principal connective of the formula on the right is a disjunction).

Now, a simple inspection of the left logical rules and the structural rules other than ($\vdash \mathsf{w}$) shows that exactly one of their premise will be of the form $\Gamma' \vdash A \vee B$ with Γ' containing no strictly positive disjunctive subformula, so we may conclude by induction.

We are therefore left with two cases:

- the last rule is ($\vdash \mathsf{w}$): then $\Gamma \vdash$ is provable, from which both $\Gamma \vdash A$ and $\Gamma \vdash B$ are provable, as required;
- the last rule is ($\vdash \vee_i$), in which case the premise is $\Gamma \vdash A$ (if $i = 1$) or $\Gamma \vdash B$ (if $i = 2$), as desired.

Point 2 is proved in a similar way. The only difference is that now it is possible for the last rule to be ($\vee \vdash$), in which case we have a proof of $\Gamma', B \vdash \exists x.A$ and a proof of $\Gamma', C \vdash \exists x.A$, where $\Gamma = \Gamma', B \vee C$. From these, the induction hypothesis gives us a proof π of $\Gamma', B \vdash F$ and a proof σ of $\Gamma', C \vdash G$ with $F = A[u_1/x] \vee \cdots \vee A[u_m/x]$ and $G = A[v_1/x] \vee \cdots \vee A[v_n/x]$, so we may obtain a proof of a sequent of the desired form as follows:

$$\dfrac{\dfrac{\dfrac{\vdots\ \pi}{\Gamma', B \vdash F}}{\Gamma', B \vdash F \vee G}\ \vee\vdash_1 \qquad \dfrac{\dfrac{\vdots\ \sigma}{\Gamma', B \vdash G}}{\Gamma', C \vdash F \vee G}\ \vee\vdash_2}{\Gamma \vdash F \vee G}\ \vdash\vee$$

Point 3 is proved as point 2, except that now it is no longer possible to have ($\vee \vdash$) as a last rule, which implies that induction applies to exactly one premise, obtaining the stated result. □

Corollary 7.5 (disjunction and existence properties) *The calculus* **LJ**^{1} *enjoys both the disjunction and the existence property.*

Proof. The disjunction and existence properties involve provability in an empty context, so we simply apply Theorem 7.4 with Γ empty, which verifies in particular the conditions required by point 1 (which gives the disjunction property) and point 3 (which gives the existence property). □

7.2 The relationship between intuitionistic and classical logic

As we defined it, intuitionistic logic looks like a restriction of classical logic. Although we will see that such a viewpoint is misleading, let us start with confirming this first impression by giving some examples of formulas which are provable in **LK** but not in **LJ**. We will need the following observation:

Lemma 7.6 *If* $\Gamma \vdash$ *is provable in* **LJ**, *then* Γ *contains the subformula* $\bot$.

Proof. By induction on the last rule of the proof of $\Gamma \vdash$. □

Example 7.3 (excluded middle) *This is the typical classical principle which is refused by intuitionistic logic. Take a propositional constant P. Then, by the disjunction property, if $\vdash \neg P \vee P$ were provable in* **LJ***, so would be one of $\vdash \neg P$ or $\vdash P$. The latter is obviously impossible (no non-cut rule concludes with a sequent containing a single propositional constant). By reversibility of the ($\vdash\Rightarrow$) rule, a proof of the former would give a proof of $P \vdash \bot$, from which by a cut with $\bot \vdash$ (which is provable by means of a ($\bot \vdash$) rule) we obtain the provability of $P \vdash$, which is impossible by Lemma 7.6.*

By contrast, the *double negation* of excluded middle is intuitionistically provable (we use the rules for negation introduced in Example 7.2):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\mathsf{id}}{A \vdash \neg A \vee A}\,\vdash\vee_2}{\neg(\neg A \vee A), A \vdash \bot}\,\neg\vdash}{\neg(\neg A \vee A) \vdash \neg A}\,\vdash\neg}{\neg(\neg A \vee A) \vdash \neg A \vee A}\,\vdash\vee_1}{\neg(\neg A \vee A), \neg(\neg A \vee A) \vdash \bot}\,\neg\vdash}{\neg(\neg A \vee A) \vdash \bot}\,\mathsf{c}\vdash}{\vdash \neg\neg(\neg A \vee A)}\,\vdash\neg$$

Also note that a subproof of the above proof shows the provability of

$$\neg(\neg A \vee A) \Rightarrow (\neg A \vee A).$$

We will use these facts in the following example.

Example 7.4 (proof by contradiction) *The quintessential form of classical reasoning which is forbidden in intuitionistic logic is the proof by contradiction. This may be stated as the principle $\neg\neg A \Rightarrow A$ (note that the converse, $A \Rightarrow \neg\neg A$, is provable intuitionistically). The fact that such a principle is not provable in* **LJ** *may be shown directly but we will obtain it as a consequence of it being equivalent to the excluded middle.*

Indeed, recalling that $\neg\neg(\neg A \vee A)$ is provable in **LJ** *(see above), we have*

$$\dfrac{\begin{array}{c}\vdots\\ \vdash \neg\neg(\neg A \vee A)\end{array} \qquad \dfrac{}{\neg A \vee A \vdash \neg A \vee A}}{\neg\neg(\neg A \vee A) \Rightarrow (\neg A \vee A) \vdash \neg A \vee A}$$

so the provability of $\vdash \neg\neg C \Rightarrow C$ for all C implies the excluded middle (let $C = \neg A \vee A$ above and apply a cut *rule). Conversely, the implication $(\neg A \vee A) \Rightarrow (\neg\neg A \Rightarrow A)$ is provable in* **LJ***:*

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg A \vdash \neg A}\,\mathsf{id} \quad \dfrac{}{\bot \vdash}\,\bot\vdash}{\neg A, \neg\neg A \vdash}\,\Rightarrow\vdash}{\neg A, \neg\neg A \vdash A}\,\mathsf{w}\vdash \qquad \dfrac{\dfrac{}{A \vdash A}\,\mathsf{id}}{A, \neg\neg A \vdash A}\,\vdash\mathsf{w}}{\neg A \vee A, \neg\neg A \vdash A}\,\vee\vdash}{\neg A \vee A \vdash \neg\neg A \Rightarrow A}\,\vdash\Rightarrow$$

Another form of proof by contradiction which is classically valid but intuitionistically invalid is expressed by the principle $(\neg A \Rightarrow A) \Rightarrow A$ (A is

"so true" that even its own negation implies it). We know (see above) that $(\neg(\neg P \vee P) \Rightarrow (\neg P \vee P)$*, so if such a principle were generally provable in* **LJ***, we would obtain the provability of excluded middle, contradicting Example 7.3.*

Example 7.5 (De Morgan's laws) *The classical implication*

$$\neg(\neg A \wedge \neg B) \Rightarrow A \vee B$$

is not provable in intuitionistic logic. Indeed, by the contextual disjunction property of Theorem 7.4.1 we would have either $\neg(\neg A \wedge \neg B) \vdash A$ *or* $\neg(\neg A \wedge \neg B) \vdash B$ *provable in* **LJ***. Suppose without loss of generality that the former is provable. Since A may be supposed to be a propositional constant, the proof must end with*

$$\dfrac{\dfrac{\dfrac{\neg(\neg A \wedge \neg B), A \vdash \bot}{\neg(\neg A \wedge \neg B) \vdash \neg A} \quad \neg(\neg A \wedge \neg B) \vdash \neg B}{\neg(\neg A \wedge \neg B) \vdash \neg A \wedge \neg B} \quad \dfrac{}{\bot \vdash A}}{\neg(\neg A \wedge \neg B) \vdash A}$$

where we used the reversibility of the $(\vdash \wedge)$ *and* $(\vdash \Rightarrow)$ *rules. Now, by cutting* $\neg(\neg A \wedge \neg B) \vdash A$ *with* $\neg(\neg A \wedge \neg B), A \vdash \bot$ *and applying a contraction, we obtain a proof of* $\neg(\neg A \wedge \neg B) \vdash \bot$*, which implies that* $\neg(\neg A \wedge \neg B) \vdash$ *is provable. A cut-free proof of this latter sequent must end with*

$$\dfrac{\vdash \neg A \wedge \neg B \quad \dfrac{}{\bot \vdash}}{\neg(\neg A \wedge \neg B) \vdash}$$

(in fact, we could apply a contraction first and obtain $\neg(\neg A \wedge \neg B) \vdash \neg A \wedge \neg B$ *as top sequent, but we already visited this sequent during our proof search, so it is pointless to reconsider it). From the above, we infer (by reversibility of the* $(\vdash \wedge)$ *rule) the provability of* $\vdash \neg A$*, which we know from Example 7.3 to be impossible.*

Example 7.6 (all binary connectives are primitive) *We saw that De Morgan's laws fail in intuitionistic logic. This prevents us from defining conjunction in terms of disjunction (or vice versa). The classical definition of implication* $A \Rightarrow B := \neg A \vee B$ *fails too. In fact, by the disjunction property in context (Theorem 7.4.1), the provability of* $A \Rightarrow B \vdash \neg A \vee B$ *would imply the provability of either* $A \Rightarrow B \vdash \neg A$ *or* $A \Rightarrow B \vdash B$*. A proof of the first sequent must end with*

$$\dfrac{\dfrac{\dfrac{}{A \vdash A} \quad B \vdash \bot}{A \Rightarrow B, A \vdash \bot}}{A \Rightarrow B \vdash \neg A}$$

and we know that $B \vdash \bot$*, which is equivalent to* $B \vdash$*, is not provable in general (Lemma 7.6). A proof of the second must end with*

$$\dfrac{\vdash A \quad \dfrac{}{B \vdash B}}{A \Rightarrow B \vdash B}$$

with $\vdash A$ *obviously unprovable in general.*

Therefore, in intuitionistic logic, the binary connectives are not *interdefinable, i.e., they must all be taken as primitive.*

Example 7.7 (prenex forms) *The classical implication*

$$((\forall x.A) \Rightarrow B) \Rightarrow (\exists x.A \Rightarrow B)$$

(with $x \notin \mathrm{fv}(B)$) is not provable in $\mathbf{LJ}^1$. Indeed, the contextual existence property of Theorem 7.4.3 would give us the provability in $\mathbf{LJ}^1$ of

$$(\forall x.A) \Rightarrow B \vdash A[t/x] \Rightarrow B$$

for some term t, which is equivalent to the provability of

$$(\forall x.A) \Rightarrow B, A[t/x] \vdash B.$$

Since A and B may be assumed atomic, the only possible ending of a (cut-free) proof of such a sequent is

$$\dfrac{\dfrac{A[t/x] \vdash A}{A[t/x] \vdash \forall x.A} \quad \dfrac{}{B \vdash B}}{(\forall x.A) \Rightarrow B, A[t/x] \vdash B}$$

To conclude, we must have $t = x$, which is impossible because in that case the rule ($\vdash \forall$) could not be applied (x would be free in the context).

Example 7.8 (drinker's formula) *The drinker's formula*

$$\exists x.(D(x) \Rightarrow \forall y.D(y))$$

is not provable in $\mathbf{LJ}^1$. Indeed, the existence property would give us the provability of $\vdash D(t) \Rightarrow \forall y.D(y)$ for some term t, which Lemma 7.1 tells us is unprovable (even in $\mathbf{LK}^1$!).

Let us now reverse the intuition given by these examples and show that, actually, classical logic may be *translated* in intuitionistic logic.

Definition 7.4 (Gödel-Gentzen's double-negation translation) *Let us define the function $(\cdot)^{\mathrm{g}}$ by induction on formulas:*

$$\begin{aligned}
P(t_1, \ldots, t_n)^{\mathrm{g}} &:= \neg\neg P(t_1, \ldots, t_n) \\
\bot^{\mathrm{g}} &:= \bot \\
\top^{\mathrm{g}} &:= \top \\
(A \Rightarrow B)^{\mathrm{g}} &:= A^{\mathrm{g}} \Rightarrow B^{\mathrm{g}} \\
(A \wedge B)^{\mathrm{g}} &:= A^{\mathrm{g}} \wedge B^{\mathrm{g}} \\
(A \vee B)^{\mathrm{g}} &:= \neg\neg(A^{\mathrm{g}} \vee B^{\mathrm{g}}) \\
(\forall x.A)^{\mathrm{g}} &:= \forall x.A^{\mathrm{g}} \\
(\exists x.A)^{\mathrm{g}} &:= \neg\neg\exists x.A^{\mathrm{g}}
\end{aligned}$$

Lemma 7.7 *Let A, B be formulas.*

1. *$\neg\neg\neg A \vdash_{\mathbf{LJ}} \neg A$ and, therefore, $\neg\neg\neg\neg A \vdash_{\mathbf{LJ}} \neg\neg A$;*
2. *$\vdash_{\mathbf{LJ}} \neg\neg\bot \Leftrightarrow \bot$ and $\vdash_{\mathbf{LJ}} \neg\neg\top \Leftrightarrow \top$;*

3. $\neg\neg(A \Rightarrow B) \vdash_{\mathbf{LJ}} A \Rightarrow \neg\neg B$;

4. $\neg\neg(A \wedge B) \vdash_{\mathbf{LJ}} \neg\neg A$ *and* $\neg\neg(A \wedge B) \vdash_{\mathbf{LJ}} \neg\neg B$;

PROOF. Left as exercise to the reader. □

Lemma 7.8 *For every formula A, $\neg\neg A^{\mathrm{g}} \vdash_{\mathbf{LJ}} A^{\mathrm{g}}$.*

PROOF. By induction on A, using Lemma 7.7. The atomic case is point 1; the cases $\bot$ and $\top$ use point 2; the implication case uses point 3; the conjunction case uses point 4; the disjunction and existential quantification cases use again point 2; finally, the induction hypothesis suffices for the universal quantification case. □

Note that the double negation introduced in front of existential quantifiers is fundamental to obtain Lemma 7.8: if we had defined $(\exists x.A)^{\mathrm{g}} = \exists x.A^{\mathrm{g}}$, we would be stuck, because $\neg\neg\exists x.A$ does not intuitionistically imply $\exists x.\neg\neg A$.

Lemma 7.9 *Let A be a formula. Then, for every term t, $A[t/x]^{\mathrm{g}} = A^{\mathrm{g}}[t/x]$.*

PROOF. A straightforward induction on A. □

In what follows, we use the notation $\Gamma \vdash_{\mathbf{LK}} \Delta$ to mean that the sequent $\Gamma \vdash \Delta$ is provable in (two-sided) $\mathbf{LK}^1$. Similarly, $\Gamma \vdash_{\mathbf{LJ}} A$ will mean that the sequent $\Gamma \vdash A$ is provable in $\mathbf{LJ}^1$. Therefore, $\vdash_{\mathbf{LK}} A$ means "A is classically provable", whereas $\vdash_{\mathbf{LJ}} A$ means "A is intuitionistically provable".

Theorem 7.10 $\Gamma \vdash_{\mathbf{LK}} \Delta$ *implies* $\Gamma^{\mathrm{g}}, \neg\Delta^{\mathrm{g}} \vdash_{\mathbf{LJ}} \bot$.

PROOF. By induction on the last rule of the proof of $\Gamma \vdash \Delta$ in (two-sided) **LK**, using Lemma 7.8. The cases of the rules $(\forall\vdash)$ and $(\vdash\exists)$ also use Lemma 7.9. □

Corollary 7.11 (Gödel, Gentzen) *There exists a transformation of formulas $(\cdot)^{\mathrm{g}}$ such that, for every formula A:*

1. $\vdash_{\mathbf{LK}} A^{\mathrm{g}} \Leftrightarrow A$;

2. $\vdash_{\mathbf{LK}} A$ *iff* $\vdash_{\mathbf{LJ}} A^{\mathrm{g}}$.

PROOF. The first point is obvious from the definition of $(\cdot)^{\mathrm{g}}$. For the second point, $\vdash_{\mathbf{LJ}} A^{\mathrm{g}}$ trivially implies $\vdash_{\mathbf{LK}} A^{\mathrm{g}}$, which implies $\vdash_{\mathbf{LK}} A$ (by point 1); for the converse, $\vdash_{\mathbf{LK}} A$ implies, via Theorem 7.10, that the sequent $\neg A^{\mathrm{g}} \vdash \bot$ is provable in **LJ**, from which we immediately obtain $\vdash \neg\neg A^{\mathrm{g}}$, from which $\vdash_{\mathbf{LJ}} A^{\mathrm{g}}$ follows by Lemma 7.8. □

Corollary 7.12 (Gödel, Gentzen) *Classical logic and intuitionistic logic are equiconsistent.*

PROOF. Immediate from the above. □

Remark 7.2 *As anticipated above, the results of Corollary 7.11 tell us that it is not quite fair to think of intuitionistic logic simply as a restriction of classical logic. It is more correct to think of it as a* refinement*: intuitionistic logic is sensitive to distinctions which classical logic ignores (typically, the difference between existence and impossibility of non-existence) but is not really less powerful because, for every classically provable formula, intuitionistic logic proves a formula which is classically equivalent to it.*

7.3 Minimal logic

It is possible to restrict even further the shape of sequents, obtaining what is known as *minimal logic*:

Definition 7.5 (minimal sequent) *A sequent is* minimal *if it is of the shape* $\Gamma \vdash A$*, i.e., if the right side contains* exactly *one formula. The rules of sequent calculus for minimal logic, denoted by* **LM***, are those of* **LJ** *(Fig. 7.1), minus the rule* $(\bot \vdash)$ *(the only one which may introduce non-minimal sequents) and the rule* $(\vdash \mathsf{w})$ *(whose premise is not minimal). The other rules are left untouched, with the proviso that* Σ *is never empty.*

Of course, cut-elimination holds for **LM** too. The cut-elimination rules are exactly those of **LJ**, with the difference that Σ is always equal to a formula.

The most important consequence of this additional restriction is the elimination of all structural rules to the right of sequents: if intuitionistic logic forbids contraction, minimal logic also forbids weakening.

The other difference with respect to intuitionistic logic is that, in minimal logic, $\bot$ has no special rule associated to it, so it behaves just like a propositional constant P. In particular, minimal logic refuses the principle known as *ex falso quodlibet*, represented by the formula $\bot \Rightarrow A$, which is unprovable in **LM**. Indeed, by reversibility of $(\vdash\Rightarrow)$, if $\vdash \bot \Rightarrow A$ were provable, then so would be $\bot \vdash A$. If we take A to be atomic, we see that no non-cut rule may have that sequent as consequence.

The interest of minimal logic will become clear when we will introduce the Curry-Howard correspondence (Chapter 9). For the moment, let us observe that the double-negative translation, Theorem 7.10 and its corollaries actually apply with **LM** instead of **LJ**. Indeed, the reader is invited to check that Lemmas 7.7 and 7.8 hold using minimal (rather than just intuitionistic) sequents. So minimal logic is merely a minor variant of intuitionistic logic and is essentially equivalent to it.

Chapter 8

Natural Deduction

Sequent calculus is centered upon the fundamental symmetry between premises and conclusions of a proof. This is reflected in the fact that each connective has two rules: a right rule describing how new conclusions may be drawn from existing ones, and a left rule describing how the existing premises may be extended to stronger premises.

Everyday mathematical practice is centered much more on the former activity than on the latter and that is why the left rules of sequent calculus may look awkward at first sight. For instance, the rule

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$$

looks perfectly natural: if Γ proves A and Γ proves B, then Γ proves $A \wedge B$. The corresponding left rules are a bit more contrived. For instance, the rule

$$\frac{\Gamma, A \vdash C}{\Gamma, A \wedge B \vdash C}$$

says that if A (and Γ) is enough to prove C, then surely $A \wedge B$ is enough too. This is straightforward, of course, but less natural. Indeed, the intuitive flow of reasoning goes from premises to conclusions (*i.e.*, left to right), whereas left logical rules force us to go backwards. In other words, intuitive proof construction works precisely along an orientation (*i.e.*, a disruption) of the symmetry of sequent calculus, resulting in half of its rules looking unnatural.

It would perhaps be more intuitive to consider the following rule:

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}$$

which states that if I know $A \wedge B$ (under the hypotheses Γ), then I know in particular A. This respects the natural flow of reasoning, because it goes from premises ($A \wedge B$) to conclusions (A).

The above rule reveals to us a different symmetry than premise/conclusion. Indeed, we see that, if the rule ($\vdash \wedge$) tells us how to build, *i.e.*, to introduce a conjunction, the second rule we suggested above tells us how to *use* a conjunction, *i.e.*, how to eliminate it. It turns out that the introduction/elimination symmetry may be taken as the hinge of another proof system, called *natural*

Minimal natural deduction:

$$\frac{}{\Gamma, A \vdash A}\ \mathsf{id}$$

$$\frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash B}{\Gamma \vdash B}\ \Rightarrow\!\mathrm{E} \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow\!\mathrm{I}$$

(no elimination rule for $\top$) $\qquad \dfrac{}{\Gamma \vdash \top}\ \top\mathrm{I}$

$$\frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_i}\ \wedge\mathrm{E}_i,\ i\in\{1,2\} \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \wedge\mathrm{I}$$

$$\frac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C}\ \vee\mathrm{E} \qquad \frac{\Gamma \vdash A_i}{\Gamma \vdash A_1 \vee A_2}\ \vee\mathrm{I}_i\ i\in\{1,2\}$$

$$\frac{\Gamma \vdash \forall x.A}{\Gamma \vdash A[t/x]}\ \forall\mathrm{E} \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x.A}\ \forall\mathrm{I}\ x\notin\mathrm{fv}(\Gamma)$$

$$\frac{\Gamma \vdash \exists x.A \quad \Gamma, A \vdash C}{\Gamma \vdash C}\ \exists\mathrm{E}\ x\notin\mathrm{fv}(\Gamma,C) \qquad \frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x.A}\ \exists\mathrm{I}$$

Additional rule for intuitionistic natural deduction:

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\ \bot\mathrm{E} \qquad \text{(no introduction rule for } \bot\text{)}$$

Additional rule for classical natural deduction:

$$\frac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash A}\ \mathrm{RAA}$$

Figure 8.1: Natural deduction: minimal (**NM**), intuitionistic (**NJ**) and classical (**NK**).

deduction, also introduced by Gentzen. The name "natural" comes, of course, from the above discussion: compared to the premise/conclusion symmetry, the introduction/elimination symmetry adheres more naturally to the intuitive flow of reasoning.

8.1 Sequent presentation

Natural deduction may be presented as a proof system manipulating minimal sequents, *i.e.*, sequents of the form $\Gamma \vdash A$ with exactly one formula to the right, regardless of the logical system. The different logical systems (minimal, intuitionistic or classical logic) are captured by suitably enabling (or disabling) certain inference rules.

The rules defining natural deduction are given in Fig. 8.1. The rules for

minimal natural deduction, denoted by **NM**, are in common to all natural deduction systems. Intuitionistic natural deduction, denoted by **NJ**, is obtained by adding the $\bot$E rule to **NM**. Classical natural deduction, denoted by **NK**, is obtained by adding the RAA rule (*reductio ad absurdum*) to **NJ**.

As anticipated above, each connective induces a pair of elimination and introduction rules. The introduction rules are identical to the right rules of **LM**. On the contrary, the elimination rules are quite different from the corresponding left rules of sequent calculus, although we will see that the two are strongly related. Indeed, an elimination of natural deduction corresponds to a combination of left rule, cut and (perhaps) structural rules in sequent calculus.

Note the absence of cut and structural rules in natural deduction. The absence of cut is in fact only apparent: as we will see in the next section, cuts correspond to certain patterns (an introduction immediately followed by an elimination) which may be eliminated through a process similar to cut-elimination. On the other hand, the structural rules are, so to speak, embedded in the remaining rules: weakening is embedded in the id rule and contraction is embedded in every binary (or ternary) rule. This is just a convenient way of presenting the system: it would be awkward to consider rules which operate on the left of $\vdash$ when every logical rule acts on the right. In fact, the absence of structural rules is not an essential ingredient of natural deduction, nor is it exclusive to it: it is possible to present sequent calculus too in a such a structural-free manner, we saw a hint of this in Example 2.2. At any rate, cut and structural rules are admissible in natural deduction:

Lemma 8.1 (Cut and structural rules) *1. If π and σ are proofs of $\Gamma \vdash A$ and $\Delta, A \vdash B$, respectively, then there is a proof of $\Gamma, \Delta \vdash B$ using exactly the logical rules of π and σ;*

2. if π is a proof of $\Gamma \vdash C$, then $\Gamma, A \vdash C$ is provable using the same logical rules as π;

3. if π is a proof of $\Gamma, A, A \vdash C$, then $\Gamma, A \vdash C$ is provable using the same logical rules as π.

PROOF. All points are proved by straightforward inductions. □

For **NK**, there is an alternative rule to RAA, which asserts directly the provability of $\neg\neg A \Rightarrow A$:

$$\frac{\Gamma \vdash \neg\neg A}{\Gamma \vdash A}\ \text{K}$$

The rules K and RAA are immediately seen to be inter-derivable:

$$\dfrac{\dfrac{\Gamma \vdash \neg\neg A}{\Gamma, \neg A \vdash \neg\neg A}\ \text{(admissible)} \qquad \dfrac{}{\Gamma, \neg A \vdash \neg A}\ \mathsf{id}}{\dfrac{\Gamma \vdash \bot}{\Gamma \vdash A}\ \text{RAA}}\ \Rightarrow\text{E}$$

which shows that K is derivable from RAA (recall that $\neg C = C \Rightarrow \bot$, and that structural rules are admissible), and

$$\dfrac{\dfrac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash \neg\neg A}\ \Rightarrow\text{I}}{\Gamma \vdash A}\ \text{K}$$

which shows that RAA is derivable from K.

8.2 Natural deduction and sequent calculus

In this section, we will show that provability in natural deduction exactly matches provability in sequent calculus. To make proofs shorter, we will only consider the fragment of logic restricted to the connectives $\Rightarrow$, $\vee$ and the constant $\bot$. The reader is invited to check that all results are valid in the presence of conjunction and quantifiers (and $\top$).

Proposition 8.2 *If $\Gamma \vdash A$ is provable in* **NM** *(resp.* **NJ***,* **NK***) then $\Gamma \vdash A$ is provable in* **LM** *(resp.* **LJ***,* **LK***).*

Proof. By induction on the last rule of the proof of $\Gamma \vdash A$ in natural deduction. The assumption rule and the introduction rules are immediate. For what concerns the elimination rules, we have (implicitly using the induction hypothesis)

$$\dfrac{\dfrac{\vdots\,1}{\Gamma \vdash B \Rightarrow C} \quad \dfrac{\vdots\,2}{\Gamma \vdash B}}{\Gamma \vdash C} \qquad \rightsquigarrow \qquad \dfrac{\dfrac{\vdots\,1}{\Gamma \vdash B \Rightarrow C} \quad \dfrac{\dfrac{\vdots\,2}{\Gamma \vdash B} \quad \dfrac{}{C \vdash C}}{\Gamma, B \Rightarrow C \vdash C}}{\dfrac{\Gamma, \Gamma \vdash C}{\Gamma \vdash C}}$$

and

$$\dfrac{\dfrac{\vdots\,1}{\Gamma \vdash B \vee C} \quad \dfrac{\vdots\,2}{\Gamma, B \vdash A} \quad \dfrac{\vdots\,3}{\Gamma, C \vdash A}}{\Gamma \vdash A} \qquad \rightsquigarrow \qquad \dfrac{\dfrac{\vdots\,1}{\Gamma \vdash B \vee C} \quad \dfrac{\dfrac{\vdots\,2}{\Gamma, B \vdash A} \quad \dfrac{\vdots\,3}{\Gamma, C \vdash A}}{\Gamma, B \vee C \vdash A}}{\dfrac{\Gamma, \Gamma \vdash A}{\Gamma \vdash A}}$$

In both cases, an elimination rule is translated into a left rule (the additive version in the case of disjunction), followed by a cut rule, followed by structural rules (*i.e.*, contractions) if necessary.

In the case of **NJ**, we also need to treat the elimination of absurdity. The pattern is the same (left rule, cut rule, structural rules, a weakening in this case):

$$\dfrac{\dfrac{\vdots\,1}{\Gamma \vdash \bot}}{\Gamma \vdash A} \qquad \rightsquigarrow \qquad \dfrac{\dfrac{\dfrac{\vdots\,1}{\Gamma \vdash \bot} \quad \dfrac{}{\bot \vdash}}{\Gamma \vdash}}{\Gamma \vdash A}$$

For **NK**, translating double negation elimination is immediate:

$$\dfrac{\dfrac{\vdots\,1}{\Gamma \vdash \neg\neg A}}{\Gamma \vdash A} \qquad \rightsquigarrow \qquad \dfrac{\dfrac{\vdots\,1}{\Gamma \vdash \neg\neg A} \quad \dfrac{\dfrac{}{A \vdash A}}{\vdash \neg A, A}}{\neg\neg A \vdash A}}{\Gamma \vdash A}$$

□

For the converse, we need an auxiliary lemma to deal with structural rules. In the following, by "proof" we mean "natural deduction proof".

Let Δ be a sequence of formulas $D_1, \ldots, D_n$. In the following, we set $\bigvee \Delta = D_1 \vee \ldots \vee D_n$ (we fix one associative pattern, they are all provably equivalent anyway), and $\bigvee \Delta = \bot$ in case $n = 0$.

Proposition 8.3 *If $\Gamma \vdash \Delta$ is provable in* **LM** *(resp.* **LJ**, **LK***), then $\Gamma \vdash \bigvee \Delta$ is provable in* **NM** *(resp.* **NJ**, **NK***).*

Proof. The proof is once more by induction on the last rule proving the sequent $\Gamma \vdash \Delta$. The case of **LM** is simpler because the sequent is actually of the form $\Gamma \vdash D$, with D a minimal formula. As usual, the axiom rule and the right rules are immediate. The cut rule and the structural rules on the left are given by Lemma 8.1. For what concerns the left rules, we have, using the induction hypothesis and point 1 of Lemma 8.1,

$$\dfrac{\begin{array}{cc}\vdots\,1 & \vdots\,2\\ \Gamma \vdash A & \Sigma, B \vdash D\end{array}}{\Gamma, \Sigma, A \Rightarrow B \vdash D} \quad \rightsquigarrow \quad \dfrac{\dfrac{}{\Gamma, A \Rightarrow B \vdash A \Rightarrow B} \quad \vdots\,1\ \Gamma \vdash A}{\Gamma, A \Rightarrow B \vdash B}}{\vdots\,2}{\Gamma, \Sigma, A \Rightarrow B \vdash D}$$

and, using the induction hypothesis and point 2 of Lemma 8.1,

$$\dfrac{\begin{array}{cc}\vdots\,1 & \vdots\,2\\ \Gamma, A \vdash D & \Gamma, B \vdash D\end{array}}{\Gamma, A \vee B \vdash D} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{}{\Gamma, A \vee B \vdash A \vee B} \quad \vdots\,1\ \Gamma, A \vee B, A \vdash D \quad \vdots\,2\ \Gamma, A \vee B, B \vdash D}{\Gamma, A \vee B \vdash D}$$

So left rules are translated, as expected, as elimination rules (plus some structural manipulations).

The translation of **LJ** is immediate: the left $\bot$ rule is just an assumption rule in **NJ**, and the right rule may only be applied when Δ is empty, so it translates to nothing in **NJ**, because in that case we already have $\bigvee \Delta = \bot$.

The case of **LK** requires some additional work, because right rules may result in sequents with more than one formula to the right, *i.e.*, in Δ, and this has non-trivial consequences with our definition of $\bigvee \Delta$. First we use the fact that, in a proof of $\Gamma \vdash \bigvee \Delta$, we may always "single out" an element of the disjunction:

Lemma 8.4 *In* **NJ**, *$\Gamma \vdash D \vee A$ provable implies $\Gamma, \neg D \vdash A$ provable.*

Proof. By point 2 of Lemma 8.1, if we have $\Gamma \vdash D \vee A$ we also have $\Gamma, \neg D \vdash D \vee A$. Then, using a disjunction elimination rule, we have

$$\dfrac{\Gamma, \neg D \vdash D \vee A \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, \neg D, D \vdash \neg D} \quad \dfrac{}{\Gamma, \neg D, D \vdash D}}{\Gamma, \neg D, D \vdash \bot}}{\Gamma, \neg D, D \vdash A} \qquad \dfrac{}{\Gamma, \neg D, A \vdash A}}{\Gamma, \neg D \vdash A}$$

□

Then, we use the fact that, in classical logic, $D \Rightarrow A$ is equivalent to $\neg D \vee A$, which is subsumed by the following:

Lemma 8.5 *In* **NK**, *we have* $\neg D \Rightarrow A \vdash D \vee A$.

Proof. We give the derivation in tree form, which is much more compact:

$$\dfrac{\dfrac{\dfrac{[\neg(D \vee A)]^{\ddagger} \quad \dfrac{[A]^{\dagger}}{D \vee A}}{\dfrac{\bot}{\neg A}\,{\scriptstyle\dagger}} \qquad \neg D \Rightarrow A}{A} \qquad \dfrac{\dfrac{[\neg(D \vee A)]^{\ddagger} \quad \dfrac{[D]^{*}}{D \vee A}}{\dfrac{\bot}{\neg D}\,{\scriptstyle *}}}{\dfrac{\bot}{\neg\neg(D \vee A)}\,{\scriptstyle\ddagger}}}{D \vee A}$$

Note the (fundamental!) use of double negation elimination. □

The proof now is easy: right weakening, disjunction and $\bot$ rules are immediate (modulo associativity of the disjunction $\bigvee \Delta$). For what concerns right contraction and right implication, the strategy is the same: we single out the active occurrence(s) through Lemma 8.4; then, we apply the appropriate rule and we use Lemma 8.5 to get the desired conclusion. We give the detail of the right implication rule, leaving contraction to the reader. We want to simulate the rule

$$\frac{\Gamma, A \vdash \Delta, B}{\Gamma \vdash \Delta, A \Rightarrow B.}$$

By the induction hypothesis, in **NK** we have $\Gamma, A \vdash \bigvee \Delta \vee B$ (perhaps modulo some re-association of the disjunctions). Using Lemma 8.4, we obtain a proof of $\Gamma, \neg \bigvee \Delta, A \vdash B$, from which we build

$$\dfrac{\dfrac{\Gamma, \neg \bigvee \Delta, A \vdash B}{\Gamma, \neg \bigvee \Delta \vdash A \Rightarrow B}}{\Gamma \vdash \neg\Delta \Rightarrow (A \Rightarrow B).}$$

From that, Lemma 8.5 gives us the desired proof of $\Gamma \vdash \bigvee \Delta \vee (A \Rightarrow B)$ (via point 1 of Lemma 8.1). □

8.3 Proof tree presentation

Natural deduction may be equivalently presented in terms of proof trees. A proof tree consists of a tree of the form

$$\begin{array}{c} C_1 \quad \ldots \quad C_n \\ \vdots \\ A \end{array}$$

in which the root A is the *conclusion*, *i.e.*, the formula which is proved, and the leaves $C_1, \ldots, C_n$ are the *hypotheses* needed to prove A (they are *occurrences*

of formulas, *i.e.*, we may have $C_i = C_j$ for some $i \neq j$). Some leaves of the tree are *discharged* (denoted by $[C_i]$), meaning that they are not to be considered hypotheses. If all leaves are discharged, then the conclusion holds without hypotheses (it is the case, for example, of every propositional tautology).

8.3.1 Minimal natural deduction

The proofs of natural deduction for minimal logic (**NM**) are built inductively as follows:

Assumption: for every formula A,

$$A$$

is a proof of conclusion A and hypothesis A.

Implication introduction: if B is provable from an arbitrary number of hypotheses A, then $A \Rightarrow B$ is provable as follows:

$$\dfrac{\begin{array}{c}[A]^* \\ \vdots \\ B\end{array}}{A \Rightarrow B}\ {}^*$$

where the hypotheses A have been "discharged", *i.e.*, they are no longer hypotheses of the proof (observe that some occurrences of A might still be left as hypotheses, and that the rule might actually discharge no hypothesis at all). The sign "$*$" is used to identify which occurrences of A are discharged by the rule.

Implication elimination: if $A \Rightarrow B$ and A are provable, then B is provable (this is *modus ponens*):

$$\dfrac{A \Rightarrow B \quad A}{B}$$

Conjunction introduction: if A and B are provable, then $A \wedge B$ is provable as follows:

$$\dfrac{A \quad B}{A \wedge B}$$

Conjunction elimination: if $A \wedge B$ is provable, then both A and B are provable, as follows:

$$\dfrac{A \wedge B}{A} \qquad \dfrac{A \wedge B}{B}$$

Disjunction introduction: if any of A or B is provable, then $A \vee B$ is provable, as follows:

$$\dfrac{A}{A \vee B} \qquad \dfrac{B}{A \vee B}$$

Disjunction elimination: if $A \vee B$ is provable, and if C is provable by assuming A and by assuming B, then C is provable:

$$\dfrac{A \vee B \quad \begin{array}{c}[A]^* \\ \vdots \\ C\end{array} \quad \begin{array}{c}[B]^* \\ \vdots \\ C\end{array}}{C}\ {}^*$$

The discharged hypotheses are subjected to the same remarks as in the arrow introduction rule.

Universal quantifier introduction: if A is provable from hypotheses not containing the free variable x, then $\forall x.A$ is provable, under the same hypotheses:

$$\frac{A}{\forall x.A}$$

Universal quantifier elimination: if $\forall x.A$ is provable, then all instances of A are provable:

$$\frac{\forall x.A}{A[t/x]}\ t$$

where t is an arbitrary term (which is specified by the rule, in case x does not actually appear in A).

Existential quantifier introduction: if $A[t/x]$ is provable for some term t, then $\exists x.A$ is provable, as follows:

$$\frac{A[t/x]}{\exists x.A}$$

Existential quantifier elimination: if $\exists x.A$ is provable, if $x \notin \mathrm{fv}(()C)$ and if from A we may deduce C without additional hypotheses on x, then we may prove C:

$$\frac{\exists x.A \qquad \begin{array}{c}[A]^* \\ \vdots \\ C\end{array}}{C}\ *$$

The discharged hypotheses are subjected to the same remarks as in the arrow introduction rule.

Example 8.1 *Here are the proofs of two well known minimal tautologies:*[1]

$$\frac{\dfrac{[A]^*}{B \Rightarrow A}\ *}{A \Rightarrow B \Rightarrow A} \qquad \frac{\dfrac{\dfrac{\dfrac{[A \Rightarrow B \Rightarrow C]^* \quad [A]^\ddagger}{B \Rightarrow C} \quad \dfrac{[A \Rightarrow B]^\dagger \quad [A]^\ddagger}{B}}{C}}{A \Rightarrow C}\ \ddagger}{(A \Rightarrow B) \Rightarrow A \Rightarrow C}\ \dagger}{(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C}\ *$$

8.3.2 Intuitionistic natural deduction

The proofs of natural deduction for intuitionistic logic (**NJ**) are those of **NM** extended as follows:

[1] Every propositional minimal tautology is a consequence of these two formulas via the *modus ponens* rule. Anticipating on Chapter 9, the discovery of what we now call the Curry-Howard correspondence began with Haskell Curry's remark that these two formulas exactly match the types of the two combinators **K** and **S**, which in turn generate every λ-term through application.

Absurdity elimination: if a set of hypotheses led us to a proof of $\bot$, then we may prove anything at all from the same hypotheses (this is *ex falso quodlibet*):

$$\frac{\bot}{A}$$

8.3.3 Classical natural deduction

The proofs of natural deduction for classical logic (**NK**) are those of **NJ** with one additional rule:

Proof by contradiction: if assuming $\neg A$ leads us to a contradiction, then we have a proof of A:

$$\dfrac{\begin{array}{c}[\neg A]^{*}\\ \vdots\\ \bot\end{array}}{A}\ {}^{*}$$

The usual remarks for discharged formulas apply.

Of course, another possible rules for obtaining **NK** from **NJ** is the following:

$$\frac{\neg\neg A}{A}$$

Note the difference between the rule for proof by contradiction RAA and the application of an implication introduction rule, perfectly valid in intuitionistic logic:

$$\dfrac{\begin{array}{c}[\neg A]^{*}\\ \vdots\\ \bot\end{array}}{\neg\neg A}\ {}^{*}$$

Classical logic comes from intuitionistic logic by identifying the impossibility of invalidity with validity, which is what the double negation elimination rule is about.

A further alternative is to use excluded middle (*tertium non datur*) as an axiom, *i.e.*, a rule with no hypothesis:

$$\frac{}{\neg A \vee A}$$

We may easily prove the equivalence of this rule with the other two introduced above. For instance, this is how we can prove the excluded middle by contradiction:

$$\dfrac{\dfrac{\dfrac{[\neg(\neg A \vee A)]^{*}\quad \dfrac{[\neg A]^{\dagger}}{\neg A \vee A}}{\bot}}{\neg\neg A}\ {}^{\dagger} \qquad \dfrac{\dfrac{[\neg(\neg A \vee A)]^{*}\quad \dfrac{[A]^{\ddagger}}{\neg A \vee A}}{\bot}}{\neg A}\ {}^{\ddagger}}{\bot}}{\neg A \vee A}\ {}^{*}$$

where the last rule is a *reductio ad absurdum*. Conversely, the double negation elimination rule may be derived using excluded middle as follows:

$$\dfrac{\dfrac{}{\neg A \vee A} \qquad \dfrac{\dfrac{\neg\neg A \quad [\neg A]^*}{\bot}}{A} \qquad [A]^*}{A}\ {}^*$$

where the last rule is a disjunction elimination, and we used absurdity elimination (*ex falso quodlibet*) in the left branch.

8.4 Normalization (cut-elimination in natural deduction)

Let us look at the translation in **NM** of a cut rule whose both premises are introduced by the rules immediately above (a principal cut). The most important case for our purposes is that in which the cut formula is of the form $A \Rightarrow B$. We furthermore suppose that the occurrence of A used by the right rule is the result of a certain number of structural rules (contractions or weakening):

$$\dfrac{\dfrac{\dfrac{\vdots\,1}{\Gamma, A, \ldots, A \vdash B}}{\Gamma, A \vdash B}}{\Gamma \vdash A \Rightarrow B} \qquad \dfrac{\dfrac{\vdots\,2}{\Delta \vdash A} \quad \dfrac{\vdots\,3}{B, \Sigma \vdash C}}{A \Rightarrow B, \Delta, \Sigma \vdash C}$$
$$\Gamma, \Delta, \Sigma \vdash C$$

The **NM** translation is

$$\dfrac{\Gamma \quad \dfrac{\dfrac{[A]^* \ldots [A]^*}{\vdots\,1}}{B}}{A \Rightarrow B}\ {}^* \qquad \dfrac{\Delta}{\vdots\,2}{A} \qquad \Sigma$$
$$\dfrac{B \qquad \Sigma}{\vdots\,3}{C}$$

We notice that *the introduction of the implication $A \Rightarrow B$ is immediately followed by its elimination*. The reader is invited to check that such an introduction/elimination pattern is present in the **NM** translation of all cases of principal cut in **LM**.

The introduction/elimination pattern looks unnecessary: in the above case, we could eliminate it by considering the proof

$$\dfrac{\Gamma \quad \dfrac{\dfrac{\Delta}{\vdots\,2}{A} \quad \ldots \quad \dfrac{\Delta}{\vdots\,2}{A}}{\vdots\,1}}{B} \qquad \Sigma$$
$$\dfrac{B \qquad \Sigma}{\vdots\,3}{C}$$

Note that the proof marked by (2) may be duplicated several times, or completely discarded, according to the number of occurrences of A discharged by the introduction rule under consideration.

The reader is invited to check that, if one translates in **NM** the reduced version of the principal cut, which is

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots\ 2}{\Delta \vdash A} \quad \dfrac{\vdots\ 1}{\Gamma, A, \ldots, A \vdash B}}{\Gamma, \Delta, \ldots, A \vdash B}}{\Gamma, \Delta, \ldots, \Delta \vdash B}}{\Gamma, \Delta \vdash B} \qquad \dfrac{\vdots\ 3}{B, \Sigma \vdash C}$$
$$\Gamma, \Delta, \Sigma \vdash C$$

(where all rules are either cuts or structural), one obtains exactly the "simplified" natural deduction proof given above. In other words, the obvious way of getting rid of the introduction/elimination pattern precisely matches the standard cut-elimination step.

For this reason, the natural deduction equivalent of a cut-free proof is a proof containing no introduction/elimination pattern as the one above. Such proofs are called *normal*. Accordingly, the cut-elimination process is replaced by a *normalization* process, which aims at eliminating all introduction/elimination patterns by means of the following rewriting rules:

$$\dfrac{\dfrac{[A]^* \ldots [A]^*}{\vdots}}{\dfrac{B}{\dfrac{A \Rightarrow B}{B}\ ^*} \quad \dfrac{\vdots}{A}} \qquad \rightarrow \qquad \dfrac{\vdots \quad \vdots}{A \ldots A}$$
$$\dfrac{\vdots}{B}$$

$$\dfrac{\dfrac{\vdots \quad \vdots}{A \quad B}}{\dfrac{A \wedge B}{A}} \qquad \rightarrow \qquad \dfrac{\vdots}{A} \qquad\qquad \dfrac{\dfrac{\vdots \quad \vdots}{A \quad B}}{\dfrac{A \wedge B}{B}} \qquad \rightarrow \qquad \dfrac{\vdots}{B}$$

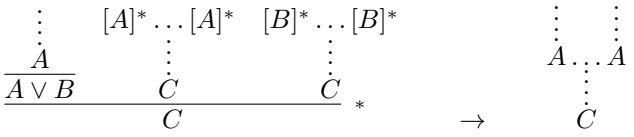
$$\dfrac{\dfrac{\dfrac{\vdots\ \pi}{A}}{\forall x.A}\ ^t}{A[t/x]} \qquad \rightarrow \qquad \dfrac{\vdots\ \pi[t/x]}{A[t/x]}$$

In the last rule, by $\pi[t/x]$ we mean the whole proof π in which every free occurrence of x is replaced by t in every formula occurring in π. This is possible because, thanks to the condition on the introduction rule of universal quantification, x does not appear free in the hypotheses of π.

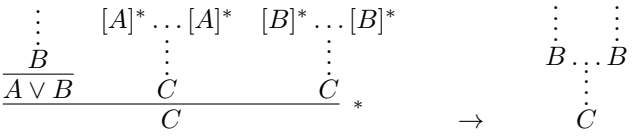
Of course, it is implicitly intended that the above rewriting rules may be applied at any place within a proof, not just at the root.

If we restrict to the fragment of **NM** containing only implication, conjunction and universal quantification (the *negative fragment*), the above rules are enough. This is in contrast with cut-elimination in sequent calculus, which

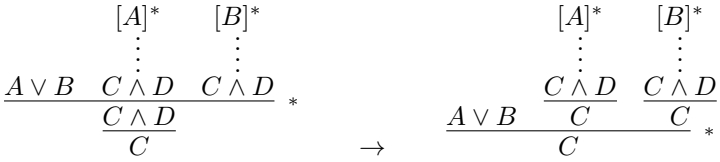
needs to deal with non-principal cuts (those that we called simply and doubly commutative). On the other hand, if we wish to consider also the positive connectives (disjunction and existential quantification), then the reduction steps corresponding to principal cuts must also be accompanied by so-called *commuting conversions*. For instance, in the case of disjunction, we have the reduction



and its symmetric counterpart



but we also have 6 commuting conversions, one for each elimination rule, such as



If we also consider existential quantification, then there are a total of 14 commuting conversions (7 for disjunction, 7 for existential quantification). These are all needed to achieve a normal form because, without them, the introduction/elimination patterns which form "across" positive elimination rules, and which still correspond to cuts in sequent calculus, would be impossible to remove.

Chapter 9

The Curry-Howard Correspondence

9.1 The simply typed λ-calculus

We fix a (countably infinite) set of *ground types* $X, Y, \ldots$ (which can be thought of as including, for instance, the type of Booleans, integers, lists of integers, etc.). *Simple types* are generated by the following grammar:

$$T, U ::= X \mid T \to U,$$

where X ranges over ground types.

We fix a countably infinite set of *variables* $x, y, \ldots$ (in spite of the identical notation, these are not to be confused with the first-order variables used in a first-order language). Additionally, we fix a function $\mathcal{T}$ from variables to types such that, for every type T, there are infinitely many variables x such that $\mathcal{T}(x) = T$.

The *simply-typed terms*, together with their assigned type, are inductively defined as follows. In the definition, we use the notation $M : T$ as an abbreviation for "M is a term of type T" and we also use the notation $\mathrm{fv}(M)$ to denote the set of free variables of a term M, defined at the same time as the terms themselves, in the usual way.

Variable: for every variable x such that $\mathcal{T}(x) = T$, $x^T : T$, and $\mathrm{fv}(x^T) = \{x\}$.

Abstraction: for every variable x such that $\mathcal{T}(x) = T$ and $M : U$, $\lambda x^T.M : T \to U$, and $\mathrm{fv}(\lambda x^T.M) = \mathrm{fv}(M) \setminus \{x\}$.

Application: for all $M : T \to U$ and $N : T$, $MN : U$, and $\mathrm{fv}(MN) = \mathrm{fv}(M) \cup \mathrm{fv}(N)$.

To make notations lighter, we stipulate that application associates to the left, *i.e.*, MNP means $(MN)P$, and we drop the type superscripts for bound variables, since their type is already specified in the abstraction: $\lambda x^T.x$ means $\lambda x^T.x^T$.

Computation in the λ-calculus is expressed by the following rewriting rule, called *β-reduction*, which is easily seen to preserve types:

$$(\lambda x^T.M)N \quad \to \quad M[N/x]$$

Logic		Computer Science
formula	=	type
proof	=	program
cut-elimination	=	execution

Table 9.1: The Curry-Howard correspondence

where $M[N/x]$ denotes, as usual, the capture-free substitution of N to all free occurrences of x appearing in M. Just like in the pure λ-calculus, a term of the form given in the left-hand side of the above rules is called *redex*. Of course, the above rewriting rules are intended to apply to any subterm which is a redex. A term containing no redex is called *normal*.

We may now observe that the simple types are in bijection with the formulas of the implication fragment of minimal logic, with the "dictionary"

- ground types $\longleftrightarrow$ atomic formulas;
- arrow type $\longleftrightarrow$ implication.

Similarly, the simply-typed terms are in bijection with the proofs of the implication fragment of **NM**, with the "dictionary":

- variables $\longleftrightarrow$ assumptions;
- abstraction $\longleftrightarrow$ implication introduction rule;
- application $\longleftrightarrow$ implication elimination rule.

Moreover, and this is the key observation, β-reduction exactly matches the normalization rule of **NM**, *i.e.*, we have

- one β-reduction step $\longleftrightarrow$ one normalization step,

as we invite the reader to check. This is the *Curry-Howard correspondence*, summarized in Table 9.1.

9.2 Product and sum types

Now that we know that each formula is a type and each proof is a program, we may wonder whether the remaining logical constructions (conjunction, disjunction, quantifiers) have any computational meaning. It turns our that the answer is positive, with the exception of first-order quantifiers, which do not give anything interesting (to address first-order logic in a computationally meaningful way, one must look at *dependent types*, which are out of the scope of these notes). Second-order quantifiers, on the contrary, are extremely interesting from a computational point of view, as we shall see later.

A *product type* is of the form $T \times U$, where T and U are types. The corresponding constructions on terms are

Pairing: if $M : T$ and $N : U$, then $\langle M, N\rangle : T \times U$;

Projections: if $M : T \times U$, then $\pi_1 M : T$ and $\pi_2 M : U$.

Pairing and projections interact through the following rewriting rule, which is added to β-reduction:

$$\pi_1\langle M, N\rangle \quad \to \quad M \qquad\qquad \pi_2\langle M, N\rangle \quad \to \quad N$$

A *sum type* is of the form $T+U$, where T and U are types. The corresponding constructions on terms are

Injections: if $M : T$ and $N : U$, then $\iota_1^U M : T + U$ and $\iota_2^T N : T + U$;

Case: if $M : T + U$ and $P, Q : V$, then $\mathsf{case}(x^T, y^U)\, M\, P\, Q : V$, and x (resp. y) becomes bound in P (resp. Q).

In presence of injections and case constructs, β-reduction is augmented with the following rewriting rules:

$$\mathsf{case}(x^T, y^U)\, (\iota_1^U M)\, P\, Q \quad \to \quad P[M/x]$$

$$\mathsf{case}(x^T, y^U)\, (\iota_2^T M)\, P\, Q \quad \to \quad Q[M/y]$$

This time, we also need the following commuting conversions:

$$(\mathsf{case}(x^T, y^U)\, M\, P\, Q)N \quad \to \quad \mathsf{case}(x^T, y^U)\, M\, (PN)\, (QN)$$

$$\pi_i(\mathsf{case}(x^T, y^U)\, M\, P\, Q) \quad \to \quad \mathsf{case}(x^T, y^U)\, M\, (\pi_i P)\, (\pi_i Q)$$

$$\mathsf{case}(x^T, y^U)\, (\mathsf{case}(z^V, w^W)\, M\, R\, S)\, P\, Q \quad \to$$

$$\mathsf{case}(z^V, w^W)\, M\, (\mathsf{case}(x^T, y^U)\, R\, P\, Q)\, (\mathsf{case}(x^T, y^U)\, S\, P\, Q)$$

We invite the reader to check that the Curry-Howard correspondence extends to product and sum types, with the following additions to the "dictionary":

- product type $\longleftrightarrow$ conjunction;
- sum type $\longleftrightarrow$ disjunction;
- pairing $\longleftrightarrow$ conjunction introduction rule;
- projections $\longleftrightarrow$ conjunction elimination rules;
- injections $\longleftrightarrow$ disjunction introduction rules;
- case $\longleftrightarrow$ disjunction elimination rule;

and, of course, the augmented β-reduction still corresponds to normalization.

Chapter 10

System F

10.1 Intuitionistic second-order propositional logic

The formulas of (the implication fragment of) second-order propositional logic are defined by the following grammar:

$$A, B ::= X \mid A \Rightarrow B \mid \forall X.A,$$

where X ranges over a denumerably infinite set of *propositional atoms*.

The set of free propositional atoms of a formula A is defined as usual, and is denoted by $\mathrm{fv}(()A)$. If A, B are formulas, $A[B/X]$ denotes, as usual, the capture-free substitution of B to all free occurrences of X in A.

The proofs of intuitionistic second-order propositional logic are those obtained using the rules of minimal natural deduction, in which second-order quantification is treated as follows:

Universal quantifier introduction: if A is provable from hypotheses not containing the free variable X, then $\forall X.A$ is provable, under the same hypotheses:

$$\frac{A}{\forall X.A}$$

Universal quantifier elimination: if $\forall X.A$ is provable, then all instances of A are provable:

$$\frac{\forall X.A}{A[B/X]}\ B$$

where B is an arbitrary formula (which is specified by the rule, in case X does not actually appear in A).

Normalization for second order quantification is very similar to the first order case:

$$\frac{\dfrac{\begin{array}{c}\vdots\ \pi\\ A\end{array}}{\forall X.A}}{A[B/X]}\ B \qquad \rightsquigarrow \qquad \begin{array}{c}\vdots\ \pi[B/X]\\ A[B/X]\end{array}$$

where $\pi[B/X]$ denotes the proof π in which every occurrence of the atom X has been replaced by the formula B (this is why B must be specified by the rule itself).

Observe that, with second-order quantification, minimal natural deduction is actually as expressive as intuitionistic natural deduction, because absurdity may be defined as $\bot := \forall X.X$. In fact, every other usual logical connective is also definable in terms of implication and second-order universal quantification:

- $A \wedge B := \forall X.(A \Rightarrow B \Rightarrow X) \Rightarrow X$;
- $A \vee B := \forall X.(A \Rightarrow X) \Rightarrow (B \Rightarrow X) \Rightarrow X$;
- $\exists X.A := \forall Y.(\forall X.A \Rightarrow Y) \Rightarrow Y$.

We invite the reader to check that, with the above definitions, the usual introduction and elimination rules for the respective connectives are derivable, and that the derivations satisfy the appropriate normalization rules for eliminating introduction/elimination patterns.

Hence, the system we are dealing with is actually second order propositional **NJ**, which is equivalent, in terms of provability, to second order propositional **LJ**, thanks to Propositions 8.2 and 8.3, which immediately carry over to second order.

10.2 Polymorphic types

The *types* of **F** are the simple types plus the *polymorphic types*:

$$T, U ::= X \mid T \to U \mid \Pi X.T.$$

Free and bound type variables are defined as usual, with ΠX being a binder.

The *terms* of **F**, together with their assigned type, are inductively defined by adding the two following formation rules to those of the simply typed λ-calculus:

Universal abstraction: for every propositional atom X and $M : T$, $\Lambda X.M : \Pi X.T$, provided that, for all $x \in \mathrm{fv}(()M)$, $X \notin \mathrm{fv}(()\mathcal{T}(x))$. In that case, $\mathrm{fv}(()\Lambda X.M) = \mathrm{fv}(()M)$.

Universal application: for all $M : \forall X.T$ and type U, $MU : T[U/X]$, and $\mathrm{fv}(()MU) = \mathrm{fv}(()M)$.

Computation in system **F** is expressed by usual β-reduction plus the following rewriting rule (which is, of course, applicable in any context):

$$(\Lambda X.M)T \quad \to \quad M[T/X],$$

where $M[T/X]$ denotes the capture-free substitution of T to all free occurrences of X appearing in M. This rule allows types to change dynamically, which is the meaning of polymorphism.

Observe that the second reduction rule is isomorphic to the normalization rule for second order quantification in **NJ**. Therefore, system **F** extends the Curry-Howard correspondence, by adding the following entries to our "dictionary":

- universal abstraction $\longleftrightarrow$ universal quantifier introduction rule;
- universal application $\longleftrightarrow$ universal quantifier elimination rule.

One consequence of this is that product and sum types may be defined in system **F**, using the second-order definition of conjunction and disjunction. For instance, for product types, we have

$$T \times U := \Pi X.(T \to U \to X) \to X,$$

and

$$\langle M, N\rangle := \Lambda X.\lambda c^{T\to U\to X}.cMN,$$

$$\pi_1^{U,V} := \lambda p^{T\times U}.pT(\lambda x^T y^U.x), \qquad \pi_2^{U,V} := \lambda p^{T\times U}.pU(\lambda x^T y^U.y).$$

These terms mirror exactly the derivations of the second-order encoding of introduction and elimination rules for conjunction. The fact that the encoding respects the usual normalization rules of natural deduction allows us to conclude immediately that

$$\pi_1^{T,U}\langle M, N\rangle \to^* M, \qquad \pi_2^{T,U}\langle M, N\rangle \to^* N,$$

as expected. Of course, we invite the reader to do the computation in system **F** and check this, also in the case of disjunction/sum types.

10.3 Programming in system F

10.3.1 Free structures

Much more than product and sum types can be encoded in system **F**. In fact, any *free structure* (or *free algebra*, as it is often called in computer science) has a natural representation in system **F**.

The most general way of defining free structures uses category theory and the concept of *initial algebra*. We recall the definition.

Definition 10.1 (Initial algebra) *Let $\mathcal{A}$ be a category, and let $F : \mathcal{A} \to \mathcal{A}$ be an endofunctor. An* F-algebra *is a pair (A, α) where A is an object of $\mathcal{A}$ and $\alpha : FA \to A$. Given two F-algebras $(A, \alpha), (B, \beta)$, an F-morphism between them is a morphism $f : A \to B$ which makes the following diagram commute:*

$$\begin{array}{ccc} FA & \xrightarrow{\alpha} & A \\ {\scriptstyle Ff}\downarrow & & \downarrow{\scriptstyle f} \\ FB & \xrightarrow[\beta]{} & B \end{array}$$

It is easy to see that F-algebras and F-morphisms form a category, denoted by $\mathcal{A}_F$. An F-algebra is initial *if it is an initial object in $\mathcal{A}_F$.*

Of course, initial algebras are unique modulo a unique isomorphism, so we may speak of *the* initial algebra of a functor. The initial algebra of F is the "smallest" fixpoint of F, in the following sense:

Proposition 10.1 *If (Z, ζ) is the initial algebra of $F : \mathcal{A} \to \mathcal{A}$, then $(FZ, F\zeta) \cong (Z, \zeta)$ in $\mathcal{A}_F$.*

PROOF. Note that, for every F-algebra (A, α), $(FA, F\alpha)$ is also an F-algebra, and it is immediate to see that α is an F-morphism from $(FA, F\alpha)$ to (A, α). In the case of (Z, ζ), its initial property gives us a (unique) F-morphism $\iota : (Z, \zeta) \to (FZ, F\zeta)$. Again using the initial property, we have $\zeta \circ \iota = id_Z$. For what concerns the opposite direction, using the fact that ι is an F-morphism and that F is a functor, we have $\iota \circ \zeta = F\zeta \circ F\iota = F(\zeta \circ \iota) = id_{FZ}$. □

In the following, we denote by $A + B$ the disjoint union of two sets A and B, and we denote by $[f, g]$ the copairing of two functions $f : A \to C$, $g : B \to C$, which is the canonical function of type $A + B \to C$ defined "case-wise from f and g. Of course the copairing notation generalizes to an arbitrary number of functions: for any family of sets $(A_i)_{i \in I}$, if $f_i : A_i \to C$, $[f_i]_{i \in I} : \sum_{i \in I} A_i \to C$.

Definition 10.2 (Free structure) *Let $C_1, \ldots, C_n : \mathbf{Set} \to \mathbf{Set}$ be functors. A* free structure *on $C_1, \ldots, C_n$ is a tuple $(\Theta, c_1, \ldots, c_n)$ where Θ is a set and $c_i : C_i\Theta \to \Theta$ for $1 \leq i \leq n$, such that $(\Theta, [c_1, \ldots, c_n])$ is the initial algebra of the functor $C_1 + \cdots + C_n : \mathbf{Set} \to \mathbf{Set}$.*

The functions $c_1, \ldots, c_n$ are to be seen as constructors, whose types are given by the functors $C_1, \ldots, C_n$. The fact that the algebra is initial for $C_1 + \cdots + C_n$ means, by Proposition 10.1, that every element of Θ is of the form $c_i(z)$ with $z \in C_i(\Theta)$ for some $1 \leq i \leq n$ and that i and z are unique.

The key consequence is that we may define functions on Θ by *induction* on the structure of its elements. For instance, if A is a set and we have functions $h_i : C_i(A) \to A$ for $1 \leq i \leq n$, we may define a function $f : \Theta \to A$ by

$$f(\theta) = h_i(C_i(f)(z)) \quad \text{whenever } \theta = c_i(z).$$

Some examples of free structures:

1. in the degenerate case $n = 0$, the sum of zero functors is the constant functor 0 yielding the empty set. Since $\emptyset$ is the initial object of **Set**, $\mathbf{Set}_0 \cong \mathbf{Set}$, so $(\emptyset, id_\emptyset)$ is the initial object of $\mathbf{Set}_0$ and the empty set is therefore the free structure on no constructors.

2. Let $n = 2$ and let C_1, C_2 be constant functors yielding two sets T, U, respectively. It is immediate that $(T + U, \iota_1, \iota_2)$ is the free structure on C_1, C_2, with ι_1, ι_2 the canonical injections. In the special case in which $T = U = 1 = \{*\}$, we get the set of Boolean values, and ι_1, ι_2 may be referred to as "true" and "false".

3. Let $n = 1$ and C_1 be the constant functor yielding a set K. In that case, it is easy to see that the free structure is K itself, with the constructor being the identity id_K. So every set is the free structure on... itself.

4. Let $n = 2$, suppose that C_1 is the constant functor yielding 1 and that C_2 is the identity functor. Then, if Θ is the underlying set of the free structure on C_1, C_2, it must come with two constructors

$$\begin{aligned} c_1 &: 1 \to \Theta, \\ c_2 &: \Theta \to \Theta, \end{aligned}$$

such that $(\Theta, [c_1, c_2])$ is initial for the functor sending a set X to $1 + X$ (*i.e.*, which adds an element $* \notin X$ to X). We invite the reader to check that $(\mathbb{N}, 0, \mathrm{succ})$ is the free structure in this case. The function $[0, \mathrm{succ}] : 1 + \mathbb{N} \to \mathbb{N}$ sends $*$ to 0 and a natural number n to $n + 1$.

5. Let $n = 2$, $C_1 = 1$ and $C_2(X) = T \times X$ for a fixed set T. The free structure is $(T^*, \varepsilon, \mathrm{cons})$, where T^* is the set of finite lists (words) of elements of T, ε is the empty list and $\mathrm{cons} : T \times T^* \to T^*$ is the function adding an element to the head of a list. Of course, by letting $T = 1$ we obtain again the integers.

6. Some free structures may be specified in more than one way. For example, setting $T = 2$ (a set with two elements) in the above example, or choosing $n = 3$, $C_1 = 1$ and both C_2, C_3 equal to the identity functor are two different ways of defining binary words.

7. Let $n = 2$, $C_1 = 1$ and $C_2(X) = X \times X$. In this case, we get binary trees, with the one-root-one-leaf tree as the base case and the function building a tree from two trees as the other constructor.

8. Let $n = 2$, $C_1 = 1$ and $C_2(X) = T \to X$, *i.e.*, the set of functions from a fixed set T to X. In this case, we get the set of trees with branches indexed by T; the constructors are the one-root-one-leaf tree and the map building a tree from a T-indexed family of trees. Of course, if we let $T = 2$ we retrieve the above example.

The reader will have understood by now that any inductive date type, *i.e.*, a set of objects whose elements are defined by induction using a finite number of constructors, may be defined as a free structure.

10.3.2 Representing free structures in system F

In system **F**, we may imagine that types represent sets: X is a "generic" set, $T \to U$ is the function space, and we have seen that we may define the Cartesian product $T \times U$ and the disjoint sum $T + U$. Then, we may suppose that, in certain cases, the functors $C_1, \ldots, C_n$ may be represented by types of **F** containing a fixed free atom X representing the argument of the functor. The fact that the C_i are covariant will be reflected by the fact X may only occur positively.[1] For example:

- if C_1 and C_2 are as in example 4, and if we admit to have a type 1 representing the singleton set (which we actually do, as we shall see), we have $C_1 = 1$ and $C_2 = X$;
- in example 5, we have $C_1 = 1$ and $C_2 = T \times X$;
- in example 7, we have $C_1 = 1$ and $C_2 = X \times X$;
- in example 8, we have $C_1 = 1$ and $C_2 = T \to X$.

[1] An occurrence of X is *positive* (resp. *negative*) in C_i if it appears to the left of an even (resp. odd) number of arrows.

In such cases, the free structure on $C_1, \ldots, C_n$ may be represented in system **F** by the type

$$\Theta = \Pi X.(C_1 \to X) \to \cdots \to (C_1 \to X) \to X.$$

In fact, the constructors of Θ may be represented by terms of system **F**, as follows.

In the sequel, we write $x^T \vdash M : U$ to say that M is a system **F** term of type U containing a distinguished free variable x^T (of type T). Given a type C in which X occurs positively only, a type C' in which X occurs negatively only and a term $x^T \vdash M : U$, we build terms

$$\begin{aligned} x^{C[T/X]} &\vdash C(\!|M|\!) : C[U/X], \\ x^{C'[U/X]} &\vdash C(\!|M|\!) : C'[T/X], \end{aligned}$$

by induction on C and C':

- if C is an atom, there are two cases:
  - $C = X$, in which case $C(\!|M|\!) = M$;
  - $C = Y$, in which case $C(\!|M|\!) = x^Y$, independently of M.

  If C' is atom, only the second case can apply.
- if $C = D' \to E$, then observe that X must occur negatively only in D' and positively only in E. This means that we know how to inductively define

  $$\begin{aligned} x^{E[T/X]} &\vdash E(\!|M|\!) : E[U/X], \\ x^{D'[U/X]} &\vdash D'(\!|M|\!) : D'[T/X], \end{aligned}$$

  from which we set

  $$C(\!|M|\!) = \lambda y^{D'[U/X]}.E(\!|M|\!)[x^{C[T/X]}D'(\!|M|\!)[y/x]/x].$$

  If $C' = D \to E'$, the definition is symmetric: just replace D' with D and E with E'.
- if $C = \Pi Y.D$, then X occurs only positively in D hand we know how to inductively define

  $$x^{D[T/X]} \vdash D(\!|M|\!) : D[U/X].$$

  From this, we set

  $$C(\!|M|\!) = \Lambda Y.D(\!|M|\!)[x^{\Pi Y.D[T/X]}Y/x].$$

  If $C' = \Pi Y.D'$, the situation is again similar: just replace D with D' and T with U.

Let now $C_1, \ldots, C_n$ be types containing X positively, as above. For $1 \leq i \leq n$, we set

$$M_i = x^{\Theta} X s_1^{C_1 \to X} \cdots s_n^{C_n \to X},$$

which is of type X. Using our notation, we have $x^{\Theta} \vdash M_i : X$, so we may build

$$x^{C_i[\Theta/X]} \vdash C_i(\!|M|\!) : C_i,$$

for all $1 \leq i \leq n$. From these, we define the terms

$$c_i = \lambda x^{C_i[\Theta/X]}.\Lambda X.\lambda s_1^{C_1 \to X} \ldots s_n^{C_n \to X}.s_i C_i(\!|M_i|\!),$$

which are of type $C_i[\Theta/X] \to \Theta$ and represent the constructor c_i of the free structure Θ.

As evidence of the fact that we have actually found a representation of the free structure in question, we give a general construction for inductively defining functions on the type Θ. Suppose we are given terms $H_i : C_i[T/X] \to T$ for all $1 \leq i \leq n$ and for some type T. We would like to define a term $x^{\Theta} \vdash F : T$ such that, for all $N : C_i[\Theta/X]$,

$$(\lambda x^{\Theta}.F)(c_i N) \to^* H_i(C_i(\!|F|\!)[N/x]),$$

which is what we expect if $\lambda x^{\Theta}.F$ has to represent the function of type $\Theta \to T$ defined by induction on Θ from $H_1, \ldots, H_n$. It is easy to verify that the solution is given by setting

$$F = x^{\Theta} T H_1 \cdots H_n.$$

Let us apply the construction to the examples given in the previous section:

1. if $n = 0$, then $\Theta = \Pi X.X$, which is the empty type, corresponding to absurdity ($\forall X.X$) and representing the empty set.

2. If $C_1 = T$ and $C_2 = U$, we have $\Theta = \Pi X.(T \to X) \to (U \to X) \to X$, corresponding to the polymorphic/second order encoding of the sum type/ disjunction.

   The special case of $T = 1$ (we shall see what is the definition of 1 in a moment) deserves attention. Since 1 logically corresponds to truth and $1 \Rightarrow X$ is the same as X, we may simplify the type $\Pi X.(1 \to X) \to (1 \to X) \to X$ into

   $$\mathbf{B} = \Pi X.X \to X \to X,$$

   which is the type of Booleans in system **F**. The two normal terms of type **B** are $\Lambda X.\lambda x^X y^X.x$ and $\Lambda X.\lambda x^X y^X.y$, representing true and false, respectively.

3. Two special cases are of interest. First, let $K = 1$. By operating the same simplification as above, we obtain the type

   $$\mathbf{U} = \Pi X.X \to X,$$

   which is the unit type, corresponding to the singleton set 1. Its only normal term is $\Lambda X.\lambda x^X.x$.

   The second case is $K = T \times U$. Since $(T \times U) \to X$ may be considered to be isomorphic to $T \to U \to X$, we get the type

   $$\Theta = \Pi X.(T \to U \to X) \to X,$$

   which is exactly the polymorphic/second order encoding of the product type/conjunction.

4. This case will be covered in detail in the next section.

5. If $C_1 = 1$ and $C_2 = T \times X$, with the simplifications $1 \to X = X$ and $(T \times X) \to X = T \to X \to X$, we get

$$\Pi X.X \to (T \to X \to X) \to X,$$

which is the type of lists of elements of type T.

6. If C_1 is as above and $C_2 = X \times X$, with the same simplifications we get the type of binary trees:

$$\Pi X.X \to (X \to X \to X) \to X.$$

7. Similarly, if C_1 is as above and $C_2 = T \to X$, we get the type of tree of branching type T:

$$\Pi X.X \to ((T \to X) \to X) \to X.$$

The last three cases are explained in detail in [GLT89]. Here, we shall content ourselves with giving the details of case 4, which is the most computationally relevant.

10.3.3 The case of integers: representable functions

If we apply the above construction, with the necessary optimizations, to the case of the two functor-types $C_1 = 1$ and $C_2 = X$, we obtain the type

$$\Pi X.X \to (X \to X) \to X.$$

Of course, the order of C_1, C_2 (and of $C_1, \ldots, C_n$ in general) is arbitrary, so we may as well consider the type

$$\mathbf{N} = \Pi X.(X \to X) \to X \to X,$$

which is what we shall take as the type of natural numbers in system $\mathbf{F}$.

The constructors obtained by pedantically applying the general definitions (but exchanging the order of C_1 and C_2 in the abstractions, and considering the optimization $1 \to X = X$) are

$$\begin{aligned} c_1 &= \lambda x^1.\Lambda X.\lambda s^{X\to X} z^X.z*, \\ c_2 &= \lambda x^{\mathbf{N}}.\Lambda X.\lambda s^{X\to X} z^X.s(xXsz). \end{aligned}$$

In c_1, the type 1 and the term $*$ (the only element of type 1) are purely formal; we may remove them in accordance with our optimization $1 \to X = X$ and obtain

$$\underline{0} = \Lambda X.\lambda s^{X\to X} z^X.z.$$

If we write $\mathrm{succ} = c_2$, we invite the reader to check that

$$\overbrace{\mathrm{succ}(\ldots\mathrm{succ}}^{n}\ \underline{0}\ldots) \to^* \Lambda X.\lambda s^{X\to X} z^X.\ \overbrace{s(\ldots s}^{n}\ z\ldots),$$

which, if we forget the type annotations, is exactly the so-called *Church numeral* $\underline{n}$, one of the standard representations of the natural number n in the λ-calculus. One can prove that all normal terms of type $\mathbf{N}$, with the exception of $\Lambda X.\lambda s^{X\to X}.s$, are of this form.[2] So the constructor $\underline{0}$ is nothing but the representation of 0, and the constructor succ represents the successor function, as expected from a representation of the natural numbers as a free structure.

The general induction scheme gives us, for any type T, $G : T$ and $H : T \to T$, the function

$$F = \lambda x^{\mathbf{N}}.xTHG : \mathbf{N} \to T,$$

which is immediately seen to represent the function $f(n) = h^n(g)$, if we consider T as a set, $g \in T$ is h a function on T.

From iteration, using polymorphism, we may obtain primitive recursive definitions. We recall that primitive recursion lets us define, from functions $g : \mathbb{N}^k \to \mathbb{N}$ and $h : \mathbb{N}^{n+2} \to \mathbb{N}$, the function $f : \mathbb{N}^{k+1} \to \mathbb{N}$ by

$$f(x, \vec{y}) = \begin{cases} g(\vec{y}) & \text{if } x = 0, \\ h(x-1, \vec{y}, f(x-1, \vec{y})) & \text{if } x > 0. \end{cases}$$

So suppose we have

$$\begin{aligned} G &: \overbrace{\mathbf{N} \to \cdots \to \mathbf{N}}^{n} \to \mathbf{N}, \\ H &: \mathbf{N} \to \overbrace{\mathbf{N} \to \cdots \to \mathbf{N}}^{n} \to \mathbf{N} \to \mathbf{N}, \end{aligned}$$

corresponding to the functions g and h, respectively. We shall use product types, which we have seen are definable in system $\mathbf{F}$. With these, we define

$$\begin{aligned} \widehat{G} &= \langle G y_1^{\mathbf{N}} \dots y_n^{\mathbf{N}}, \underline{0} \rangle : \mathbf{N} \times \mathbf{N}, \\ \widehat{H} &= \lambda p^{\mathbf{N}\times\mathbf{N}}.\langle H(\pi_2^{\mathbf{N},\mathbf{N}} p) y_1^{\mathbf{N}} \dots y_n^{\mathbf{N}} (\pi_1^{\mathbf{N},\mathbf{N}} p), \mathrm{succ}(\pi_2^{\mathbf{N},\mathbf{N}} p) \rangle : \mathbf{N} \times \mathbf{N} \to \mathbf{N} \times \mathbf{N}, \end{aligned}$$

and then we set

$$F = \lambda x^{\mathbf{N}} y_1^{\mathbf{N}} \dots y_n^{\mathbf{N}}.\pi_1^{\mathbf{N},\mathbf{N}}(x(\mathbf{N} \times \mathbf{N})\widehat{H}\widehat{G}) : \mathbf{N} \to \overbrace{\mathbf{N} \to \cdots \to \mathbf{N}}^{n} \to \mathbf{N}.$$

We let the reader check that F represents indeed the function f as defined above. The idea is that the computation is done on a pair of integers (m, n), with n being the current stage of the recursion and m being the partial result at the n-th stage. Computation starts with the pair $(g(\vec{y}), 0)$, and ends by projecting on the first component of the pair. If the argument leading the recursion is 0, we immediately obtain the desired result. Otherwise, one application of the term $\widehat{H}$ gives us the pair $(h(0, \vec{y}, g(\vec{y})), 1)$. If the argument leading the recursion is 1, we can stop and obtain the expected result. Otherwise, we reapply $\widehat{H}$, which gives us $(h(1, \vec{y}, h(0, \vec{y}, g(\vec{y})), 2)$, and so on.

Now that we know how to represent integers, we may give the following definition:

[2] See [GLT89], Sect. 15.1.1 for a proof.

Definition 10.3 *A partial function* $f : \mathbb{N} \to \mathbb{N}$ *is* representable *in system* $\mathbf{F}$ *if there is a term* $F : \mathbf{N} \to \mathbf{N}$ *such that, for every* $x \in \mathbb{N}$*, if* x *is in the domain of* f*, we have*

$$M\underline{x} \to^* \underline{f(x)},$$

and if x *is not in the domain of* f*,* $M\underline{x}$ *must have no normal form.*

We have seen so far that in system $\mathbf{F}$ we can represent at least all primitive recursive functions. In fact, we can do much, much more. However, there are limits, the biggest one being that we may only represent *total* functions. In other words, the second clause of Definition 10.3 never applies. This is because, as we shall see, *every term of system* $\mathbf{F}$ *is normalizable.*

Chapter 11

Normalization of System F

11.1 Towards normalization: the reducibility technique

A *normalization theorem* is a result stating that every object of a rewriting system has a normal form. There is a very simple and elegant proof of the normalization theorem for the simply typed λ-calculus, due to Tait [Tai67]. By the Curry-Howard isomorphism, and by the correspondence between normalization in natural deduction and cut-elimination in sequent calculus, this is actually an alternative proof of the cut-elimination theorem for propositional **LJ**. For simplicity, we shall restrict to the implication fragment only (so to **NM** and **LM**).

In the following, we call *neutral* a term which does not start with an abstraction. The fundamental property of neutral terms is that they are stable under application: if N is neutral, then for every M the term NM is still neutral.

For each simple type T (which, we remind, is either an atom X or of the form $U \to V$), we define:

- $\mathrm{Norm}(T)$ as the set of all normalizable terms of type T;
- $\mathrm{Neut}(T)$ as the set of all normal neutral terms of type T.

We also define the set of *reducible* terms of type T, denoted by $\mathrm{Red}(T)$, by induction on T, as follows:

- $\mathrm{Red}(X) = \mathrm{Norm}(X)$;
- $\mathrm{Red}(U \to V) = \{M : U \to V \mid \forall N \in \mathrm{Red}(U), MN \in \mathrm{Red}(V)\}$.

The normalization theorem is a consequence of the Adequacy and Adaptation Lemmas, given below. Each of them requires one auxiliary result.

Lemma 11.1 *For every type T, $\mathrm{Red}(T)$ is closed under β-expansion, i.e., if $M \in \mathrm{Red}(T)$ and $M_1 \to M$, then $M_1 \in \mathrm{Red}(T)$.*

Proof. By induction on T. For the atomic case, obviously normalizable terms are closed under β-expansion. So let $T = U \to V$, let $M \in \mathrm{Red}(U \to V)$ and let $M_1 \to M$. Take any $N \in \mathrm{Red}(U)$. To conclude, it is enough to show that

$M_1 N \in \mathrm{Red}(V)$. But $M_1 N \to MN \in \mathrm{Red}(V)$, so we conclude by the induction hypothesis. □

Lemma 11.2 (Adequacy) *Let $M : U$ and let $x_1^{T_1}, \ldots, x_n^{T_n}$ be the free variables of M. For all $N_1 \in \mathrm{Red}(T_1), \ldots, N_n \in \mathrm{Red}(T_n)$, we have*

$$M[N_1/x_1, \ldots, N_n/x_n] \in \mathrm{Red}(U).$$

Proof. By induction on M. We shall use the shorthand notation $P[\vec{N}/\vec{x}]$ for denoting the simultaneous substitution of every N_i to x_i in a term P.

If $M = x^U$, the result is trivial.

Let $M = \lambda y^V.M'$, with $U = V \to V'$ and $M' : V'$. Let $P \in \mathrm{Red}(V)$. The induction hypothesis gives us $M_1 = M'[\vec{N}/\vec{x}][P/y] \in \mathrm{Red}(V')$. But $M[\vec{N}/\vec{x}]P \to M_1$, so we conclude by applying Lemma 11.1.

Let $M = PQ$, with $P : V \to U$ and $Q : V$. If we set $P' = P[\vec{N}/\vec{x}]$, $Q' = Q[\vec{N}/\vec{x}]$, using the induction hypothesis we have $P' \in \mathrm{Red}(V \to U)$, $Q' \in \mathrm{Red}(V)$. Then, by definition $M[\vec{N}/\vec{x}] = P'Q' \in \mathrm{Red}(U)$. □

The following is a simple property of λ-terms, which holds regardless of types (and we therefore formulate it without them).

Lemma 11.3 *For every λ-term M, if Mx is normalizable, then so is M.*

Proof. Consider a reduction $Mx \to^* N$, where N is a normal form. If all reducts of M (including M itself) are neutral, then we actually have $N = N'x$, with N' normal and neutral, and $M \to^* N'$, so we conclude. Otherwise, we must have $Mx \to^* (\lambda x.P)x \to P \to^* N$, which means that $M \to^* \lambda x.P \to^* \lambda x.N$, which is normal because N is. □

Lemma 11.4 (Adaptation) *For every type T,* $\mathrm{Neut}(T) \subseteq \mathrm{Red}(T) \subseteq \mathrm{Norm}(T)$.

Proof. Again, by induction on T. The atomic case is immediate from the definition. Let $T = U \to V$, and let $N : U \to V$ be normal and neutral. Take $M \in \mathrm{Red}(U)$. It is enough to show that $NM \in \mathrm{Red}(V)$. By definition, NM is still neutral. By the induction hypothesis, M is normalizable. If M_0 is its normal form, we have (using once more the induction hypothesis) $NM_0 \in \mathrm{Norm}(V) \subseteq \mathrm{Red}(V)$, so we conclude by applying Lemma 11.1.

Let now $M \in \mathrm{Red}(U \to V)$. We want to show that M is normalizable. Every variable is obviously normal and neutral, so by the induction hypothesis $x^U \in \mathrm{Red}(U)$. But by definition, $Mx^U \in \mathrm{Red}(V)$, so by the induction hypothesis it is normalizable and Lemma 11.3 allows us to conclude. □

Theorem 11.5 (Normalization for simple types) *Every simply-typed term is normalizable.*

Proof. Let $M : U$, with free variables $x_1^{T_1}, \ldots, x_n^{T_n}$. By the first inclusion of the Adaptation Lemma 11.4, for all $1 \leq i \leq n$, $x_i^{T_i} \in \mathrm{Red}(T_i)$. Then, the Adequacy Lemma 11.2 gives us that $M[x_1^{T_1}/x_1, \ldots, x_n^{T_n}/x_n] = M \in \mathrm{Red}(U)$, so we conclude by using the second inclusion of the Adaptation Lemma 11.4. □

11.2 Reducibility and polymorphism

The reducibility technique is very powerful. After minor adjustments, it can be used to prove the normalization theorem of Gödel's system **T**, which is not provable in first-order Peano arithmetic [GLT89]. It may also be adapted to prove the *strong normalization* theorem, that is to say that *every* reduction sequence eventually terminates (while normalization simply states that there is always *at least one* terminating sequence).

However, to make it work for system **F**, non-trivial modifications are needed. We immediately realize this by attempting to define $\mathrm{Red}(\Pi X.U)$. The intuition would be to set

$$\mathrm{Red}(\Pi X.U) = \{M : \Pi X.U \mid \forall V,\ MV \in \mathrm{Red}(U[V/X])\},$$

where V should range over *every* type, because any type may be substituted to X in $\Pi X.U$ (this is the meaning, and power, of polymorphism!). But then the definition is obviously ill-founded, because $U[V/X]$ may be much more complex than $\Pi X.U$ (think of $V = \Pi X.U$ itself...).

Girard solved this apparently inextricable tangle, which is due to the intrinsic impredicativity of second order quantification, by means of his so-called *reducibility candidates*. Instead of directly defining reducibility of a given type, we give an abstract notion of what a set of reducible terms should look like (a "candidate"), independently of types (and therefore of induction). Then, we define a *parametric* reducibility, in terms of a *valuation* which assigns a reducibility candidate to every free atom of the type under consideration. In the second-order case, instead of letting the type vary over all possible instances (which is incompatible with induction), we shall let the valuation vary over all possible assignments, which is sound because valuations are not defined by induction.

The key is to identify which abstract properties characterize reducibility. Originally, Girard found the right conditions by trial and error [GLT89]. Here, we are able to identify them by inspecting the proof of Theorem 11.5: the only two properties we ever need to show about the sets $\mathrm{Red}(T)$ is that they are closed under β-expansion (Lemma 11.1) and that they enjoy the adaptation conditions (Lemma 11.4). The other lemmas and parts of the proof, although of course necessary, are not properties of the sets $\mathrm{Red}(T)$.

11.3 The reducibility candidates technique

We start by extending the definition of neutrality:

Definition 11.1 (Neutrality for system F) *A term is neutral if it is neither an abstraction nor a universal abstraction.*

The sets $\mathrm{Norm}(T)$ and $\mathrm{Neut}(T)$ are defined just as in the simply-typed λ-calculus (with the updated notion of neutrality, of course).

Definition 11.2 (Reducibility candidate) *A* reducibility candidate *of type T is a set R of terms of type T such that:*

CR1. *R is closed under β-expansion;*

CR2. $\mathrm{Neut}(T) \subseteq R \subseteq \mathrm{Norm}(T)$.

We denote by $\mathrm{CR}(T)$ *the set of all reducibility candidates of type* T*.*

For any type T, the set $\mathrm{Norm}(T)$ is the prototypical reducibility candidate of type T. Therefore, $\mathrm{CR}(T)$ is never empty.

Definition 11.3 (Valuation) *A* valuation *is a partial function* ρ *from atomic types to reducibility candidates, whose domain is finite and denoted by* $\mathrm{dom}\rho$*.*

If R *is a reducibility candidate, we denote by* $\rho[X := R]$ *the valuation whose domain is* $\mathrm{dom}\rho \cup \{X\}$ *and which coincides with* ρ *everywhere except on* X*, where it is equal to* R*.*

A valuation ρ covers *a type* T *if the free atoms of* T *are all in* $\mathrm{dom}\rho$*; it* covers *a term* $M : U$ *whose free variables are* $x_1^{T_1}, \ldots, x_n^{T_n}$ *if it covers every one of* $U, T_1, \ldots, T_n$*.*

The default valuation *on* $\Gamma = X_1, \ldots, X_n$ *is the valuation* ρ_Γ *such that* $\mathrm{dom}\rho_\Gamma = \{X_1, \ldots, X_n\}$ *and such that* $\rho_\Gamma(X_i) = \mathrm{Norm}(X_i)$ *for all* $1 \leq i \leq n$*.*

Definition 11.4 (Substitution induced by a valuation) *Let* ρ *be a valuation, and let* X *be an atomic type. We set*

$$X^\rho = \begin{cases} U & \text{if } X \in \mathrm{dom}\rho \text{ and } U \text{ is the type of } \rho(X); \\ X & \text{if } X \notin \mathrm{dom}\rho. \end{cases}$$

Let T *be a type whose free atoms are* $X_1, \ldots, X_n$*. We set*

$$T^\rho = T[X_1^\rho / X_1, \ldots, X_n^\rho / X_n].$$

Similarly, if $M : T$ *and* $X_1, \ldots, X_n$ *are all the free atoms appearing in* T *and in the types of the free variables of* M*, we define*

$$M^\rho = M[X_1^\rho / X_1, \ldots, X_n^\rho / X_n],$$

which is of type T^ρ*.*

Observe that, if $M : T$ and if Γ contains all the free atoms appearing in M and T, then $T^{\rho_\Gamma} = T$ and $M^{\rho_\Gamma} = M$, *i.e.*, the default valuation induces the identity substitution.

Given a type T and a valuation ρ covering T, we define the set of terms $\mathrm{Red}_\rho(T)$ by induction on T:

- $\mathrm{Red}_\rho(X) = \rho(X)$;
- $\mathrm{Red}_\rho(U \to V) = \{M : (U \to V)^\rho \mid \forall N \in \mathrm{Red}_\rho(U),\ MN \in \mathrm{Red}_\rho(V)\}$;
- $\mathrm{Red}_\rho(\Pi X.U) = \{M : (\Pi X.U)^\rho \mid \forall V\, \forall R \in \mathrm{CR}(V),\ MV \in \mathrm{Red}_{\rho[X:=R]}(U)\}$.

Lemma 11.6 *Let* $M : \Pi X.U$*, let* V *be a type, and suppose that* MV *is normalizable. Then,* M *is normalizable.*

Proof. Entirely similar to the proof of Lemma 11.3. □

Lemma 11.7 *For every type* T *and valuation* ρ *covering* T*,* $\mathrm{Red}_\rho(T) \in \mathrm{CR}(T^\rho)$*.*

PROOF. The fact that all terms in $\mathrm{Red}_\rho(T)$ have type T^ρ is immediate from the definition. Properties CR1 and CR2 are verified by induction on T.

The atomic case is trivial. For the arrow case, we follow exactly the arguments given for Lemma 11.1 (for CR1) and Lemma 11.4 (for CR2). So we may assume that $T = \Pi X.U$.

For CR1, suppose that $M_1 \to M \in \mathrm{Red}_\rho(T)$. Take any type V and any $R \in \mathrm{CR}(V)$. We have $M_1 V \to MV \in \mathrm{Red}_{\rho[X:=R]}(U)$ by hypothesis, so we infer $M_1 V \in \mathrm{Red}_{\rho[X:=R]}(U)$ by applying CR1, which holds for U. But, by genericity of V and R, this is enough to conclude that $M_1 \in \mathrm{Red}_\rho(T)$.

For what concerns the first inclusion of CR2, let $N \in \mathrm{Neut}(T^\rho)$. Take any type V and any $R \in \mathrm{CR}(V)$. Since N is neutral and normal, NV is also neutral and normal, of type $U^{\rho[X:=R]}$. But CR2 holds for U, so $NV \in \mathrm{Red}_{\rho[X:=R]}(U)$, which is enough to show that $N \in \mathrm{Red}_\rho(T)$.

Let us turn to the second inclusion of CR2. Let $M \in \mathrm{Red}_\rho(T)$, let V be any type and $R \in \mathrm{CR}(V)$. Since CR2 holds for U, we know that $MV \in \mathrm{Red}_{\rho[X:=R]}(U) \subseteq \mathrm{Norm}(U^{\rho[X:=R]})$. But then Lemma 11.6 allows us to conclude that $M \in \mathrm{Norm}(T^\rho)$. □

Lemma 11.8 (Substitution) *For every type V, W and every assignment ρ covering them, we have* $\mathrm{Red}_\rho(V[W/Y]) = \mathrm{Red}_{\rho[Y:=\mathrm{Red}_\rho(W)]}(V)$.

PROOF. A straightforward induction on V. □

Lemma 11.9 (Adequacy) *Let $M : U$ and let $x_1^{T_1}, \ldots, x_n^{T_n}$ be the free variables of M. For every valuation ρ covering M and for every $N_1 \in \mathrm{Red}_\rho(T_1), \ldots, N_n \in \mathrm{Red}_\rho(T_n)$, we have*

$$M^\rho[N_1/x_1, \ldots, N_n/x_n] \in \mathrm{Red}_\rho(U).$$

PROOF. By induction on M. We shall use the shorthand notation $P[\vec{N}/\vec{x}]$ for denoting the simultaneous substitution of every N_i to x_i in a term P.

If $M = x^U$, the result is trivial.

Let $M = \lambda y^V.M_1$, with $U = V \to V_1$ and $M_1 : V_1$. Let $P \in \mathrm{Red}_\rho(V)$. The induction hypothesis gives us $M_2 = M_1^\rho[\vec{N}/\vec{x}][P/y] \in \mathrm{Red}_\rho(V_1)$. But $M^\rho[\vec{N}/\vec{x}]P \to M_2$, so we conclude by CR1 (which holds thanks to Lemma 11.7).

Let $M = PQ$, with $P : V \to U$ and $Q : V$. If we set $P_1 = P^\rho[\vec{N}/\vec{x}]$, $Q_1 = Q^\rho[\vec{N}/\vec{x}]$, using the induction hypothesis we have $P_1 \in \mathrm{Red}_\rho(V \to U)$, $Q_1 \in \mathrm{Red}_\rho(V)$. Then, by definition $M^\rho[\vec{N}/\vec{x}] = P_1 Q_1 \in \mathrm{Red}_\rho(U)$.

Let $M = \Lambda Y.M_1$, with $U = \Pi Y.U_1$ and $M_1 : U_1$. Note that, by α-equivalence, we may always suppose $Y \notin \mathrm{dom}\rho$. Let V be any type, let $R \in \mathrm{CR}(V)$, and let $M_2 = M_1^{\rho[Y:=R]}[\vec{N}/\vec{X}]$. The induction hypothesis gives us $M_2 \in \mathrm{Red}_{\rho[Y:=R]}(U_1)$. But by definition $M_1^{\rho[Y:=R]} = M_1^\rho[V/Y]$, so $M^\rho[\vec{N}/\vec{x}]V \to M_2$, which is enough to prove $M^\rho \in \mathrm{Red}_\rho(U)$.

Let $M = PW$ with $P : \Pi Y.V$ and W a type, so that $U = V[W/Y]$. Let $P_1 = P^\rho[\vec{N}/\vec{X}]$. By induction, we know that $P_1 \in \mathrm{Red}_\rho(\Pi Y.V)$, which gives us that $P_1 W^\rho \in \mathrm{Red}_{\rho[Y:=\mathrm{Red}_\rho(W)]}(V)$. But $P_1 W^\rho = M^\rho[\vec{N}/\vec{x}]$, so we conclude by applying Lemma 11.8. □

Theorem 11.10 (Normalization of system F) *Every term of system* **F** *is normalizable.*

PROOF. Let $M : U$ be a system **F** term, whose free variables are $x_1^{T_1}, \ldots, x_n^{T_n}$ and such that the free atoms of $U, T_1, \ldots, T_n$ are in the list Γ. By the first inclusion of CR2, we have $x_i^{T_i} \in \mathrm{Red}_{\rho_\Gamma}(T_i)$ (remember that $T_i^{\rho_\Gamma} = T_i$). Hence, by the Adequacy Lemma 11.9, we have $M^{\rho_\Gamma}[x_1^{T_1}/x_1, \ldots, x_n^{T_n}/x_n] = M \in \mathrm{Red}_{\rho_\Gamma}(U)$, so we conclude by the second inclusion of CR2. □

Bibliography

[GLT89] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*. Cambridge University Press, 1989.

[NvP01] Sara Negri and Jan von Plato. *Structural Proof Theory*. Cambridge University Press, 2001.

[Tai67] William W. Tait. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic*, 32(2):198–212, 1967.