

CDM

## Propositional Reasoning

Klaus Sutner  
Carnegie Mellon University  
www.cs.cmu.edu/~sutner

PropLogic

1

### Battleplan

- Propositional Logic
- Valuations and Semantics
- Tautologies and Satisfiability
- Hilbert Style System
- Soundness and Completeness

PropLogic

2

## Propositional Statements

PropLogic

3

### Example: Fields

If we try to describe a field (in the algebraic sense) such as the rationals or the reals we have to cope with the fact that 0 does not have an inverse. So we need a law like

$$x \neq 0 \rightarrow i(x) \cdot x = 1$$

where  $i$  describes the inverse function (which is undefined on 0).

So we are using equations, inequalities, terms, variables and logical implication to express some basic property of a field.

Let's avoid having to deal with quite so many entities all at once and instead focus on just the logical structure:

$$A \rightarrow B$$

PropLogic

4

### Example: Lists

Likewise, if we want to describe formally the properties of lists we may come up with assertions such as

- $\text{head}(\text{prep}(L, a)) = a$
- $\text{tail}(\text{prep}(L, a)) = L$
- $\text{not isempty}(\text{prep}(L, a))$
- $\text{isempty}(L)$  or for some  $a$  and some  $K: L = \text{prep}(K, a)$
- $\text{prep}(L, a) = \text{prep}(K, b)$  implies  $K = L$  and  $a = b$
- If  $P(\emptyset)$  holds and if for all  $L$  and  $a$ ,  $P(L)$  implies  $P(\text{prep}(L, a))$ , then  $P(K)$  holds for an arbitrary list  $K$ .

For the time being we ignore most of this and think of these assertions as being of the form

$$A \quad \neg A \quad A \vee B \quad A \rightarrow B \quad A \wedge B \rightarrow C$$

PropLogic

5

### Propositional Reasoning

More precisely, we will focus on the part having to do with logical connectives such as "and", "or", "implies" and "not".

We will discuss a simple framework that allows us to represent these connectives in a very formal manner.

The next step is to build a deduction system that captures our intuitive way of reasoning. For example, everyone would agree that if we have somehow established an assertion " $A$  and  $B$ " then we have in particular also established  $B$ .

Or if we manage to establish both " $A$ " and " $A$  implies  $B$ " then it is safe to conclude that we also know " $B$ ".

On the other hand, if we only know " $A$ " so far it would be an error to conclude " $A$  and  $B$ ".

And so on and so forth.

## Why Bother?

Isn't this all blindingly obvious?

Propositional reasoning on small, natural examples is indeed more or less obvious – simply because we are trying to capture precisely the fundamental laws of thought that everyone understands and agrees upon.

So, for simple cases of propositional reasoning the human mind is perfectly adequate, but when we have to deal with huge assertions involving thousands of connectives efficient algorithms are necessary. Many practical problems such as checking a train schedule for correctness naturally produces huge propositional expressions.

Another important issue is that we are trying to design algorithms that are capable of propositional reasoning. To this end we need rather fastidious definitions and a very careful description of the deduction system, no appeals to intuition are admissible.

Lastly, when we further extend our system to full predicate logic things become much, much more complicated – you can think of propositional logic as an excellent (and directly useful) warm-up exercise.

## Atomic Propositions

We start with atomic assertions, that are either true or false, but cannot be broken apart (at least not in the current framework).

- “stack  $S$  is empty”
- “18 is a prime number”
- “the GCD of 100 and 35 is 5”
- “polynomial  $p$  has no integer roots”
- “there are more reals than rationals”
- “the algorithm terminates on all inputs”

Then we arrange these atomic assertions into more complicated, compound ones using logical connectives.

## Logical Connectives

Complicated assertions can be built up from atomic assertions by *logical connectives* such as “not”, “and”, “or”, and “implies”.

### Example 1.

- “ $p$  and  $q$  implies  $p$ ”
- “ $p$  implies  $q$ , implies  $q$ ”
- “ $p$  implies  $q$ , implies not  $p$  implies not  $q$ ”
- “ $p$  or not  $p$ ”

We want to understand how these connectives work.

In particular, we want to be able to determine the truth (or falsehood) of a compound statement from its atomic pieces and the connectives.

## Propositional Formulae

We fix a language for propositional formulae, much like a programming language.

$\perp, \top$	constants false, true
$p, q, r, \dots$	propositional variables
$\neg$	not
$\wedge$	and, conjunction
$\vee$	or, disjunction
$\rightarrow$	conditional (implies)
$\leftrightarrow$	biconditional (equivalent)

Negation is unary, all the others a binary.

### Example 2.

$$(p \wedge q \rightarrow r) \rightarrow ((\neg p \vee q) \rightarrow (p \rightarrow r))$$

## Propositional Formulae

**Definition.** The set of propositional formulae is defined by:

- Every constant and variable is a formula.
- If  $\varphi$  and  $\psi$  are formulae, so are
  - $\neg\varphi$ ,
  - $\varphi \wedge \psi$ ,
  - $\varphi \vee \psi$ ,
  - $\varphi \rightarrow \psi$ ,
  - $\varphi \leftrightarrow \psi$ .

Strictly speaking also need parentheses, but we will ignore technical details (details that would be relevant for implementations). As usual, for legibility drop unnecessary parens assuming precedence:

$$\neg > \wedge > \vee > \rightarrow > \leftrightarrow$$

## Examples: Some Implications

$$\begin{aligned} &\perp \rightarrow p \\ &(p \rightarrow \perp) \rightarrow \neg p \\ &p \rightarrow (q \rightarrow p) \\ &\neg(p \wedge q) \rightarrow (\neg p \vee \neg q) \\ &(p \wedge (p \rightarrow q)) \rightarrow q \\ &(p \wedge q \rightarrow r) \rightarrow ((\neg p \vee q) \rightarrow (p \rightarrow r)) \quad (p \wedge q \rightarrow r) \rightarrow ((\neg p \vee q) \rightarrow (p \rightarrow r)) \end{aligned}$$

**Exercise 1.** For each of these, try to determine if they are intuitively valid.

## Semantics

## Beyond Syntax

We have used familiar notation to express propositional formulae, so everybody knows that  $A \wedge B$  is supposed to express a conjunction.

However, strictly speaking all we have so far is syntax, a grammar for a class of expressions – actually, we didn't even bother to write down a precise grammar, but it would not be difficult to do so, and we are not interested in parsers at this point.

However, we do need to be more careful in pinning down the semantics, the meaning of our formulae.

For propositional formulae this is not very difficult, but it's a good warm-up exercise for more complicated systems later (e.g., the semantics of an expression in predicate logic is quite a bit more difficult to describe).

## Semantics

**Definition 1.** A **truth assignment** is a map that associates a truth value with each variable.

Given a formula  $\varphi$  and a truth assignment  $\sigma$  for all the variables, we can determine the truth value of  $\varphi$  under  $\sigma$ , written  $\llbracket \varphi \rrbracket_\sigma$ .

$$\begin{aligned}\llbracket \perp \rrbracket_\sigma &= 0 \\ \llbracket \top \rrbracket_\sigma &= 1 \\ \llbracket \neg \varphi \rrbracket_\sigma &= H_{\text{not}}(\llbracket \varphi \rrbracket_\sigma) \\ \llbracket \varphi \vee \psi \rrbracket_\sigma &= H_{\text{or}}(\llbracket \varphi \rrbracket_\sigma, \llbracket \psi \rrbracket_\sigma) \\ \llbracket \varphi \wedge \psi \rrbracket_\sigma &= H_{\text{and}}(\llbracket \varphi \rrbracket_\sigma, \llbracket \psi \rrbracket_\sigma) \\ \llbracket \varphi \rightarrow \psi \rrbracket_\sigma &= H_{\text{imp}}(\llbracket \varphi \rrbracket_\sigma, \llbracket \psi \rrbracket_\sigma) \\ \llbracket \varphi \leftrightarrow \psi \rrbracket_\sigma &= H_{\text{equ}}(\llbracket \varphi \rrbracket_\sigma, \llbracket \psi \rrbracket_\sigma)\end{aligned}$$

Of course,  $\llbracket p \rrbracket_\sigma = \sigma(p)$  is directly given for propositional variables  $p$ .

## Semantics

**Definition 2.** Let  $\sigma$  be a truth assignment and  $\varphi$  a propositional formula.  $\sigma$  **satisfies or models**  $\varphi$  if  $\varphi$  is true under  $\sigma$ :  $\llbracket \varphi \rrbracket_\sigma = 1$ .

A truth assignment satisfies a set of formulae  $\Phi$  if it satisfies each formula in the set.

In symbols

$$\sigma \models \varphi \quad \text{and} \quad \sigma \models \Phi$$

Incidentally, this notation is lifted from Frege's Begriffsschrift.

Strictly speaking, a truth assignment need only be defined on the variables appearing in the formula and could be defined as a partial map from the collection of all variables but we will not quibble.

## Semantic Consequence

**Definition 3.** Let  $\Phi$  be a set of formulae and  $\varphi$  a propositional formula.  $\varphi$  is a **semantic consequence** of  $\Phi$  if for every truth assignment  $\sigma$  that satisfies  $\Phi$  also satisfies  $\varphi$ :  $\sigma \models \Phi$  implies  $\sigma \models \varphi$

In symbols,

$$\Phi \models \varphi$$

**Example 3.**

$$\varphi, \varphi \rightarrow \psi \models \psi.$$

This is the famous *modus ponens*, a classical rule of inference.

**Example 4.** From  $\Phi, \varphi \models \perp$  it follows that  $\Phi \models \neg \varphi$ .

**Example 5.** From  $\Phi, \varphi \models \psi$  it follows that  $\Phi \models \varphi \rightarrow \psi$ .

## Tautologies, Contradictions and Contingencies

**Definition.** A formula  $\varphi$  is a **tautology** if it is true under all truth assignments. In symbols,

$$\models \varphi$$

A formula  $\varphi$  is a **contradiction** if no assignment satisfies  $\varphi$ .

A formula  $\varphi$  is a **contingency** (or **satisfiable**) if there is some assignment that satisfies  $\varphi$ .

Tautologies are what corresponds to the informal notion of a "true" statement or a "valid" statement (at least in propositional logic).

**Example 6.**  $p \wedge \neg p$  is a contradiction,  $p \vee \neg p$  is a tautology.

**Exercise 2.** Show that all the implications from above are tautologies.

## Associated Problems

Naturally, we can associate decision problems with all these definitions. Finding good algorithms to solve these problems computationally will turn out to be quite a challenge.

**Problem: Tautology**

Instance: A propositional formula  $\varphi$ .

Question: Is  $\varphi$  a tautology?

**Problem: Satisfiability**

Instance: A propositional formula  $\varphi$ .

Question: Is  $\varphi$  a contingency?

**Problem: Satisfiability (search)**

Instance: A propositional formula  $\varphi$ .

Solution: A satisfying truth assignment.

## A Connection

Likewise we could introduce a decision problem **Contradiction**.

Note that these problems are all closely related, though:

- $\varphi$  is a tautology iff  $\neg\varphi$  is not satisfiable.
- $\varphi$  is a contradiction iff  $\varphi$  is not satisfiable.

In practice, Satisfiability testing algorithms are particularly important.

Hence fast (and general, more later) algorithms for one problem could be used to solve the others.

## Truth Tables

Here is a brute-force attack: we construct a *truth table*, an idea apparently due to Lewis Carroll. The table is a brute-force list of the truth values for all possible assignments. For example the 3-variable formula

$$\varphi = (((p \wedge q) \rightarrow r) \wedge (p \rightarrow q)) \rightarrow (p \rightarrow r)$$

produces the table

$p$	$q$	$r$	$(p \wedge q) \rightarrow r$	$p \rightarrow q$	$p \rightarrow r$	$\varphi$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	0	0	1
1	0	1	1	0	1	1
1	1	0	0	1	0	1
1	1	1	1	1	1	1

So,  $\varphi$  is a tautology. This is easy to implement in principle: just one big loop ranging over all truth assignments, plus a little evaluation routine.

## But, it's useless . . .

Regrettably, this method falls apart if we have many variables: the table has  $2^n$  rows for  $n$  variables. In applications,  $n$  may be several thousand.

Let's suppose we have 500 variables. Let's further assume every stable particle in the universe (about  $10^{80}$ ) is a computer that can check  $10^9$  assignments per second. Lastly assume that the universe is 15 billion years old and that our "universal computer" has been running non-stop ever since the Big Bang. There are

$$3.27339 \cdot 10^{150}$$

truth assignments to check, but so far we would have handled only

$$4.7304 \cdot 10^{106}$$

of these assignments, a minuscule fraction.

## Equivalence

Perhaps one could somehow simplify formulae so that it's easy to check whether they are tautologies or contingencies?

**Definition 4.** Two formulae  $\varphi$  and  $\psi$  are **equivalent** if for any truth assignment  $\sigma$ :  $[\varphi]_\sigma = [\psi]_\sigma$ .

In symbols:

$$\varphi \equiv \psi$$

**Claim 1.**  $\varphi$  and  $\psi$  are equivalent if, and only if,  $\varphi \leftrightarrow \psi$  is a tautology.

*Proof.* To see this note

$$\begin{aligned} [\varphi \leftrightarrow \psi]_\sigma &= H_{equ}([\varphi]_\sigma, [\psi]_\sigma) \\ &= H_{equ}([\varphi]_\sigma, [\varphi]_\sigma) = 1 \end{aligned}$$

□

## Some Equivalences

$$\varphi \wedge \perp \equiv \perp$$

$$\varphi \wedge \top \equiv \varphi$$

$$\varphi \wedge \varphi \equiv \varphi$$

$$\varphi \wedge \psi \equiv \psi \wedge \varphi$$

$$\varphi \wedge (\psi \wedge r) \equiv (\varphi \wedge \psi) \wedge r$$

$$\varphi \wedge (\psi \vee r) \equiv (\varphi \wedge \psi) \vee (\varphi \wedge r)$$

$$\varphi \wedge \psi \equiv \neg(\neg\varphi \vee \neg\psi)$$

$$\varphi \rightarrow \perp \equiv \neg\varphi$$

**Exercise 3.** Establish all these equivalences.

### Minor Digression: Implications

In the formula  $\varphi \rightarrow \psi$

- $\varphi$  is the antecedent, hypothesis or premise
- $\psi$  is the consequent or conclusion

No one has any problems understanding the meaning of conjunction, disjunction and negation, but some feel uneasy about implications: if the hypothesis is false, the whole implication is true regardless of the conclusion.

So  $\perp \rightarrow \varphi$  is a tautology, regard less of  $\varphi$ .

Some people find this counterintuitive.

**Claim 2.**  $\varphi \rightarrow \psi$  is equivalent to  $\neg\varphi \vee \psi$ , and to  $\neg(\varphi \wedge \neg\psi)$ .

### Variants

Any implication can be associated with 3 natural variants.

Implication	$\varphi \rightarrow \psi$
Converse	$\psi \rightarrow \varphi$
Inverse	$\neg\varphi \rightarrow \neg\psi$
Contrapositive	$\neg\psi \rightarrow \neg\varphi$

**Claim 3.** An implication and its contrapositive are equivalent.

However, in general an implication is not equivalent to its converse, nor it inverse.

### Deciding Equivalences

It is immediate from the definitions that

- $\varphi$  is a tautology iff  $\varphi \equiv \top$
- $\varphi$  is a contradiction iff  $\varphi \equiv \perp$
- $\varphi$  is satisfiable iff not  $\varphi \equiv \perp$

Hence testing whether two formulae are equivalent is at least as hard as our decision problems.

It's no worse, though: recall that two formulae are equivalent iff  $\varphi \iff \psi$  is a tautology.

Thus Equivalence Testing and Tautology are of the same complexity.

### Equivalences and Algebra

These equivalences can be used to simplify complicated formulae.

Note that this is very similar to our analysis of digital circuits in terms of Boolean algebra.

- Instead of circuits we have formulae,
- instead of inputs we have truth assignments,
- instead of equations we have equivalences.

But other than that, we have the same laws.

Needless to say, this cannot be coincidence. Before we explore this idea, let's pin down a formal system for propositional logic.

## Deduction Systems

### A Deduction System

If we cannot determine efficiently whether a given formula is a tautology, perhaps we generate all tautologies?

Perhaps there is some nice inductive definition?

If we model our system after the pattern of logical reasoning we need

- **Axioms:** pick a few simple tautologies.
- **Rules of Inference:** rules to form more complicated tautologies from simpler ones.

Of course, there are many ways such a system can be organized. The ones that are easiest to explain typically do a very poor job at modeling actual reasoning.

## Hilbert Style System

For simplicity, the following system uses only two connectives:  $\neg$  and  $\vee$  (the others could be defined from these).

Logical Axioms:

$$\text{tertium non datur} \quad \neg\varphi \vee \varphi$$

Rules of Inference:

$$\begin{array}{l} \text{expansion} \quad \frac{\varphi}{\psi \vee \varphi} \\ \text{contraction} \quad \frac{\varphi \vee \varphi}{\varphi} \\ \text{associativity} \quad \frac{\varphi \vee (\psi \vee \chi)}{(\varphi \vee \psi) \vee \chi} \\ \text{cut rule} \quad \frac{\varphi \vee \psi \quad \neg\varphi \vee \chi}{\psi \vee \chi} \end{array}$$

## Gödel Style Proofs

**Definition.** A **derivation** or a **proof** in this system is just a sequence of applications of the rules:

$$\varphi_1, \varphi_2, \varphi_3, \dots, \varphi_n$$

where each  $\varphi_i$  is either an axiom, a premise or obtained from previous  $\varphi_j$ 's by a rule of inference.

Notation:

$$\underbrace{\psi_1, \psi_2, \dots, \psi_n}_{\text{premises}} \vdash \underbrace{\psi}_{\text{conclusion}}$$

If there are no premises one writes

$$\vdash \psi$$

## Sample Derivation

**Claim 4.**  $p, \neg p \vee q \vdash q$

1	$p$	<i>prem</i>
2	$q \vee p$	<i>e, 1</i>
3	$\neg q \vee q$	<i>axiom</i>
4	$p \vee q$	<i>cut, 2, 3</i>
5	$\neg p \vee q$	<i>prem</i>
6	$q \vee q$	<i>cut, 4, 5</i>
7	$q$	<i>c, 6</i>

We have annotated the formulae by the proof rules that were used to obtain them. Note that this annotation makes it trivial to check correctness.

## In Tree Form

Alas, linear proofs are hard to read, even with annotations. For humans, a corresponding proof tree is often much easier to deal with.

$$\frac{\frac{\frac{p}{q \vee p} (e) \quad \neg q \vee q (cut)}{p \vee q} \quad \neg p \vee q (cut)}{\frac{q \vee q (c)}{q} (c)}$$

The root is the proven formula, and the hypotheses and axioms are at the leaves. The labels indicate the type of rule used and there is no need for back-references.

**Exercise 4.** Explain the connection between the tree and the linear form of the proof.

## Headache

A **formal proof** like this is blindingly boring: it is comprised of a long, long sequence of tiny little steps that all tend to be trivial individually – though the whole proof well be incomprehensible.

But note the good news: the steps are so simple that they can easily be verified by a computer (proof checker).

So in principle we could check all proofs for accuracy.

In reality, interesting proof tend to be very long when formalized, and it is not so clear what is actually meant when someone says “in principle we could rewrite this as a formal proof”.

- But see <http://mizar.org>, and
- google for Bourbaki.

## Soundness

For any deduction system to be useful, it has to be **sound**: in our case, we have to make sure that only tautologies can be derived (without premises).

**Theorem.** *Soundness Theorem*

*The system is sound: only tautologies can be derived in it.*

This is easy to see for a Hilbert system by induction:

- The axioms are tautologies.
- Application of any proof rule to given tautologies produces another tautology.

## Completeness

The second requirement is much harder to deal with: we want the system to be *complete* so that all tautologies can be derived.

### Theorem. Completeness Theorem

*The system is complete: all tautologies can be derived in it.*

Given the fact that our example system has only one kind of axiom and relatively simple rules of inference it is by no means clear that we do not miss out on some tautologies.

What if the formula has thousands of variables and the parse tree is hundreds of levels deep?

We give a sketch of the completeness proof, make sure you are able to supply all the necessary details.

## Sketch

**Lemma 1.** *Let  $1 \leq i_1, i_2, \dots, i_m \leq n$ .*

$$\varphi_{i_1} \vee \varphi_{i_2} \vee \dots \vee \varphi_{i_m} \vdash \varphi_1 \vee \varphi_2 \vee \dots \vee \varphi_n$$

*Proof.* Ugly induction on  $m$ . □

To lighten notation, write  $\psi \rightarrow \varphi$  for  $\neg\psi \vee \varphi$ .

**Theorem 1. Deduction Theorem**

$\Phi, \psi \vdash \varphi$  implies  $\Phi \vdash \psi \rightarrow \varphi$ .

*Proof.* Induction on the length of the derivation of  $\varphi$  from  $\Phi, \psi$ . □

## Sketch, contd.

For the completeness proof it suffices to show that any tautology of the form

$$\varphi = \varphi_1 \vee \varphi_2 \vee \dots \vee \varphi_n$$

is provable in the system.

This is done by induction on the total number of connectives in  $\varphi$ .

Note: not induction on  $n$ , which is what one would probably try first.

**Exercise 5.** *Give a detailed proof of the deduction theorem.*

**Exercise 6.** *Carry out the details of the soundness and completeness proof.*

## Consistency

Intuitively, if we add more and more formulae to a given collection  $\Gamma$  it becomes harder and harder to find a satisfying truth assignment. When too many requirements have been added  $\Gamma$  becomes contradictory. How is this reflected in our deduction system?

**Definition 5.** *A set  $\Gamma$  of formulae is consistent if  $\perp$  is not provable from  $\Gamma$ .*

It is clear that any inconsistent set of formulae cannot have a satisfying truth assignment:  $\Gamma \vdash \varphi$  and  $\sigma \models \Gamma$  immediately implies that  $\sigma(\perp) = \text{tt}$ , a contradiction.

Much more surprising is that consistency is all that is needed to guarantee satisfiability.

## Consistency Theorem

**Theorem 2.** *If  $\Gamma$  is consistent, then there exists a truth assignment  $\sigma$  such that  $\sigma \models \Gamma$ .*

*Proof.*

Assume that for all  $\sigma$ ,  $\sigma \not\models \Gamma$ .

Then for all  $\sigma$ ,  $\sigma \models \Gamma$  implies  $\sigma \models \perp$  (recall: ex falsum quodlibet).

But then  $\Gamma \models \perp$ .

By Completeness,  $\Gamma \vdash \perp$ , i.e.,  $\Gamma$  is inconsistent. □

Note that Completeness is critical here to transfer a statement about semantic consequence into a statement about derivations.

## Kalmar's Lemma

Another proof of completeness is based on the following lemma. For any formula  $\varphi$  and valuation  $\sigma$  define

$$\hat{\varphi} = \begin{cases} \varphi & \text{if } \llbracket \varphi \rrbracket_{\sigma} = 1, \\ \neg\varphi & \text{otherwise.} \end{cases}$$

**Lemma 2.** *Let  $\varphi(p_1, \dots, p_n)$  be a formula with propositional variables as indicated,  $\sigma$  a valuation. Then  $\hat{p}_1, \dots, \hat{p}_n \vdash \hat{\varphi}$*

*Proof.* By induction on the build-up of  $\varphi$ . □

**Lemma 3.** *If  $\Gamma, \psi \vdash \varphi$  and  $\Gamma, \neg\psi \vdash \varphi$  then  $\Gamma \vdash \varphi$ .*

*Proof.* Easy using the Deduction Theorem. □

**Exercise 7.** *Give an alternative proof of completeness based on the last two lemmata.*

### Other Systems

There is nothing sacred about the Hilbert system from above, sound and complete deduction systems for propositional logic can be constructed in many different ways. Here is just one other system, this one based on connectives  $\neg$  and  $\rightarrow$ .

There is only one rule of inference, the modus ponens  $\frac{A, A \rightarrow B}{B}$ . But there are three types of axioms:

$$A \rightarrow (B \rightarrow A) \quad (K)$$

$$(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)) \quad (S)$$

$$(\neg A \rightarrow \neg B) \rightarrow (B \rightarrow A) \quad (T)$$

Again it is easy to show that this system is sound, but completeness requires some effort.

### The $K$ and $S$ Axioms

The choice of axioms may seem more or less arbitrary; as long as the formulae are tautologies they won't do any harm – but there is the question of elegance, we want few axioms and preferably they should make intuitive sense.

For the last example, there is an excellent reason to choose  $(K)$  and  $(S)$ . Let  $\mathcal{S}$  be a system for propositional logic that contains modus ponens as a rule of inference.

**Theorem 3.**  $\mathcal{S}$  satisfies the Deduction Theorem if, and only if,  $\mathcal{S}$  proves all instances of axioms  $(K)$  and  $(S)$ .

**Exercise 8.** Show that  $A \rightarrow A$  is derivable in the last system.

**Exercise 9.** Prove the last theorem. Left to right is easy, for the opposite direction use induction on the length of a Gödel proof for  $\Gamma, A \vdash B$ .

### Back To Algebra

Given a collection  $\Gamma$  of propositional formulae we can construct a Boolean algebra as follows. Let  $F$  be the collection of all propositional formulae. We define an equivalence relation on  $F$  by

$$\varphi \equiv \psi \iff \Gamma \vdash (\varphi \leftrightarrow \psi)$$

Now define operations on the quotient structure  $F/\equiv$ :

$$[\varphi] \vee [\psi] = [\varphi \vee \psi]$$

$$[\varphi] \wedge [\psi] = [\varphi \wedge \psi]$$

$$\neg[\varphi] = [\neg\varphi]$$

Then  $(F/\equiv, \vee, \wedge, \neg, [\perp], [T])$  is a Boolean algebra that provides a lot of information about  $\Gamma$ .

This is the so-called Lindenbaum algebra of  $\Gamma$ .

## Natural Deduction

### A Usable Formal System

Hilbert style systems are notoriously difficult to use in actual applications. To get a more practical system, consider how one proves an implication  $p \rightarrow q$  in real life.

- First, assume  $p$ , then
- argue around till  $q$  pops up, hopefully, and lastly
- conclude that  $p$  indeed implies  $q$  (discharge the assumption).

Systems that allow this type of reasoning are called *natural deduction systems* and are due to G. Gentzen. They are much easier to use than Hilbert style systems, but a bit harder to explain: the rules of inference are more complicated than in a Hilbert system.

Typically we have an *introduction rule* and an *elimination rule* for each logical connective.

The good news: we do not need any logical axioms at all! Everything is built into the rules. Also note that all connectives appear directly, there is no need to define them in terms of others.

### Natural Deduction Rules

**And**

$$\frac{\phi \quad \psi}{\phi \wedge \psi} (\wedge i) \quad \frac{\phi \wedge \psi}{\phi} (\wedge e) \quad \frac{\phi \wedge \psi}{\psi} (\wedge e)$$

**Or**

$$\frac{\phi}{\phi \vee \psi} (\vee i) \quad \frac{\psi}{\phi \vee \psi} (\vee i) \quad \frac{\phi \vee \psi \quad \begin{array}{c} [\phi] \\ \vdots \\ \chi \end{array} \quad \begin{array}{c} [\psi] \\ \vdots \\ \chi \end{array}}{\chi} (\vee e)$$

**Implication**

$$\frac{\phi \quad \phi \rightarrow \psi}{\psi} (\rightarrow e) \quad \frac{\neg\psi \quad \phi \rightarrow \psi}{\neg\phi} (mt) \quad \frac{\begin{array}{c} [\phi] \\ \vdots \\ \psi \end{array}}{\phi \rightarrow \psi} (\rightarrow i)$$

## Discharging Assumptions

The notation

$$\frac{\phi \vee \psi \quad \begin{array}{c} [\phi] \\ \vdots \\ \chi \end{array} \quad \begin{array}{c} [\psi] \\ \vdots \\ \chi \end{array}}{\chi} (\vee e)$$

bears some explanation:

- Assume  $\phi$  and reason that  $\chi$  is a consequence.
- Assume  $\psi$  and reason that  $\chi$  is a consequence.
- If we have  $\phi \vee \psi$  we can discharge the assumptions  $\phi$  and  $\psi$  and conclude  $\chi$ .

In reality, assumptions should be tagged (say, by a numerical subscript) so that one can keep track of where they are discharged.

► No proof is complete before all assumptions have been discharged.

## Rules for Negation

Falsum, negation

$$\frac{}{\perp} (\perp e) \quad \frac{\neg\phi \quad \phi}{\perp} (\neg e) \quad \frac{[\phi] \quad \dots}{\neg\phi} (\neg i)$$

Double negation

$$\frac{\neg\neg\phi}{\phi} (\neg\neg e) \quad \frac{\phi}{\neg\neg\phi} (\neg\neg i)$$

The first rule is "ex falsum quodlibet": once you have a contraction, anything follows.

The third really is proof by contradiction.

## The Indirect Rules

Rules like  $(\wedge e)$  or  $(\vee i)$  are easy to use.

But there are 3 rules where an *assumption* is made temporarily, and then discharged later.

These are tricky, but crucial for the system.

**Or Elimination**

$$\frac{\phi \vee \psi \quad \begin{array}{c} [\phi] \\ \vdots \\ \chi \end{array} \quad \begin{array}{c} [\psi] \\ \vdots \\ \chi \end{array}}{\chi} (\vee e)$$

**Implication Introduction**

$$\frac{\begin{array}{c} [\phi] \\ \vdots \\ \psi \end{array}}{\phi \rightarrow \psi} (\rightarrow i)$$

**Negation Introduction**

$$\frac{[\phi] \quad \dots}{\neg\phi} (\neg i)$$

## Some Derivations

We prove  $p \wedge q, r \vdash q \wedge r$ .

1	$p \wedge q$	premise
2	$r$	premise
3	$q$	$\wedge e_2, 1$
4	$q \wedge r$	$\wedge i, 3, 2$

The next proof is slightly more complicated. We will prove that  $(p \wedge q) \wedge r \vdash p \wedge (q \wedge r)$ .

1	$(p \wedge q) \wedge r$	premise
2	$p \wedge q$	$\wedge e_1, 1$
3	$p$	$\wedge e_1, 2$
4	$q$	$\wedge e_2, 2$
5	$r$	$\wedge e_2, 1$
6	$q \wedge r$	$\wedge i, 4, 5$
7	$p \wedge (q \wedge r)$	$\wedge i, 3, 6$

## Modus Tollens

We can show that a proof rule analogous to *modus tollens* is admissible in this system, in the following sense.

$$\frac{\neg\psi \quad \phi \rightarrow \psi}{\neg\phi} (mt)$$

We'll use variables  $p$  and  $q$  for clarity.

1	$p \rightarrow q$	premise
2	$p$	assmp.
3	$q$	$\rightarrow e, 1, 2$
4	$\neg q$	premise
5	$\perp$	$\neg e, 3, 4$
6	$\neg p$	$\neg i, 2-5$

Note that assumption line 2 is properly discharged in line 6.

## Derivations as Trees

**Claim 5.**  $p \vee q, p \rightarrow r, q \rightarrow s \vdash r \vee s$

$$\frac{p \vee q \quad \frac{[p] \quad p \rightarrow r}{r} (\rightarrow e) \quad \frac{[q] \quad q \rightarrow s}{s} (\rightarrow e)}{r \vee s} (\vee i) \quad \frac{}{r \vee s} (\vee e)$$

**Claim 6.**  $(p \rightarrow q) \rightarrow r \vdash p \rightarrow (q \rightarrow r)$

$$\frac{[p] \quad \frac{[q] \quad (p \rightarrow q) \rightarrow r}{q \rightarrow r} (\rightarrow e)}{p \rightarrow (q \rightarrow r)} (\rightarrow i)$$

Note that we also have shown  $(p \rightarrow q) \rightarrow r \vdash q \rightarrow r$

## Negation

Derivations involving negation can be a bit more problematic.

**Claim 7.**  $p \rightarrow q, \neg p \rightarrow q \vdash q$

1	$p \rightarrow q$	$prem$
2	$\neg p \rightarrow q$	$prem$
3	$\neg q$	$assm$
4	$\neg p$	$(mt), 1, 3$
5	$\neg \neg p$	$(mt), 2, 3$
6	$p$	$(\neg \neg e), 5$
7	$\perp$	$(\neg e), 4, 6$
8	$\neg \neg q$	$(\neg i), 3 - 7$
9	$q$	$(\neg \neg e), 8$

## Tertium Non Datur

We don't need tertium non datur as a logical axiom! It's built into our derivation rules.

**Corollary 1.**  $\vdash p \vee \neg p$

To see this note that  $p \rightarrow p \vee \neg p$  and  $\neg p \rightarrow p \vee \neg p$  are provable.

**Exercise 10.** Fill in the details in the last proof.

## Knights in Shiny Armor

Remember our chess problem? Can a knight starting at  $(1, 1)$  reach all squares on a chessboard?

Can describe the moves as a big set of assumptions  $\Gamma$ :

$$\{ p_{1,1} \rightarrow p_{2,3} \wedge p_{3,2}, \\ p_{1,2} \rightarrow p_{3,1} \wedge p_{3,3} \wedge p_{2,4}, \\ \dots, \\ p_{8,8} \rightarrow p_{7,6} \wedge p_{6,7} \}$$

Then

$$\Gamma, p_{1,1} \vdash p_{1,1} \wedge p_{1,2} \wedge \dots \wedge p_{8,8}$$

This is not even hard, just tedious: Use modus ponens and  $(\wedge_e)$  to get all  $p_{i,j}$ . Then use  $(\wedge_i)$  to assemble the conjunction.

## Deduction Theorem

**Theorem 4.**  $\Gamma, \phi \vdash \psi$  if, and only if,  $\Gamma \vdash (\phi \rightarrow \psi)$ .

*Proof.* (Sketch)

$\Leftarrow$ : Is easy, apply modus ponens.

$\Rightarrow$ : Is hard, use induction on the length of the proof of  $\psi$ . For simplicity, consider only a binary rule as the last step:

$$\frac{\gamma_1 \quad \gamma_2}{\psi} (R)$$

Then by IH,  $\Gamma \vdash \phi \rightarrow \gamma_i$ . Now consider

$$\frac{\frac{[\phi] \quad \phi \rightarrow \gamma_1 \quad (\rightarrow e)}{\gamma_1} \quad \frac{[\phi] \quad \phi \rightarrow \gamma_2 \quad (\rightarrow e)}{\gamma_2}}{\frac{\psi}{\phi \rightarrow \psi} \quad (\rightarrow i)} (R)$$

Similar arguments work in the remaining cases. □

## Meta-Theorem

Note that the Deduction Theorem is an assertion about derivations, not a derivation in the system itself.

Of course, Soundness and Completeness are also meta-theorems in this sense: they are assertions about the system, not in the system – in fact the system is nowhere near expressive enough to make these assertions.

This is the beauty of logic: one reasons about a system whose purpose it is to describe reasoning. And one uses essentially the same rules in doing so.

## Natural Deduction

Summarizing, the crucial features of natural deduction systems are

- Proofs take place in the context of assumptions.
- Arbitrary assumptions can be made at any time, but
- they all must be discharged to complete the proof.

This is all indeed very natural, but there are a few vexing details: one has to be careful about managing the assumptions. Here is a typical problem: prove

$$A \rightarrow A \wedge A$$

**Exercise 11.** Explain the difficulties that arise in a natural deduction proof of the last assertion.

## Sequent Calculus

It is often preferable to bring the assumption mechanism out into the open.

**Definition 6.** A **sequent** consists of two finite sets of formulae  $\Gamma$  and  $\Delta$ .  $\Gamma$  is the **antecedent** and  $\Delta$  is the **consequent** of the sequent.

*Notation:*

$$\Gamma \supset \Delta.$$

The idea is that the conjunction of the formulae in  $\Gamma$  (all the assumptions) implies the disjunction of the formulae in  $\Delta$ .

It may sound tempting to replace  $\Delta$  by a single formula but it turns out that the given version is more appropriate for our purposes.

We will think of the sets of formulae as sets rather than lists, so order and multiplicity do not matter. Of course, in implementations the opposite choice is more natural.

## Axioms

Keeping the interpretation of “conjunction of antecedent implies disjunction of consequent” in mind, all sequents where the antecedent and consequent contain the same formula are trivially valid.

These are sometimes called **basic sequents**.

All basic sequents will be adopted as axioms:

$$\Gamma, A \supset A, \Delta.$$

In natural deduction this corresponds to inferring  $A$  from assumption  $A$ .

## Summary

- Propositional reasoning sometimes suffices for simple arguments.
- It is often used in more complicated arguments to simplify intermediate results.
- The semantics of a propositional formula can be expressed in terms of truth tables (which are, alas, of exponential size).
- Tautologies are the “true” formulae of propositional logic.
- Formal deductive systems can be used to derive all tautologies.
- Soundness of such systems is usually trivial to see, but completeness may require a bit of effort.