Mathematical Logic 1

0525250 Christoph Redl[1]

Winter term 2008/2009
Version 1.10, 26th of January, 2009

[1]E-mail: e0525250@mail.student.tuwien.ac.at

Contents

Preface

This document is a digitalization of the course's notes of Agata Ciabattoni's lecture on Mathematical Logic 1. It was written by Christoph Redl in winter term 2008/09 an is an inofficial script which was created during the preparation for the written exam, but not reviewed by the lecturer. Therefore it is provided as is without any warranties for completeness or correctness.

The sources are mentioned below. However, it is mainly based on my own notes.

Chapter 1

Classical Propositional Logic

Classical propositional logic is also well known as zeroth-order-logic oder PL0. The following subsections introduce syntax and semantics of PL0.

1.1 Syntax

PL0 formulas are defined by the minimum set WFF s.t.:

1. Atoms $A, B, C, ..., \bot \in WFF$
2. if $P \in WFF$ then $\neg P \in WFF$
3. if $P \in WFF$ and $Q \in WFF$ then $P \wedge Q, P \vee Q, P \supset Q \in WFF$

Additional connectives and the logical constant $\top$ can be introduced in order to provide a more comfortable syntax. However, the listet connections are sufficiant since they fulfill a property called *functional completeness*. Alternative sets of connectives may be function complete too, e.g. *nand*.

1.2 Semantics

PL0 is 2-valued whereas the the values are often referred to as 0/1, true/false, etc..

An interpretation e in PL0 is a mapping of well-formed formulas to truth values: $e : WFF \mapsto \{0, 1\}$. e is defined individually for atom formulas by assigning a truth value to them. The evaluation of bottom is: $e(\bot) = 0$. The evaluation of composed formulas is done with the following semantics:

- $e(\neg P) = 1$ iff $e(P) = 0$
- $e(P \wedge Q) = 1$ iff $e(P) = 1$ and $e(Q) = 1$

- $e(P \vee Q) = 1$ iff $e(P) = 1$ or $e(Q) = 1$
- $e(P \supset Q) = 1$ iff $e(P) = 0$ or $e(Q) = 1$

Definition. *A model of a formula* $F \in WFF$ *is an interpretation* $e : WFF \mapsto \{0, 1\}$ *with* $e(F) = 1$.

Definition. *A formula* $F \in WFF$ *is satisfiable iff there exists an* $e : WFF \mapsto \{0, 1\}$ *with* $e(F) = 1$.

Definition. *A formula* $F \in WFF$ *is valid iff* $e(F) = 1$ *for arbitrary* $e : WFF \mapsto \{0, 1\}$

Definition. *A formula* $F \in WFF$ *is unsatisfiable iff* $e(F) = 0$ *for arbitrary* $e : WFF \mapsto \{0, 1\}$

Definition. *Semantical consequence:* $\gamma \models P$ *iff* $\exists A_i \in \gamma : e(A_i) = 0$ *or* $e(P) = 1$

Theorem. $\gamma \models P$ *iff* $\gamma \cup \{\neg P\}$ *is unsatisfiable*

Proof. The equivalence has to be shown in both directions.
$\Longrightarrow$
Case 1: $e(\gamma) = 1$, then also $e(P) = 1$ by definition of $\models$. Therefore $e(\neg P) = 0$ and $\gamma \cup \{\neg P\}$ is unsatisfiable
Case 2: $e(\gamma) = 0$, then $\gamma \cup \{\neg P\}$ is unsatisfiable
$\Longleftarrow$
Case 1: $\gamma \cup \{\neg P\}$ is unsatisfiable because $e(\neg P) = 0$. Then $e(P) = 1$ and $\gamma \models P$ holds independent from $e(\gamma)$
Case 2: $\gamma \cup \{\neg P\}$ is unsatisfiable because $e(\gamma) = 0$, then $\gamma \models P$ trivially holds □

1.3 Theorems

The following subsections present two important theorems in classical propositional logic: the *Compactness Theorem* and *Cook's Theorem*. The former is especially important in it's negated form whereas the latter implies that the satisfiability problem in PL0 is already NP-complete.

1.3.1 Compactness Theorem

Theorem. *A set* Γ *of well-formed formulas is satisfible iff every finite subset* Δ *of* Γ *is satisfiable.* Γ *may be infinite.*

Proof. The equivalence is shown in both directions.
$\Longrightarrow$ If Γ is satisfiable, then all subsets are trivially satisfiable since each model of Γ is also a model of $A \in \Gamma \forall A$.
$\Longleftarrow$ The proof is done by induction on the number of distinct atoms in a formula. Let M_i be the set of WFF which contain only the first i atoms. Then $M_i \subset WFF$ and thus has a model due to our premiss.
Note that M_i will contain at most 2^i formulars with different truth tables. Reduce M_i to M_i' which consists of just all formulas of M_i with different truth tables. Each model of M_i' will also be a model of M_i. The latter is therefore also satisfible due to our premiss.
Since there is only a finite number of atoms, say n, M_n' is also finite with at most 2^n elements. Therefore M_n is satisfiable. Note that in fact $M_n = \Gamma$. $\square$

The Compactness Theorem states that one can make a proposition about the satisfiability of an infinite set of formulars by considering only it's finite subsets. The negated form of the theorem is even more important, which says that an infinite set of formulas γ is unsatisfiable if one can show that *at least one* finite subset $A \in \gamma$ is unsatisfiable.

1.3.2 Cook's Theorem

Cook's theorem states that the satisfiability problem in classical propositional logic is already NP-complate.

Theorem. *PL0 SAT is NP-complete*

Proof. The proof is done by reducing the semantics of an arbitrary decision problem which is computable in bounded time to the satisfiability problem. Since decision problems (even with bounded time of computation) are in general NP-complete it follows that SAT can not have a lower complexity.

Assume that there exists a Turing Machine $M = (Q, \Sigma, s, F, \delta)$ which decides a given language L, i.e. ich checks whether $w \in L$ for it's input word w. Q is the set of states, Σ the tape alphabet, $s \in Q$ the start state, $F \subseteq Q$ the set of final states and $\delta \subseteq Q \times \Sigma \times Q \times \Sigma \times \{-1, +1\}$ the (nondeterministic) transition relation.

We now map the question if M will finally reach a $f \in F$ to the SAT in PL0.

Define the following propositional variables:

- $T_{ijk} = 1$ iff tape cell i contains symbol j at step k of computation
- $H_{ik} = 1$ iff the head is over cell i at step k of computation

- $Q_{qk} = 1$ iff the machine is in state q at step k of computation

Assume that we have a computation with n steps. Then our initial condition is $I = Q_{s0} \wedge H_{00} \wedge T_{0\triangle 0}$ where $\triangle$ denotes the start symbol.

The final states are defined as $X = \bigvee_{f \in F} Q_{fn}$

The transition relation is mapped to: $R = \bigwedge(T_{i\sigma k} \wedge H_{ik} \wedge Q_{qk}) \supset ((T_{i\sigma'(k+1)} \wedge H_{(i+1)(k+1)} \wedge Q_{q'(k+1)}) \forall (q, \sigma, q', \sigma', d) \in \delta$.

Additionally we need conditions to state that the machine is in exactly one state, that the head is over exactly one cell, that each cell contains exactly one symbol and a cell is only changed by the read/write head:

$C_1 = \bigwedge T_{ijk} \supset \neg T_{ij'k} \forall j' \neq j$
$C_2 = \bigwedge H_{ik} \supset \neg H_{i'k} \forall i' \neq i$
$C_3 = \bigwedge Q_{qk} \supset \neg Q_{q'k} \forall q' \neq q$
$C_4 = \bigwedge T_{ijk} \supset (T_{ij(k+1)} \vee H_{ik})$

The question if the machine will reach a final state after n steps of computation is now equivalent to the question if $I \wedge X \wedge R \wedge C_1 \wedge C_2 \wedge C_3 \wedge C_4$ is satisfiable. □

It directly follows that TAUT (tautology problem) in PL0 is CoNP-complete. Note that if a decision problem P is in complexity class C, then the negated problem $\neg P$ is in class $Co - C$.

Theorem. *PL0 TAUT is CoNP-complete*

Proof. Since "Is a formula F a tautology?" is equivalent to "Is a formula $\neg F$ unsatisfiable?", which is the negated question of SAT, we see that $TAUT$ is in CoNP. □

Note: It is still unknown if $NP = CoNP$ or $NP \neq CoNP$.

Chapter 2

Lukasiewicz Logic

2.1 Introduction

Lukasiewicz Logic is a generalization of the two-valued classical propositional logic to n levels of truth. The syntax is the same as in PL0. However, the semantics is defined differently.

2.2 Semantics

An interpretation e is a mapping $e : WFF \mapsto \{0, 1/n, 2/n, ...(n-1)/n, 1\}$ for an (n+1)-valued Lukasiewicz Logic.

The evaluation of a formula is done in the following way:

- $e(\neg A) = 1 - e(A)$
- $e(A \wedge B) = max\{0, e(A) + e(B) - 1\}$
- $e(A \vee B) = max\{e(A), e(B)\}$
- $e(A \rightarrow B) = min\{1, 1 - e(A) + e(B)\}$

Note that for $n = 1$, Lukasiewicz Logic is the same as PL0 and for "$n = \infty$" it coincides with Fuzzy Logic.

2.3 Twenty Questions

Imagine a game with two players A and B. A selects a number in the range from 1 to 1 million. B has to guess by questioning "Is the number smaller or equal to x, or is it larger?".

Obviously B can come to know the number with at most 20 questions using binary search. Formally, the problem can be stated as follows:

Let $S_i : R \mapsto \{0, 1\}$ whereby R is the legal range, i.e. $R = \{1, 2, ..., 10^6\}$. The mapping assigns 0 or 1 to each possible number with $S_i(n) = 1$ meaning that number n is still in the game concerning the answer to question i. On the contrary $S_i(n) = 0$ means that n is excluded by answer i.

Example: If B asks "Is it smaller or equal to 10?" and the answer is "no", then $S_i(1) = S_i(2) = ... = S_i(10) = 0$ and $S_i(x) = 1 \forall x \in R, x > 10$.

The state of the game after question m was answered can be derived by evaluating $S_{total} = S_1 \wedge S_2 \wedge ... \wedge S_{m-1} \wedge S_m$. The game is over as soon as $S_{total}(q) > 0$ and $S_{total}(x) = 0 \forall x \neq q$, i.e. only one number remains in the game.

Now imagine that player A is allowed to lie once. In this case a number can not be excluded from the game before it was falsified at least twice. This can be easily expressed using a 3-valued Lukasiewicz Logic. $S_{total}(n) = 1$ means that n was not falsified so far, $S_{total}(n) = 0.5$ means that n was falsified once and $S_{total}(n) = 0$ that it was falsified at least twice.

The game can be played in nearly exactly the same way as before with the only difference that B may has to ask a further question in order to make some $S_{total}(x) = 0.5$ become $S_{total}(x) = 0$ before he can pronounce the final result.

Chapter 3

Intuitionistic Logic

3.1 Introduction

Intuitionistic Logic was introduced to force a kind of "constructivism" in mathematical proofs. Classical logic allows a proof technique which is well-known as *proof by contradiction*. Despite this technique can proof something, it does not provide an algorithm to acutally do it.

Classical logic has also some tautologies which are intuitively undesired. $A \vee \neg A$ or $\neg\neg A \supset A$ are such examples. To overcome all these problems intuitionistic logic was introduced. The syntax is exactly the same as for PL0.

3.2 Semantics

In intuitionistic logic, formulas are evaluated over a so called Kripke structure. This can be drawn as a graph consisting of nodes C and edges but without cycles. This results in one or more trees defining a semi-ordering $\leq$ of the nodes.

An interpretation is a binary relation between the set of nodes C and propositional variables. It assigns a set of atoms to each of the nodes, i.e. $n \Vdash \{A_1, A_2, ..., A_n\}$. It is required that the following monotonicity condition holds: if $c \leq c'$ and $c \Vdash A$ then $c' \Vdash A$, i.e. a successor of a node must force at least the same set of atoms as it predecessor does.

The Kripke structure can now be denoted as $K = (C, \leq, \Vdash)$.

The semantics is defined in the following way:

- $c \Vdash P \vee Q$ iff $c \Vdash P$ or $c \Vdash Q$

- $c \Vdash P \wedge Q$ iff $c \Vdash P$ and $c \Vdash Q$
- $c \Vdash P \rightarrow Q$ iff $\forall c' \geq c$ it holds that if $c' \Vdash P$ then $c' \Vdash Q$
- $c \Vdash \neg P$ iff $\forall c' \geq c : c' \nVdash P$
- $c \nVdash \bot \forall c \in C$

Intuitively spoken, the semantics forces that knowledge can be expanded but never revised. While for $\vee$ and $\wedge$ the monotonicity condition of $\Vdash$ is sufficiant due to this requirement, $\supset$ and $\neg$ need additional conditions in their evaluation rules.

Note that classical propositional logic can be simulated by a Kripke structure with only one node. The single node forces all atoms which shall be set to true.

3.3 Examples

$\neg\neg A \rightarrow A$

The given formula does not hold in intuitionistic logic. Consider the following counter example:

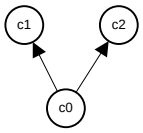
with $c_0 \Vdash \{\}$ and with $c_1 \Vdash \{A\}$

Then we see that
$c_0 \Vdash \neg\neg A$ because $c_0 \nVdash \neg A$ and $c_1 \nVdash \neg A$
($c_1 \nVdash \neg A$ because $c_1 \Vdash A$)
but
$c_0 \nVdash A$

$A \vee \neg A$

The law of the excluded middle does not hold in intuitionistic logic. We give the following counter example:

Then we see that
$c_0 \nVdash A$ and $c_0 \nVdash \neg A$ because $c_2 \Vdash A$
Therefore $c_0 \nVdash A \vee \neg A$



with $c_0 \Vdash \{\}$, $c_1 \Vdash \{\}$ and $c_2 \Vdash \{A\}$

$A \rightarrow \neg\neg A$

The given formula does hold in intuitionistic logic. If in a given node A holds: $c_0 \Vdash A$, then trivially $c_0 \Vdash \neg\neg A$ since $c_0 \nVdash \neg A$.

If a node does not force A then the formula holds trivially.

$A \rightarrow B$

This simple implementation is of course not a tautology. But it can be true in specific nodes.

with $c_0 \Vdash \{A\}$, $c_1 \Vdash \{A, B\}$

Then we see that $c_0 \nVdash A \rightarrow B$ because $c_0 \Vdash A$ but $c_0 \nVdash B$
However, $c_1 \Vdash A \rightarrow B$

Chapter 4

Proof Theory

4.1 Introduction and definitions

Proofs of logical formulas are rarely done in semantics. While this is possible in principal (e.g. by the use of truth tables in PL0), it becomes difficult in other logics like first order logic (PL1) or non-classical logics.

The better approch is to build a deductive system where the proofs can be derived in a syntactic way.

Definition. *A deductive system $\Sigma = (L, W, R, Ax)$ is a 4-tuple consisting of a language L, a set of well-formed formulas $W \subseteq L$, a finite set of r-ary relations on W: $R \subseteq W^r$ (inference rules) and a set of Axioms $Ax \subseteq W$.*

Definition. *A proof in a deductive system is a finite sequence P_m of well-formed formulas $\in W$ such that $\forall i = 1, ..., m$ either $P_i \in Ax$ or P_i is a direct consequence of $P_1, ..., Pi - 1$ through R, i.e.: $(P_i, A_1, A_2, ..., A_r) \in R$ with $A_j \in \{P_1, ..., Pi - 1\}$.*

4.2 Inference rules

In this section we define intuitive inference rules for the following calculi in an abstract form. Intuitively we see that this inference rules hold for the semantics of PL0 and intuitionistic logic. A formal connection between syntax and semantics is given later on.

Unconditional inference rules for PL0 and intuitionistic logic:

$(\wedge - e_1)$ $A \wedge B \vdash A$
$(\wedge - e_2)$ $A \wedge B \vdash B$
$(\wedge - i)$ $A, B \vdash A \wedge B$
$(\vee - i_1)$ $A \vdash A \vee B$
$(\vee - i_2)$ $B \vdash A \vee B$
$(\supset - e)$ $A, A \supset B \vdash B$ (modus ponens)
$(\bot - e)$ $\bot \supset A$

Additionally we have a set of conditional rules for PL0 and intuitionistic logic:
$(\vee - e)$ if $A \vdash C$ and $B \vdash C$ then $A \vee B \vdash C$
$(\supset - i)$ if $A \vdash B$ then $\vdash A \supset B$ (deduction theorem)
$(\neg - i)$ if $A \vdash \bot$ then $\vdash \neg A$
$(\neg - e)$ if $A, \neg A$ then $\bot$

The only difference between classical propositional logic and intuitionistic logic is the following rule which is denoted as *reductio ad absurdum*:
(RAA) if $\neg A \vdash \bot$ then $\vdash A$ (reductio ad absurdum)

Alternativly to RAA the following rule could be used since the RAA can be derived from it.
if $A \vdash B$ and $\neg A \vdash B$ then $\vdash B$

Proof. We rewrite the above rule by substituting B by A: if $A \vdash A$ and $\neg A \vdash A$ then $\vdash A$. The first condition trivially holds and we can reduce the rule to: if $\neg A, \vdash A$ then $\vdash A$. $\neg A \vdash A$ yields to $\neg A \vdash (\neg A, A) \vdash \bot$ by application of the rule $(\neg - e)$. By substituting we get: if $\neg A \vdash \bot$ then $\vdash A (RAA)$. □

4.3 Natural Deduction

4.3.1 Introduction

Natural Deduction is a proof system with many inference rules but no axioms. The above general inference rules can easily be rewritten in Natural Deduction notation:

$$\frac{\bot}{A}\,\bot - e$$

$$\frac{A \wedge B}{A}\,\wedge - e_1$$

$$\frac{\begin{matrix}[A] & [B]\\ A \vee B & C \quad C\end{matrix}}{C}\,[\vee - e]$$

$$\frac{A \quad B}{A \wedge B}\,\wedge - e_2 \qquad \frac{\begin{matrix}[A]\\ B\end{matrix}}{A \supset C}\,[\supset - i]$$

$$\frac{A}{A \vee B}\,\vee - i_1$$

$$\frac{\begin{matrix}A\\ \bot\end{matrix}}{\neg A}\,\neg - i$$

$$\frac{B}{A \vee B}\,\vee - i_2$$

$$\frac{A \quad \neg A}{\bot}\,\neg - i$$

$$\frac{A \quad A \supset B}{B}\,\supset - e$$

$$\frac{\begin{matrix}\neg A\\ \bot\end{matrix}}{A}\,RAA$$

Squared brackets denote an assumption which can be dropped later on by including this assumption into the conclusion.

Definition. *The set of derivations in Natural Deduction is the smallest set X s.t.*
(i) The one elements $P \in WFF$ are in X
(ii) if $\frac{D}{P} \in X$ and $\frac{E}{Q} \in X$ then

$$\frac{\begin{matrix}D & E\\ P & Q\end{matrix}}{P \wedge Q}\,\wedge - i$$

$\in X$, wherey $\frac{D}{P} \in X$ denotes a proof named D of P.
(...) analogiously to (ii) for all other inference rules

4.3.2 Examples

Proofs can either be read (and written) top-down or bottom-up. If a proof is read top-down, formulas enclosed in squared brackets can be interpreted as assumptions which can be dropped as soon as the assumption is included in the conclusion by rules like $\supset - i$ or RAA.

If a proof is read bottom-up, the dropping of a premiss (e.g. $\supset -i$ rule) can be thought of as pushing the premiss on a kind of "bonus stack". Whenever an element on the bonus stack occurs as a node it can be cancelled (denoted by putting squared brackets around it). Note that the stack must be managed independently for each branch of the proof.

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

$$\dfrac{\dfrac{\dfrac{\dfrac{[A \wedge B]_2}{A} \wedge -e \qquad [A \supset (B \supset C)]}{B \supset C} \supset -e \qquad \dfrac{[A \wedge B]_2}{B} \wedge -e}{C} \supset -e}{\dfrac{A \wedge B \supset C}{(A \supset (B \supset C)) \supset (A \wedge B \supset C)} [\supset -i]_2} [\supset -i]_1$$

The proof holds both for PL0 and intuitionistic logic.

$\vdash A \supset \neg\neg A$

$$\dfrac{\dfrac{\dfrac{[A]_1 \qquad [\neg A]_2}{\bot} \neg -e}{\neg\neg A} [\bot -e]_2}{A \supset \neg\neg A} [\supset -i]_1$$

The proof holds both for PL0 and intuitionistic logic.

$\vdash \neg\neg A \supset A$

$$\dfrac{\dfrac{\dfrac{[\neg\neg A]_1 \qquad [\neg A]_2}{\bot} \neg -e}{A} [RAA]_2}{\neg\neg A \supset A} [\supset -i]_1$$

Since RAA was used, the proof only holds for PL0 but not for intuitionistic logic.

$\vdash A \vee \neg A$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[A]_2}{A \vee \neg A} \vee -i \qquad [\neg(A \vee \neg A)]_2}{\bot} \neg -e}{\neg A} [\bot -e]_2}{A \vee \neg A} \vee -i \qquad [\neg(A \vee \neg A)]_1}{\bot} \neg -e}{A \vee \neg A} [RAA]_1$$

Since RAA was used, the proof only holds for PL0 but not for intuitionistic logic.

4.4 Axiomatic Systems

4.4.1 Introduction

In constrast to Natural Deduction, Axiomatic Systems (also called Hilbert Systems) use many axioms and only very few inference rules. There exist several types of Hilbert calculi. We will consider one which uses the single rule *modus ponens*:

$$\frac{A \qquad A \supset B}{B}\, MP$$

Axiomatic Systems are important from a theoretical point of view, however they are infeasable in practice since also simple formulas have complex proofs.

4.4.2 Axioms

The axioms are derived from the generalized inference rules for PL0 and intuitionistic logic. By the use of the $\supset -i$ rule, we can rewrite each of these rules in such a way that the premiss becomes empty, e.g. instead of $A \wedge B \vdash A$ we simply write $\vdash (A \wedge B) \supset A$. If we do that for all of the rules, we get the following:

$(A_{\wedge -e_1})$	$\vdash A \wedge B \supset A$	
$(A_{\wedge -e_2})$	$\vdash A \wedge B \supset B$	
$(A_{\wedge -i})$	$\vdash A, B \supset A \wedge B$	
$(A_{\vee -i_1})$	$\vdash A \supset A \vee B$	
$(A_{\vee -i_2})$	$\vdash B \supset A \vee B$	
$(A_{\supset -e})$	$\vdash A, A \supset B \supset B$	(modus ponens)
$(A_{\perp -e})$	$\vdash \perp \supset A$	
$(A_{\vee -e})$	$\vdash (A \supset C) \supset ((B \supset C) \supset (A \vee B \supset C))$	
$(A_{\supset -i})$	$\vdash (A \wedge (A \supset B)) \supset B$	(deduction theorem)
$(A_{\neg -i})$	$\vdash (A \supset \perp) \supset (\neg A)$	
$(A_{\neg -e})$	$\vdash (A \wedge \neg A) \supset \perp$	
(A_{RAA})	$\vdash (\neg A \supset \perp) \supset A$	(reductio ad absurdum)

We now declare these properties as *axioms*. In this way we can "simulate" our inference rules by the use of axioms.

However, in order to get these axioms we have used the $\supset -i$ rule. Therefore we have to add additional axioms which help simulating this rule. We do that inductively on the number of applications of modus ponens.

If modus ponens is never applied in a proof $A \vdash B$, then either $B = A$ or B is already an axiom. This means that $A \vdash A$ and $B \vdash A \supset B$ need

to be tautologies in our axiomatic system. Therefore we get the following additional axioms:

(I) $\vdash A \supset A$
(K) $\vdash B \supset (A \supset B)$

Now we do the inductive step. Assume that the claim holds for n applications of modus ponents. Then we show $n \mapsto (n+1)$.

$$\frac{\begin{array}{cc} \text{A} & \text{A} \\ B \supset C & B \end{array}}{C} MP$$

The derivation of the premises contain at most n applications of modus ponents. Due to induction hypothesis we know that $\vdash A \supset (B \supset C)$ and $\vdash A \supset B$ are provable. To appoint that also $A \vdash C$ shall be valid in our system, we need the following additional axiom:

(S) $\vdash (A \supset (B \supset C)) \supset ((A \supset B) \supset (A \supset C))$

In fact axiom I is redundant since one can show that it is derivable from the other two axioms.

4.5 Sequent Calculus

4.5.1 Introduction

The Sequent Calculus (SC) was introduced by Gentzen. It is as powerful as Axiomatic Systems and Natural Deduction, i.e. $\vdash_{ND} \equiv \vdash_{H} \equiv \vdash_{SC}$.

In SC rules are rather applied on structures called *sequents* than on formulas. A sequent consists of zero or more hypothesis and one conclusion formula with a $\vdash$ in between: $A_1, A_2, ..., A_n \vdash B$. In contrast to that, the $\vdash$ in ND and Axiomatic Systems is not part of the object language (the language in which the formulas are written) but rather of the meta language (e.g. mathematical English) meaning "B is derivable from A_i" or "A_i proofs B".

4.5.2 Rules

Introduction and derivation

Rules can only introduce connectives, not eliminate them like in ND. There exist 4 different kinds of rules:

- axioms
- cut

- structural rules
- logical rules

Axioms are of the form $\gamma \vdash A$ with $A \in \gamma$. It has the informal meaning that a set of formulas is sufficiant to proof a formula which is contained in this set.

The cut rule

$$\frac{\Gamma, A \vdash B \qquad \Sigma \vdash A}{\Gamma, \Sigma \vdash B}\, cut$$

is theoretically important as we will see. However, it is practically of little interest since it has a major drawback. It does not fulfill the subformula property, meaning that the premises are subformulas of the conclusion. This makes the rule computationally intractable since the procedure has to guess the A, which has in general nothing to do with Γ, Σ and B.

The **Cut-Elimination Theorem** states that applications of the cut rule can always be eliminated!

Structural rules are necessary as a substitution of the missing connective elimination rules:

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\, weakening - l \qquad \frac{\Gamma \vdash}{\Gamma \vdash B}\, weakening - r$$

$$\frac{\Gamma, B, A \vdash C}{\Gamma, A, B \vdash C}\, exchange \qquad \frac{\Gamma, A, A, \Delta \vdash C}{\Gamma, A, \Delta \vdash C}\, contraction$$

The majority of the rules are logical rules which can be formulated using the intuitive properties of the inference relation (see 4.2).

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C} \wedge - l \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge - r$$

$$\frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C} \vee - l \qquad \frac{\Gamma \vdash A_1, A_2}{\Gamma \vdash A_1 \vee A_2} \vee - r$$

$$\frac{\Gamma \vdash A \qquad \Gamma, B \vdash C}{\Gamma, A \supset B \vdash C} \supset - l \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B} \supset - r$$

$$\frac{\Gamma \vdash A}{\Gamma, \neg A \vdash} \neg - l \qquad \frac{\Gamma, A \vdash}{\Gamma \vdash \neg A} \neg - r$$

This set of rules is sound and complete for intuitionistic logic and it is sound (but not complete!) for PL0. To make it complete for PL0, *reductio ad absurdum* has to be added to the set of rules.

However, just adding

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

as an additional rule is a bad idea since it violates the subformula property and it makes the cut-rule to fail in some situations. A better approch is to extend all the upper rules with an additional Δ (which is a set of rules) on the right side of the $\vdash$ (Note that the original formulas from above do not allow arbitrary formulas on the right side but just one specific form!).

Rules for PL0

This leads to the following set of rules which is sound and complete for PL0. However, since it incorporates RAA, it is not sound anymore for intuitionistic logic.

$$\frac{\Gamma, A \vdash C, \Delta}{\Gamma, A \wedge B \vdash C, \Delta} \wedge -l \qquad \frac{\Gamma \vdash A, \Delta \qquad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta} \wedge -r$$

$$\frac{\Gamma, A \vdash C, \Delta \qquad \Gamma, B \vdash C, \Delta}{\Gamma, A \vee B \vdash C, \Delta} \vee -l \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \vee B \Delta} \vee -r$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma, B \vdash \Delta}{\Gamma, A \supset B \vdash \Delta} \supset -l \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \supset B, \Delta} \supset -r$$

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

One can further enhance the above set of rules by modifiing $\wedge - l$ and $\vee - r$.

$$\frac{\Gamma, A, B \vdash C, \Delta}{\Gamma, A \wedge B \vdash C, \Delta} \wedge -l\prime \qquad \frac{\Gamma \vdash A_1, A_2, \Delta}{\Gamma \vdash A_1 \vee A_2 \Delta} \vee -r\prime$$

In fact this makes the structural rules redundant since $\wedge - l$ can be derived from $\wedge - l\prime$ (and the structural rules) and vice versa (the same holds for $\vee - r$ and $\vee - r\prime$).

Theorem. *$\wedge - l$ can be derived from $\wedge - l\prime$ and vice versa, i.e. $\wedge - l \iff \wedge - l\prime$.*

Proof. We show in both directions that one rule is sufficiant to give a proof for the other one.
$\Longrightarrow$

$$\dfrac{\dfrac{\dfrac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B, B \vdash \Delta} \wedge -l}{\Gamma, A \wedge B, A \wedge B \vdash \Delta} \wedge -l}{\Gamma, A \wedge B \vdash \Delta} \mathit{contraction}$$

$\Longleftarrow$

$$\dfrac{\dfrac{\dfrac{\Gamma, A \vdash \Delta}{\Gamma, A, B \vdash \Delta} \mathit{weakening} - l}{\Gamma, A \wedge B \vdash \Delta} \wedge - l\prime$$

□

By using the enhanced rules instead of the native ones we can get rid of the structural rules.

4.5.3 Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathit{axiom}}{A, B \vdash A} \quad \dfrac{\mathit{axiom}}{A, B \vdash B}}{A, B \vdash A \wedge B} \wedge - r\prime}{A \vdash B \supset (A \wedge B)} \supset -r}{\vdash A \supset (B \supset (A \wedge B))} \supset -r$$

4.6 Axiomatic Systems, Natural Deduction and Sequent Calculus revisited

Axiomatic Systems, Natural Deduction and the Sequent Calculus are in fact all equivalent. All formulas which can be proved in one of these calculi have also a proof in the other ones.

Theorem. $\Gamma \vdash_H A \iff \Gamma \vdash_{ND} A \iff \Gamma \vdash_{SC} A$

Proof. The equivalence is shown in two steps. We first show that (i) $\Gamma \vdash_H A \iff \Gamma \vdash_{ND}$ A and then that (ii) $\Gamma \vdash_H A \iff \Gamma \vdash_{SC} A$. It is then a direct consequence that $\Gamma \vdash_{ND} A \iff \Gamma \vdash_{SC} A$.

(i) The equivalence between $\vdash_H$ and $\vdash_{ND}$ is shown in both directions.

$\Longrightarrow$ This direction is trivial. One can show that each proof in the Hilbert System has a proof in Natural Deduction by proving each axiom and the modus ponens in ND.

$\Longleftarrow$ The proof is done by induction on the length of the proof.
$\|l\| = 1$: The only possible case where the proof consists of just one step is when $A \in \Gamma$ and $\Gamma \vdash_{ND} A$. Then A itself is a proof in the Hilbert System by definition.
$\|l\| = n \mapsto \|l\| = (n+1)$: The proof has to be done for each rule of ND, here it is shown only for $\wedge - e$. Assume that we have a proof

$$\frac{\begin{array}{c}\Gamma \\ \cdots \\ A \wedge B\end{array}}{A} \wedge - e$$

in ND. Then by induction hypothesis we have also a proof for $\Gamma \vdash_H A \wedge B$ in the Hilbert system. What remains to show is that $\vdash_H A \wedge B \supset A$. This can be done, but is left out here to make the proof sketch more readable.

(ii) We now show the equivalence between $\vdash_H$ and $\vdash_{SC}$ in both directions.

$\Longrightarrow$ This direction is again trivial. One can show that each proof in the Hilbert system has a proof in Sequent Calculus by proving each axiom and the modus ponens in SC. The proof of the latter one requires the use of the cut rule!

$\Longleftarrow$ The proof is also done by induction on the length of the proof.
$\|l\| = 1$: The only possible case where the proof consists of just one step is when $A \in \Gamma$ and $\Gamma \vdash_{SC} A$. Then A itself is a proof in the Hilbert system by definition.
$\|l\| = n \mapsto \|l\| = (n+1)$: The proof has to be done for each rule of SC, here it is shown only for $\supset -l$. Assume that we have a proof

$$\frac{\Gamma \vdash_{SC} A, \Delta \qquad \Gamma, B \vdash_{SC} \Delta}{\Gamma, A \supset B \vdash_{SC} \Delta} \supset -l$$

in SC. Then by induction hypothesis we have also a proof for $\bigwedge \Gamma \vdash_H A \vee \bigvee \Delta$ and $\bigwedge \Gamma \wedge B \vdash_H \bigvee \Delta$ in the Hilbert System. Note that the set of formulas Γ and Δ needs to be written as $\bigwedge \Gamma$ and $\bigvee \Delta$ since the Hilbert System operates only on formulas rather than sequents. What remains to show is that $\vdash_H ((\bigwedge \Gamma \supset (A \vee \Delta)) \wedge (\bigwedge \Gamma \wedge B \supset A)) \supset ((\bigwedge \Gamma \wedge (A \supset B)) \supset \bigvee \Delta)$. This can be done but is again left out here for reasons of clearance.
□

4.7 Soundness and Completeness

So far we have assumed intuitive inference rules (see 4.2), have written them in the syntax of ND and SC, and we have reduced them to axioms and modus

ponens to form an Axiomatic System.

However, we still work with *intuitive* inference rules. What remains to show is that they do in fact correspond to their semantic pendants, i.e. $\vdash_{SC} \equiv \models$. Due to the fact that we have already shown that $\vdash_H \equiv \vdash_{ND} \equiv \vdash_{SC}$, we can conclude that $\vdash_H \equiv \models$ and $\vdash_{ND} \equiv \models$.

The equivalence $\vdash_{SC} \equiv \models$ is shown in both directions. We first show the soundness ($\Longrightarrow$) of Sequent Calculus.

Theorem. *$\vdash_{SC}$ is sound (correct), i.e. if a formula is deivable, then it is semantically valid.*

Proof. We have to show the soundness of each logical rule. Here this is done for $\vee - r$.

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

Assume that the premises are semantically true. We show that then also the consequence is semantically true. There are different possibilities which can make the premises true:
(i) $e(\gamma) = 0$ for a $\gamma \in \Gamma$: Then also the conclusion is true because of $e(\gamma) = 0$ for a $\gamma \in \Gamma$
(ii) $e(\delta) = 1$ for a $\delta \in \Delta$: Then also the conclusion is true because of $e(\delta) = 1$ for a $\delta \in \Delta$
(iii) $e(A) = 1$: Then $e(A \vee B) = 1$ which makes the consequence true

If similar proofs are done for all rules, then it follows that whenever a formula can be derived, it must be valid, since all rules are correct and the axioms are trivially correct because in $\Gamma, A \vdash_{SC} \Delta, A$ whenever the premise is true it especially makes $e(A) = 1$ and thus the consequence true. □

Now we show the completeness ($\Longleftarrow$) of Sequent Calculus.

Theorem. *$\vdash_{SC}$ is complete, i.e. if a formula is valid, then it can be syntactically derived.*

Proof. Assume that the proof is made bottom-up. Observe the following property given rule set (see 4.5.2) with $\wedge - l$ and $\vee - r$ being replaced by $\wedge - l\prime$ and $\vee - r\prime$. Not only the premises imply the conclusion (semantically) but also the other way round. Therefore, if we do the proof bottom-up, the validity is preserved in each application of an rule. Thus if a formula is actually valid, and in each application of a rule one connective is stripped off, we must end with nothing but axioms. □

As an interesting property of the Sequent Calculus consider the following. If a trial proof does not end only with axioms, then the anti-axioms describe those formulas which can not be proven. This means that if all anti-axioms are $\wedge-$connected it describes exactly the necessary knowledge base to conclude the original formula. By reformulating each anti-axiom $A_1, ..., A_n \vdash_{SC} B_1, ..., B_m$ into $\neg A_1 \vee \neg A_2 \vee ... \vee \neg A_n \vee B_1 \vee B_2 \vee ... \vee B_m$ and $\wedge-$connecting all those formulas, we have a new algorithm to construct the CNF (conjunctive normal form) of a PL0 formula. (Note that if the derivation in SC succeeds then the formula is a tautology, thus we can simply use $A \vee \neg A$ (or any other valid formula) as it's CNF.)

4.8 Cut-Elimination Theorem

The Cut-Elimination Theorem states that Sequent Calculus for PL0 without the cut rule is still sound and complete.

Recall the cut rule:

$$\frac{\Gamma, A \vdash B \qquad \Sigma \vdash A}{\Gamma, \Sigma \vdash B}\, cut$$

Theorem. *Each proof in Sequent Calculus for classical propositional logic which contains applications of cut can be translated into a proof without cut.*

Proof. The proof is done inductively on the depth of the cut application in the proof tree. It will be shown that the cut application can be shifted upwards and can be eliminated when it's premises are atoms.

Suppose

$$\dfrac{\dfrac{\begin{matrix}d_r\\ \cdots\\ \Gamma, A \vdash B\end{matrix} \qquad \dfrac{\begin{matrix}d_l\\ \cdots\\ \Sigma \vdash A\end{matrix}}{\cdots}}{\Gamma, \Sigma \vdash B}\, cut}{\cdots}$$

We distinct three cases.

(i) At least one of d_r and d_l is an axiom. Then the cut rule can be eliminated in the following way.

If the application is of the pattern

$$\dfrac{\Gamma, A \vdash A \qquad \dfrac{\begin{matrix}d_r\\ \cdots\\ \Sigma \vdash A\end{matrix}}{\cdots}}{\Gamma, \Sigma \vdash A}\, cut$$

it can be replaced by

$$\dfrac{\begin{array}{c}d_r\\ \vdots\\ \Gamma \vdash A\end{array}}{\Gamma, \Sigma \vdash A}\, weakening$$

The second possibility where one of the premises is an axiom is

$$\dfrac{\begin{array}{c}d_r\\ \vdots\\ \Gamma, A \vdash B\end{array} \qquad \Sigma, A \vdash A}{\Gamma, \Sigma, A \vdash B}\, cut$$

which can be replaced by

$$\dfrac{\begin{array}{c}d_r\\ \vdots\\ \Gamma, A \vdash B\end{array}}{\Gamma, \Sigma, A \vdash B}\, weakening$$

(ii) If none of the premises is an axiom, but both premises which were used during the cut application are principals (Note: a formula is a principal if it was just introduced by an inference rule; the formulas used for the introduction are called auxiliary formulas), we can not directly eliminate cut. But we can convert the proof in such a way that the length of the proof *before* cut application decreases. Thus the cut application can be moved upwards until we can apply case 1.

The proof needs to be done for each connective in Sequent Calculus. We have to consider how we can move the cut application upwards if the principals are the result of a specific rule application.

Consider for instance the $\wedge - rule$. We have

$$\dfrac{\dfrac{\begin{array}{c}d_1\\ \vdots\\ \Gamma, A \vdash C\end{array}}{\Gamma, A \wedge B \vdash C}\wedge - l \qquad \dfrac{\begin{array}{c}d_2\\ \vdots\\ \Sigma \vdash A\end{array} \qquad \begin{array}{c}d_3\\ \vdots\\ \Sigma \vdash B\end{array}}{\Sigma \vdash A \wedge B}\wedge - r}{\Gamma, \Sigma \vdash C}\, cut$$

This proof can simplified to

$$\dfrac{\begin{array}{c}d_1\\ \vdots\\ \Gamma, A \vdash C\end{array} \qquad \begin{array}{c}d_2\\ \vdots\\ \Sigma \vdash A\end{array}}{\Gamma, \Sigma \vdash C}\, cut$$

Note that the two proofs before the cut application became shorter! In this way we can transform the proof for each connective.

(iii) The last case considers proofs where none of the premises of cut is an axiom, and the two formulas which are treated by cut were not introduced in the previous step. In this case we consider just the last rule application in one of the two proof branches.

For instance, if in one of the two branches the last rule application was $\supset -l$ we have

$$\dfrac{\dfrac{\begin{array}{c}d_1\\ \cdots\\ \Gamma, C \vdash D\end{array}}{\Gamma, \Sigma, A \supset B \vdash D}\, cut \qquad \dfrac{\begin{array}{c}d_2\\ \cdots\\ \Sigma \vdash A, C\end{array} \qquad \begin{array}{c}d_3\\ \cdots\\ \Sigma, B \vdash C\end{array}}{\Sigma, A \supset B \vdash C}\, \supset -l}$$

This can be converted to

$$\dfrac{\dfrac{\begin{array}{c}d_1\\ \cdots\\ \Gamma, C \vdash D\end{array} \qquad \begin{array}{c}d_3\\ \cdots\\ \Sigma, B \vdash C\end{array}}{\Gamma, \Sigma, B \vdash D}\, cut \qquad \dfrac{\begin{array}{c}d_2\\ \cdots\\ \Sigma \vdash A\end{array}}{\Gamma, \Sigma \vdash A}\, weakening - l}{\Gamma, \Sigma, A \supset B \vdash D}\, \supset -l$$

Note that the two proof branches in the premises of cut became shorter again.

This can be done for the most rules. However, there are still cases which can not be solved in this way. Consider

$$\dfrac{\dfrac{\begin{array}{c}d_1\\ \cdots\\ \Gamma, A, A \vdash B\end{array}}{\Gamma, A \vdash B}\, \text{contraction} \qquad \begin{array}{c}d_2\\ \cdots\\ \Sigma \vdash A\end{array}}{\Gamma, \Sigma \vdash B}\, cut$$

This proof can be converted to

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}d_1\\ \cdots\\ \Gamma, A, A \vdash B\end{array} \qquad \begin{array}{c}d_2\\ \cdots\\ \Sigma \vdash A\end{array}}{\Gamma, \Sigma, A \vdash B}\, cut \qquad \begin{array}{c}d_2\\ \cdots\\ \Sigma \vdash A\end{array}}{\Gamma, \Sigma, \Sigma \vdash B}\, cut}{\Gamma, \Sigma \vdash B}\, \text{contraction}$$

But this does not help eliminating the cut rule, since the left branch of the proof before the second cut application did not became shorter. Gentzen's solution for this problem is to eliminate a generalization of the cut rule (mix) rather than the cut rule itself.

$$\frac{\Gamma, A, ..., A \vdash B \qquad \Sigma \vdash A}{\Gamma, \Sigma \vdash B}\, mix$$

Note that this generalization is directly derivable from the basic cut rule by iterative application of cut. □

4.9 Linear Logic

Linear Logic was introduced by Girard for proof theoretical reasons. In this logic intuitively $A \equiv A \wedge A$ does not hold because a proposition stands for a resource.

Imagine for instance that E means *1 Euro*, M means *A pack of Marlboro* and C means *A pack of Camel*. Then if a pack of cigarettes costs 1 Euro, $E \supset M$ holds as well as $E \supset C$. But of course $E \supset (M \wedge C)$ should not hold. Also $E \wedge E$ (2 Euros) is different from just E.

Chapter 5

Classical First Order Logic

Classical First Order Logic (FOL, Predicate Logic, PL1) is an extension of PL0. It allows the propositions to have an inner structure, i.e. atoms are not just propositional variables but they depend on terms. FOL further allows the quantification over variables.

5.1 Syntax

Generally we have an alphabet of all constant symbols $\mathcal{C}$, function symbols $\mathcal{F}$, predicate symbols $\mathcal{P}$, variables $\mathcal{V}$, quantifiers ($\forall$, $\exists$), paranthesis and other meta symbols $(,),$, and propositional connectives $\neg, \wedge, \vee$, etc.. Constant symbols can be seen as function symbols of arity 0.

Then the syntax of FOL is defined in two steps.

A term is defined as follows:

- Each variable $v \in \mathcal{V}$ is a term, i.e. $v \in \mathcal{T}$
- Each constant symbol $c \in \mathcal{C}$ is a term, i.e. $c \in \mathcal{T}$
- If $t_1, t_2, ..., t_n \in \mathcal{T}$ and f is a function symbol of arity n, then $f(t_1, t_2, ..., t_n) \in \mathcal{T}$

Terms are solely given by the above definitions.

On top of the term definition we can define atomic formulas as follows:

- If $p \in \mathcal{P}$ is a predicate symbol of arity n and $t_1, t_2, ..., t_n \in \mathcal{T}$ are terms, then $p(t_1, t_2, ..., t_n) \in \mathcal{F}$ is an atomic formula

Composed formulas are then given by:

- If $a \in \mathcal{F}$ is a formula, then $\neg a \in \mathcal{F}$ is a formula
- If $a, b \in \mathcal{F}$ are formulas, then $a \wedge b \in \mathcal{F}$ is a formula
- If $a, b \in \mathcal{F}$ are formulas, then $a \vee b \in \mathcal{F}$ is a formula
- ... (for all propositional connectives) ...
- If $a, b \in \mathcal{F}$ are formulas, then $a \supset b \in \mathcal{F}$ is a formula

In contrast to propositional logic, FOL also allows quantification over variables:

- If $f \in \mathcal{F}$ is a formula and $x \in \mathcal{V}$ is a variable, then $\forall x.f$ is a formula
- If $f \in \mathcal{F}$ is a formula and $x \in \mathcal{V}$ is a variable, then $\exists x.f$ is a formula

Formulas are solely given by the above definitions.

5.2 Semantics

The semantics of FOL is defined recursively. Generally an interpretation consists of a structure $\mathcal{S} = (\mathcal{U}, \mathcal{I})$ and an environment $\mathcal{E}$.

$\mathcal{U}$ is a non-empty set of arbitrary elements, called Herbrand-Universe. $\mathcal{I}$ is an interpretation of the constant-, function- and predicate symbols:

- $\mathcal{I}(c) = c'$ for a constant symbol $c \in \mathcal{C}$, $c' \in \mathcal{U}$ is the constant for symbol c
- $\mathcal{I}(f) = f'$ for a function symbol $f \in \mathcal{F}$, $f' : \mathcal{U}^n \mapsto \mathcal{U}$ is the n-ary function for symbol f
- $\mathcal{I}(p) = p'$ for a predicate symbol $p \in \mathcal{P}$, $p' \subseteq \mathcal{U}^n$ is an n-ary relation over $\mathcal{U}$ consisting of all n-tuples which make p' true

$\mathcal{E}$ is the environment which assigns a value to each variable, i.e. $\mathcal{E} : \mathcal{V} \mapsto \mathcal{U}$.

With these definitions we are now able to give a recursive definition of the meaning function $\mathcal{M}_{\mathcal{S}}(\mathcal{E}, f)$ over an arbitrary structure $\mathcal{S}$ and environment $\mathcal{E}$ and a given formula $f \in \mathcal{F}$.

First we define the evaluation of terms.

- $\mathcal{M}_{\mathcal{S}}(\mathcal{E}, c) = \mathcal{I}(c)$ for a constant symbol $c \in \mathcal{C}$

- $\mathcal{M}_\mathcal{S}(\mathcal{E}, f(t_1, t_2, ..., t_n)) = \mathcal{I}(f)(\mathcal{M}_\mathcal{S}(\mathcal{E}, t_1), \mathcal{M}_\mathcal{S}(\mathcal{E}, t_2), ..., \mathcal{M}_\mathcal{S}(\mathcal{E}, t_n)) = f'((\mathcal{M}_\mathcal{S}(\mathcal{E}, t_1), \mathcal{M}_\mathcal{S}(\mathcal{E}, t_2), ..., \mathcal{M}_\mathcal{S}(\mathcal{E}, t_n))$ for a function symbol $f \in \mathcal{F}$ of arity n and terms $t_1, t_2, ..., t_n \in \mathcal{T}$
- $\mathcal{M}_\mathcal{S}(\mathcal{E}, x) = \mathcal{E}(x)$ for a variable $x \in \mathcal{V}$

The evaluation of formulas is done by the application of the following rules:

- $\mathcal{M}_\mathcal{S}(\mathcal{E}, p(t_1, t_2, ..., t_n)) = true$ iff $(\mathcal{M}_\mathcal{S}(t_1), \mathcal{M}_\mathcal{S}(t_2), ..., \mathcal{M}_\mathcal{S}(t_n)) \in \mathcal{M}_\mathcal{S}(\mathcal{E}, p)$ for a predicate symbol $p \in \mathcal{P}$
- $\mathcal{M}_\mathcal{S}(\mathcal{E}, a \wedge b) = true$ iff $\mathcal{M}_\mathcal{S}(\mathcal{E}, a) = true$ and $\mathcal{M}_\mathcal{S}(\mathcal{E}, b) = true$
- $\mathcal{M}_\mathcal{S}(\mathcal{E}, a \vee b) = true$ iff $\mathcal{M}_\mathcal{S}(\mathcal{E}, a) = true$ or $\mathcal{M}_\mathcal{S}(\mathcal{E}, b) = true$
- ... (for all propositional connectives) ...
- $\mathcal{M}_\mathcal{S}(\mathcal{E}, \forall x.f) = true$ iff $\mathcal{M}_\mathcal{S}(\mathcal{E}', f)$ for all $\mathcal{E}'$ with $\mathcal{E}'(y) = \mathcal{E}(y)$ for $y \neq x$
- $\mathcal{M}_\mathcal{S}(\mathcal{E}, \exists x.f) = true$ iff $\mathcal{M}_\mathcal{S}(\mathcal{E}', f)$ for at least one $\mathcal{E}'$ with $\mathcal{E}'(y) = \mathcal{E}(y)$ for $y \neq x$

5.3 Cook's Theorem revisited

Remember Cook's Theorem from section 1.3.2. It is a reduction of arbitrary Turing Machines with at most t steps of computation to the propositional satisfiability problem (SAT). It implies that SAT is NP-hard, since Turing Machines (with restricted calculation time) are also NP-hard.

The theorem can be extended to show that PL1 is undecidable. In fact PL1 is semi-decidable, which means that one can write an algorithm which will always stop and answer *true* in the case a formula is satisfiable. However, in the contrary case the algorithm may does not terminate.

The only difference to Cook's Theorem is that we can now drop the restriction of bounded time since we have quantors.

We define the following predicates:

- $T(i, j, k) = true$ iff tape cell i contains symbol j at step k of computation
- $H(i, k) = true$ iff the head is over cell i at step k of computation
- $Q(q, k) = true$ iff the machine is in state q at step k of computation

Further we define the following formulas:

$C_1 = \bigwedge T(i,j,k) \supset \neg T(i,j',k) \forall j' \neq j$
$C_2 = \bigwedge H(i,k) \supset \neg H(i',k) \forall i' \neq i$
$C_3 = \bigwedge Q(q,k) \supset \neg Q(q',k) \forall q' \neq q$
$C_4 = \bigwedge T(i,j,k) \supset (T(i,j,k) \vee H(i,k))$

This enables us to set up the formula $\exists k.C_1 \wedge C_2 \wedge C_3 \wedge C_4$, which is satisfiable iff the encoded Turing Machine will finally reach an accept state. Since general Turing Machines are undecidable, FOL can not be decidable.

5.4 Prenex Normal Form

A formula in PL1 is said to be in prenex normal form if it is written as a string of quantifiers followed by a quantifier-free part, i.e.
$Q_1 x_1 Q_2 x_2 ... Q_n x_n P(x_1, x_2, ..., x_2)$
where $Q_i \in \{\forall, \exists\}$ and variables x_i (which are pairwise disjunct). $P(x_1, x_2, ..., x_n)$ is a quantifier-free formula with (free) variables x_i.

Example: $\forall x \exists y \forall z (x \wedge y \supset z)$.

Each formula in PL1 has an equivalent formula in prenex normal form.

5.5 Undecidability of PL1

PL1 is undecidable, i.e. it is impossible to find a procedure which tests if an arbitrary formula is valid or not. However PL1 is semidecidable, meaning that an algorithm can be given which gives at least in the positive case always an answer.

The idea of the proof is well-known as *diagonal method*, which was introduced by Georg Cantor and can for instance be used to show the infinity of irrational numbers.

Another possibility is the reduction of another undecidable problem (e.g. the Halting problem) to PL1, which shows that PL1 can not be of lower complexity.

5.6 Sequent Calculus for PL1

The Sequent Calculus for PL1 is obtained by adding additional rules for the quantifiers to the Sequent Calculus for PL0 (see 4.5). We have 2 rules for

each quantifier:

$$\frac{\Gamma, A(t) \vdash \Delta}{\Gamma, \forall x A(x) \vdash \Delta} \forall - l \qquad \frac{\Gamma, A(y) \vdash \Delta}{\Gamma, \exists x A(x) \vdash \Delta} \exists - l$$

$$\frac{\Gamma \vdash A(y), \Delta}{\Gamma \vdash \forall x A(x), \Delta} \forall - r \qquad \frac{\Gamma \vdash A(t), \Delta}{\Gamma \vdash \exists x A(x), \Delta} \exists - r$$

where y is is variable which does not occur (free) yet in the sequent, i.e. $y \notin \Gamma \cup \Delta$, a so called *eigenvariable* and t is an arbitrary term. Informally the rules can be explained as follows.

- If $A(t)$ occus on the left side, we can replace it by $\forall x A(x)$ in the next step. Note that the models of $\forall x A(x)$ are a subset of the models of $A(t)$. Thus the left side has fewer models after the rule application than before. This makes the implication stay valid.
- If $A(y)$ occus on the left side and y *does not occur somewhere else*, we can replace it by $\exists x A(x)$ in the next step. The premise is satisfied if a formula in Γ is false, the conclusion is true or $A(y)$ is false. In the first two cases, also the conclusion is satisfied. In the third case, remember that y is a formula which does not occur somewhere else in the sequent, thus it holds for arbitrary formulas. Therefore, $\exists x A(x)$ is false, which makes the sequent true.
- If $A(t)$ occurs on the right side, then we can replace it in the next step by $\exists x A(x)$ since we have found a witness for that, namely t.
- If $A(y)$ occus on the right side and y *does not occur somewhere else* we can replace it by $\forall x A(x)$ in the next step. Because of the fact that the sequent does not refer to y somewhere else, it does not make any presumptions about it and thus it holds for arbitrary (=all) y.

Recall the Sequent Calculus for PL0. We did enhance the basic set of rules with two additional rules $\wedge - l'$ and $\vee - r'$ which helped us to drop the structural rules (see section 4.5.2). One can do a similar extension for the quantifiers $\exists$ and $\forall$.

$$\frac{\Gamma, A(t), \forall x A(x) \vdash \Delta}{\Gamma, \forall x A(x) \Delta B} \forall - l'$$

$$\frac{\Gamma \vdash A(t), \exists x A(x), \Delta}{\Gamma \Delta \exists x A(x), B} \exists - r'$$

The rules state that it is impossible to "include" a further atom into already existing (and appropriate) quantified formulas. This extension will be needed in the completeness proof.

5.6.1 Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A(y) \vdash A(y)}{\vdash \neg A(y), A(y)}\neg - r}{\vdash \exists x \neg A(x), A(y)}\exists - r}{\vdash \exists x \neg A(x), \forall x A(x)}\forall - r}{\neg \exists x \neg A(x) \vdash x A(x)}\neg - l$$

Reading this proof bottom-up we must first when we apply $\forall - r$ use a new y. In the next step, when we apply $\exists - r$ we can chose an arbitrary t. We substitute t by y and can finally proof the formula.

5.6.2 The Cut-Elimination Theorem in PL1

The idea of the proof is like for PL0 again to move the application of the cut rule upwards and reduce the length of the proofs before it.

If the cut rule is applied to sequents where the cut formula is an all-quantified term like $\forall x A(x)$, which is principal in both premises, it can simply be removed. In this case replace

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}_l}{\dots}}{\Sigma, A(t) \vdash \Pi}}{\Sigma, \forall x A(x) \vdash \Pi}\forall - l \quad \dfrac{\dfrac{\dfrac{\mathcal{D}_r}{\dots}}{\Sigma, A(y) \vdash \Delta}}{\Gamma \vdash \forall x A(x), \Delta}\forall - r}{\Sigma, \Gamma \vdash \Pi, \Delta}\,cut$$

by

$$\dfrac{\dfrac{\dfrac{\mathcal{D}_l}{\dots}}{\Sigma, A(t) \vdash \Pi} \quad \dfrac{\dfrac{\mathcal{D}_r}{\dots}}{\Gamma \vdash A(t), \Delta}}{\Sigma, \Gamma \vdash \Pi, \Delta}\,cut$$

5.6.3 Soundness and Completeness

The Sequent Calculus for first order logic is sound and complete.

Theorem. *PL1 SC is sound (correct), i.e.* $\Gamma \vdash A \Longrightarrow \Gamma \models A$

Proof. The proof has to be done separately for each rule. Here this is done exemplary for $\forall - r$. (For the rules without quantifiers, the proof is the same as for PL0 SC.)

Recall the inference rule:

$$\frac{\Gamma \vdash A(y), \Delta}{\Gamma \vdash \forall x A(x), \Delta} \forall - r$$

A is of the form $\forall A_1$. We have to show that each model of the premisse $(\mathcal{S}, \mathcal{E}) \models \Gamma$ is also a model of the conclusion $(\mathcal{S}, \mathcal{E}) \models A$.

Due to the premisse we know that if $(\mathcal{S}, \mathcal{E}) \models \Gamma$ then $(\mathcal{S}, \mathcal{E}) \models A_1[y/x]$, where $A_1[y/x]$ denotes the formula where each free occurrence of x is replaced by y.

Because of the fact that y is an eigenvariable (i.e. $y \notin \Gamma$) we know that $(\mathcal{S}, \mathcal{E}[b/y]) \models \Gamma$ for all $b \in \mathcal{U}$. $\mathcal{E}[b/y]$ denotes the environments where y has a specific value $b \in \mathcal{U}$ and all other variables are assigned to the same value as in $\mathcal{E}$. The premisse states that all of these models are also models of $A[y/x]$. We can conclude that $(\mathcal{S}, \mathcal{E}[b/y]) \models A_1[y/x] \iff (\mathcal{S}, \mathcal{E}[b/x]) \models A_1[b/x]$. Thus $(\mathcal{S}, \mathcal{E}) \models \forall x A$.

□

Theorem. *PL1 SC is complete, i.e.* $\Gamma \models A \Longrightarrow \Gamma \vdash A$

Proof.

Definition. *First we define a concept called positive constituent and negative constituent. A formula A of a sequent $\Gamma \vdash \Delta$ is a positive constituent with respect to a given interpretation $(\mathcal{S}, \mathcal{E})$, if this formula is sufficient in the interpretation to make the whole sequent semantically true, i.e. $A \in \Delta$ and $\mathcal{M}_{\mathcal{S}}(\mathcal{E}, A) = 1$ or $A \in \Gamma$ and $\mathcal{M}_{\mathcal{S}}(\mathcal{E}, A) = 0$. A negative constituent is a formula with the opposite characteristics.*

Theorem. *It can be shown that if we have an interpreatation $(\mathcal{S}, \mathcal{E})$ and a positive constituent A of the conclusion $\Gamma \vdash \Delta$ of a sequent of a rule of the Sequent Calculus, and A is of the form $\neg A_1$, $A_1 \wedge A_2$, $A_1 \vee A_2$, $A_1 \rightarrow A_2$, $\forall x A_1$ (in Δ) or $\exists x A_1$ (in Γ), then in each premisse of the rule one of the auxiliary formulas is a positive constituent. Recall: The formula which is introduced by a logical rule is called principal, the formulas which were used*

for the introduction are the auxiliary formulas.

Example (propositional case due to simplicity):

$$\frac{A \vdash A \supset B \qquad B \vdash A \supset B}{\vdash A \vee B \vdash A \supset B} \vee -l$$

with the interpretation $e(A) = e(B) = 0$. In this case $e(A \vee B) = 0$ and thus is a positive constituent of the sequent $\vdash A \vee B \vdash A \supset B$. The auxiliary formulas in the premises are A repectively B, which are obviously positive constiturents in their sequents (with this interpretation).

It is sufficient to show that if a formula can not be proven (i.e. $\Gamma \nvdash A$) it does not hold that $\Gamma \models A$.

Now we start with the actual proof.

Imagine an algorithm which gets a sequent $\Gamma \vdash \Delta$ as parameter and tries to apply backward a logical or quantifier rule to this sequent. In odd steps, it applies a rule to the first compound formula of Γ, in even steps it applies one to the first compound formula of Δ. It stops on axioms or anti-axioms.

This algorithm will (a) either find a proof for the formula (making the theorem true in this case) or (b) fails.

If it fails, there are two possible reasons. (i) At least one branch ends with an anti-axiom. In this case the a counter model for $\Gamma \models \Delta$ can be directly extracted from the anti-axiom. (ii) A branch in infinite. In this case we get a countably infinite sequence of sequents:

$$\frac{\frac{\frac{\frac{\ldots}{\Gamma_3 \vdash \Delta_3}}{\Gamma_2 \vdash \Delta_2}}{\Gamma_1 \vdash \Delta_1}}{\Gamma_0 \vdash \Delta_0}$$

We define two sets $\mathcal{M} = \{af(\Gamma_0), af(\Gamma_1), af(\Gamma_2), af(\Gamma_3), ...\}$ and $\mathcal{N} = \{af(\Delta_0), af(\Delta_1), af(\Delta_2), af(\Delta_3), ...\}$ where $af(X)$ returns the atomic functions in set X.

Note that all Γ_i and Δ_j are pairwise disjoint, otherwise we would have encountered an axiom due to the fact that the algorithm applies rules alternately to formulas on the left and the right side.

With this definitions we can create our counter model for $\Gamma \models \Delta$: $\mathcal{M}_{\mathcal{S}}(\mathcal{E}, P(t_1, t_2, ..., t_n) = true$ iff $P(t_1, t_2, ..., t_n) \in \mathcal{M}$. This makes *all* formulas in Γ_i and *none* in Δ_i true. Thus, by our construction, *all* elementary constituents are negative.

However due to the theorem introduced at the beginning of the proof, if a sequent contains a positive constituent (which is of course the case if a formula is valid) of order n, there must be sequents in the branch of the proof tree of order $n-1$, $n-2$ and so on. By n repetitions of this theorem we get that there must also be atomic positive constituents, which is a contradiction to our construction. Therefore $\Gamma \models \Delta$ can not be valid if the proof fails, i.e. PL1 Sequent Calculus is complete.

□

Bibliography

[1] Prof. Agata Ciabattoni - TU Wien
Lecture Slides - Winter term 2008/09

[2] T. Krennwallner *Course's notes*

[3] Jean H. Gallier
Logic For Computer Science - Foundations of Automatic Theorem Proving, 2003

[4] Haskell B. Curry
Foundations Of Mathematical Logic

[5] English Wikipedia (22.01.2009)