CS 745 / ECE 725
Computer Aided Verification

Lecture 2: Propositional Logic

Jo Atlee

DC 2337, jmatlee@uwaterloo.ca

Office Hours: Mon 1:00-2:00, Wed 1:00-2:00

http://www.student.cs.uwaterloo.ca/~cs745

Announce

Anyone registered in CS745 will automatically receive an account on mudge.waterloo.ca (unless you already have an account).

The password will be set to the password of your general CS account.

Today's Agenda

- What is logic?
- Propositional Logic

Review: Verification

Verification involves checking a satisfaction relation, usually in the form of a sequent:

$$\mathcal{M} \models \phi$$

where

- $\mathcal{M}$ is a model (or implementation)
- ϕ is a property (or specification)
- $\models$ is a relationship that should hold between $\mathcal{M}$ and ϕ, i.e., $(\mathcal{M}, \phi) \in \models$

We say that the model satisfies or "has" the property, or that we can conclude the property from the model.

Models and Properties

The term “model” is used loosely here. It might not be executable, and it might not be a complete description of the system’s behaviour.

The terms “implementation” and “specification” are relative. An implementation generally contains more details than a specification. The specification for one level of verification might be the implementation at a higher level of verification.

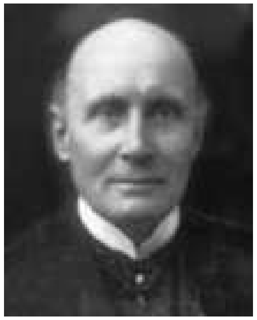
In hardware, often the model is a description of the circuit in a hardware description language such as VHDL or Verilog. The real thing is the physical realization of the chip.

Sometimes the model is actually a specification and the property is an attribute such as completeness or consistency.

Hilbert, Russell, Whitehead

David Hilbert

Bertrand Russell



Alfred Whitehead

Russell and Whitehead wrote Principia Mathematica (1910–1913).

Src: http://www-groups.dcs.st-and.ac.uk/history/BiogIndex.html

Gödel’s Incompleteness Theorem

“This theorem is one of the most important proven in this century, ranking with Einstein’s Theory of Relativity.”

In 1931, the Czech-born mathematician Kurt Gödel demonstrated that “in any axiomatic mathematical system there are propositions that cannot be proved or disproved within the axioms of the system.”

Src:

http://www-groups.dcs.st-and.ac.uk/ history/Mathematicians/Godel.html

References

- John Kelly, *The Essence of Logic*. Prentice Hall, 1997.
- Michael Huth and Mark Ryan. *Logic in Computer Science* Cambridge University Press, 2000. (DC Library: QA76.9.L63 H88 2000 – requested to be on 1 day reserve) (don’t forget to check the errata)
- Nimal Nissanke, *Introductory Logic and Sets for Computer Scientists*, Addison-Wesley, 1999. (SE112, CS245 text)

More advanced:

- Melvin Fitting. *First-Order Logic and Automated Theorem Proving*. Springer, 1996. Second Edition. (DC Library: QA76.9.A96F68 1996 – requested to be on 1 day reserve)

. . . and many more!

What is a Logic?

According to Kelly [Kel97], p. 1:

> In general, logic is about reasoning. It is about the validity of arguments, consistency among statements (. . .) and matters of truth and falsehood. In a formal sense logic is concerned only with the form of arguments and the principles of valid inferencing.

Induction vs Deduction

These are two branches in the philosophical study of logic.

Induction

- From specific to general
- $P(0), P(1)$, ...; therefore: $\forall i \bullet P(i)$.

Deduction

- From general to specific
- “All humans are mortal”; therefore “Socrates is mortal”

Induction

Induction is “the process of deriving general principles from particular facts or instances. ” [FrO]

Example:
Coffee shop burger #1 was greasy.
Coffee shop burger #2 was greasy.
Coffee shop burger #3 was greasy.
. . .
Coffee shop burger #100 was greasy.
Therefore, all coffee shop burgers are greasy. [Sub02]

In induction, conclusions are probable but not conclusive.

To use induction to conclusively prove a theorem, deductively apply the theorem of mathematical induction (see **??**).

Deduction

Deduction is “the process of reasoning in which a conclusion follows necessarily from the stated premises; inference by reasoning from the general to the specific.” [FrO].

Mathematical Induction: a method of proving statements about well-ordered sets. The most common use of mathematical induction is for the natural numbers where there is a base case and an induction hypothesis. Mathematical induction is a form of deduction because the conclusions are conclusive.

We will be studying deduction and using mathematical induction.

Elements of a Logic

A logic consists of:

1. syntax
2. semantics
3. proof procedure(s) (also called proof theory)

Syntax and Semantics

- syntax:
  - define “well-formed formula”
- semantics:
  - define “ $\models$ ” (“satisfies”)
    $\mathcal{M} \models \phi$ (satisfaction relation)
  - define $\phi_1, \phi_2, \phi_3, \ldots \models \psi$ (“entails”, or semantic entailment) means:

    from the premises $\phi_1, \phi_2, \phi_3, \ldots$,
    we may conclude ψ,
    where $\phi_1, \phi_2, \phi_3, \ldots$ and ψ are all well-formed formulae in the logic

Proof Procedure

- proof procedure(s):
  - define “ $\vdash$ ” (pronounced “proves”)
  - a proof procedure is a way to calculate $\phi_1, \phi_2, \phi_3, \ldots \vdash \psi$ (also called a sequent). By “calculation”, we mean that there is a procedure for deriving ψ from $\phi_1, \phi_2, \phi_3, \ldots$
  - there may be multiple proof procedures, which we will indicate by subscripting $\vdash$, e.g., the natural deduction proof procedure for propositional logic will be $\vdash_{\text{ND}}$
  - for some logics, there isn’t a proof procedure that always terminates for any sequent

Soundness and Completeness

The semantics and the proof procedures ($\models$ and $\vdash$) are related in the concepts of soundness and completeness.

Definition. A proof procedure is sound if $\phi_1, \phi_2, \phi_3 \vdash \psi$ then $\phi_1, \phi_2, \phi_3 \models \psi$.

A proof procedure is sound if it proves only tautologies.

Definition. A proof procedure is complete if $\phi_1, \phi_2, \phi_3 \models \psi$ then $\phi_1, \phi_2, \phi_3 \vdash \psi$.

A proof procedure is complete if it proves every tautology.

Note that in the literature, there is not consistent use of the symbols $\models$ and $\vdash$.

Consistency

Definition. A proof procedure is consistent if it is not possible to prove both A and $\neg A$, i.e.,

$$\text{not both} \vdash A \;\; \text{and} \vdash \neg A$$

Today's Agenda

- What is logic?
- **Propositional Logic**

Propositional Logic

Invented by George Boole (1815-64). "An Investigation of the Laws of Thought on which are founded The Mathematical Theories of Logic and Probabilities".

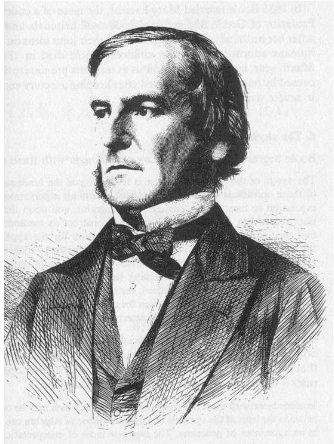


Image scanned from "Makers of Mathematics" by Stuart Hollingdale, Penguin Books, 1994.

Propositional Logic

Propositional logic is also called sentential logic, i.e., the logic of sentences. It is also called propositional calculus or sentential calculus.

1. syntax (well-formed formulas)
2. semantics (truth tables)
3. proof theory
   - axiom systems
   - natural deduction
   - sequent calculus
   - and others

Propositional Logic: Syntax

Its syntax consists of:

1. Two constant symbols: **true** and **false**
2. Proposition letters: traditionally lowercase letters $p, q, r, ...$
3. Propositional connectives: $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$
4. Brackets: ( ), [ ]

Propositions

Definition. Proposition letters represent declarative sentences, i.e., sentences that are true or false. Sentences matching proposition letters are atomic (non-decomposable), meaning they don’t contain any of the propositional connectives.

Here are some examples:

- It is raining outside.
- The sum of 2 and 5 equals 3.
- The value of program variable a is 42.

Sentences that are interrogative (questions), or imperative (commands) are not propositions.

Using Symbols

Because in logic, we are only concerned with the structure of the argument and which structures of arguments are valid, we “encode” the sentences in symbols to create a more compact and clearer representation of the argument. We call these propositional symbols or proposition letters.

DO NOT use T, F, t, or f in any font as symbols representing sentences!

Propositional Connectives

Definition. The propositional (logical) connectives are:

Symbol	Informal Meaning
$\neg$	negation (not)
$\wedge$	conjunction (and, both)
$\vee$	disjunction (or, at least one of)
$\Rightarrow$	implication (implies, logical consequence, conditional, if . . . then)
$\Leftrightarrow$	equivalent (biconditional, if and only if)

Others may use different symbols for these operations.

Terminology

For an implication $p \Rightarrow q$:

- p is the premise or antecedent or hypothesis
- q is the consequent or conclusion

$\neg b \Rightarrow \neg a$ is called the contrapositive of $a \Rightarrow b$.

The set of connectives $\{\neg, \wedge\}$ are complete in the sense that all the other connectives can be defined using them, e.g., $a \vee b = \neg(\neg a \wedge \neg b)$. Other subsets of the binary connectives are also complete in the same sense.

Example of Using Symbols

Example: If the train arrives late and there are no taxis at the station, then John is late for his meeting. John is not late for his meeting. The train did arrive late. Therefore, there were taxis at the station.

$$\begin{array}{l} (p \wedge \neg q) \Rightarrow r \\ \neg r \\ p \\ \hline q \end{array}$$

Prop. Letter	Sentence
p	the train is late
q	there are taxis at the station
r	John is late for his meeting

Well-formed formulas

The following is an expression formed out of propositional symbols, brackets, and propositional connectives:

$$a(\wedge c \Rightarrow)b$$

but it's not a formula in propositional logic! Next, we make precise the notion of a well-formed formula

Well-formed formulas (modified)

Definition. The well-formed formulae of propositional logic are those obtained by the following construction rules:

1. **true**, **false**, and the proposition letters are atomic formulas.
2. If ϕ and ψ are formulas, then each of the following are formulas:

$$(\neg\phi) \quad (\phi \wedge \psi) \quad (\phi \vee \psi) \quad (\phi \Rightarrow \psi) \quad (\phi \Leftrightarrow \psi)$$

No other expressions are formulas.

Note that this is an inductive definition, meaning the set is defined by basis elements, and rules to construct elements from elements in the set.

Well-formed Formulas

Brackets around the outermost formula are usually omitted. Brackets can be omitted using the following rules of precedence of operators: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$.

Associativity: $\Rightarrow$ is right associative meaning $p \Rightarrow q \Rightarrow r$ is $p \Rightarrow (q \Rightarrow r)$.

Note: Some texts do not use exactly these rules of precedence, they rank $\wedge$ and $\vee$ at the same level of precedence, and $\Rightarrow$ and $\Leftrightarrow$ at the same level of precedence.

Propositional Logic

1. syntax (well-formed formulas)
2. semantics (truth tables)
3. proof theory
   - axiom systems
   - natural deduction
   - sequent calculus

Semantics

Semantics means “meaning”. Semantics relate two worlds. Semantics provide an interpretation (mapping) of expressions in one world in terms of values in another world.

The second world is called the semantic domain.

In propositional logic, semantics map formulas (syntax) onto truth values (T, F).

Proof procedures transform the syntax of a logic in ways that respect the semantics.

Boolean Valuations

Classical logic is two-valued. The two possible truth values are T, and F, which are two distinct values.

Let $\mathbb{Tr} = \{\mathsf{T}, \mathsf{F}\}$ be the set of truth values.

The valuation (or interpretation) of a formula ϕ is an assignment of each atomic proposition in ϕ to a truth value.

Here’s an example valuation for the formula $(p \Rightarrow q) \wedge r$:

$$v(p) = \mathsf{T}, v(q) = \mathsf{F}, v(r) = \mathsf{F}, v(\textbf{false}) = \mathsf{F}, v(\textbf{true}) = \mathsf{T}$$

Note that there are 2^n valuations for a formula that has n atomic propositions.

Semantics of Propositional Connectives

The semantics of propositional connectives are defined as functions over $\mathbb{T}\mathrm{r}$, whose arguments are truth values and whose result is a truth value.

We specify these functions using truth tables.

p	$\neg p$
T	F
F	T

p	q	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

Evaluations of Formulae

We evaluate a formula ϕ by computing the result of its top-level connective as applied to the value(s) of the connective's argument(s) – recursively evaluating the arguments if they are not atomic:

Definition. The value of a formula ϕ, written $v(\phi)$, is recursively defined as follows:

$$
\begin{array}{ll}
v(\textbf{true}) = \text{T} & v(\phi \vee \psi) = v(\phi) \vee v(\psi) \\
v(\textbf{false}) = \text{F} & v(\phi \wedge \psi) = v(\phi) \wedge v(\psi) \\
v(\neg\phi) = \neg(v(\phi)) & v(\phi \Rightarrow \psi) = v(\phi) \Rightarrow v(\psi) \\
 & v(\phi \Leftrightarrow \psi) = v(\phi) \Leftrightarrow v(\psi)
\end{array}
$$

Note that $\neg(v(\phi))$ and $v(\phi) \circ v(\psi)$ is given by the truth tables on the previous slide.

Evaluations of Formulae

Here's an example valuation for the formula $(p \Rightarrow q) \wedge r$:

$$v(p) = \text{T}, v(q) = \text{F}, v(r) = \text{F}, v(\textbf{false}) = \text{F}, v(\textbf{true}) = \text{T}$$

We *evaluate* formula $(p \Rightarrow q) \wedge r$ given the above *valuation* by (1) recursively parsing the formula into subformulae, and (2) using the semantics of the connectives to compute the values of the subformulae:

$$
\begin{array}{rcl}
v((p \Rightarrow q) \wedge r) & = & v(p \Rightarrow q) \wedge v(r) \\
 & = & (v(p) \Rightarrow v(q)) \wedge v(r) \\
 & = & (\text{T} \Rightarrow \text{F}) \wedge \text{F} \\
 & = & \text{F} \wedge \text{F} \\
 & = & \text{F}
\end{array}
$$

A formula's value is uniquely determined by the values of its atomic propositions.

Terminology

- A formula is satisfiable if it is satisfied by **some** valuation.
- A formula is a tautology if it is satisfied by **all** valuations.
- A formula is a contradiction if satisfied by **no** valuation.

We sometimes say that the formula "has a satisfying assignment" to mean that it is satisfiable.

In Software Engineering, we are mostly interested in valuations that satisfy a given formula. Such valuations are called models of the formula.

Note that a formula a is a tautology iff $\neg a$ is not satisfiable.

Semantic Entailment

$$\phi_1, \phi_2, \phi_3 \models \psi$$

means that for all valuations where $v(\phi_1) = v(\phi_2) = v(\phi_3) =$ T, then $v(\psi) =$ T

Which is equivalent to saying $(\phi_1 \wedge \phi_2 \wedge \phi_3) \Rightarrow \psi$

$$\boxed{(\phi_1, \phi_2, \phi_3 \models \psi) \quad \equiv \quad ((\phi_1 \wedge \phi_2 \wedge \phi_3) \Rightarrow \psi)}$$

Models and Entailment

In propositional (and predicate) logic, $\models$ is overloaded and has two meanings (see Huth and Ryan [HR00], p. 137, p. 149):

- $\mathcal{M} \models \phi$ relates a model to a formula, saying that $\mathcal{M}$ satisfies the formula ϕ. This is called a satisfaction relation.
- $\psi \models \phi$ relates two formulas, saying that forall valuations, if $v(\psi) =$ T then $v(\phi) =$ T. This is called semantic entailment.

These two uses can be distinguished by their context.

Consistency

Definition. A collection of formulae is consistent if the formulae can all be true simultaneously.

A collection of formulas is consistent if there is a valuation that satisfies all formulae.

Consistency

Inconsistent premises are a problem because they can be used to prove a contradiction, i.e.,

$$p, \neg p \models \textbf{false}$$

Worse, they can be used to "prove" anything

$$p, \neg p \models \phi$$

It is standard practice in verification to check that one's premises are not inconsistent to avoid this problem.

Example of Checking Consistency

Are the following statements consistent?

> Sales of houses fall off if interest rates rise.
> Auctioneers are not happy if sales of houses fall off.
> Interest rates are rising. Auctioneers are happy.

Src: Kelly [Kel97], p. 15

Example of Checking Consistency

Does the following have a satisfying assignment?

$$(r \Rightarrow s) \wedge (s \Rightarrow \neg h) \wedge r \wedge h$$

s	r	h	$r \Rightarrow s$	$s \Rightarrow \neg h$	$(r \Rightarrow s) \wedge (s \Rightarrow \neg h) \wedge r \wedge h$
F	F	F	T	T	F
F	F	T	T	T	F
F	T	F	F	T	F
F	T	T	F	T	F
T	F	F	T	T	F
T	F	T	T	F	F
T	T	F	T	T	F
T	T	T	T	F	F

Decidability

A question is decidable if there is an algorithm that will always terminate and deliver the correct answer.

A logic is decidable if there is an algorithm to determine if any formula of the logic is a tautology (is a theorem, is valid).

Propositional logic is decidable because we can always construct the truth table for the formula.

Proof Procedures

Proof procedures for propositional logic are an alternate means to determine tautologies. As long as the proof procedure is sound, we can use the proof procedure in place of truth tables to determine tautologies.

A proof procedure is a set of rules we use to transform premises into conclusions:

- A goal is a formula that we want to prove is a tautology.
- A proof is a sequence of proof rules that when chained together relate the premise of the goal to the conclusion of the goal.

Forward and Backward Proof

In forward proof, we work from premises to conclusions. We apply rules that infer new formulas from premises. After many steps, the final inferred formulas should match the conclusion to have a proof.

In backward proof, we work from conclusions to premises. We use the proof rules backwards to reduce a conclusion to a formula closer to the premises. After many steps, the final reduced formulae should match the premise.

Proof Procedures for Propositional Logic

There are many proof procedures for propositional logic. Some match the human reasoning process. Others are better suited to automation by computers. Examples of proof procedures are:

- Resolution (Fitting, p. 51)
- Semantic Tableaux (Fitting, p. 42; Kelly, p. 27)
- Natural Deduction (Fitting, p. 86; Huth and Ryan, p. 6, Kelly, p. 42)
- Sequent Calculus (Fitting, p. 92; Kelly, p. 54)
- Hilbert Systems (axiomatic systems) (Fitting, p. 77; Kelly, p. 4)
- Davis-Putnam (Fitting, p. 98)

Proof Procedures for Propositional Logic

There are many proof procedures for propositional logic. Some match the human reasoning process. Others are better suited to automation by computers. Examples of proof procedures are:

- Resolution (Fitting, p. 51)
- Semantic Tableaux (Fitting, p. 42; Kelly, p. 27)
- Natural Deduction (Fitting, p. 86; Huth and Ryan, p. 6, Kelly, p. 42)
- Sequent Calculus (Fitting, p. 92; Kelly, p. 54)
- Hilbert Systems (axiomatic systems) (Fitting, p. 77; Kelly, p. 4)
- Davis-Putnam (Fitting, p. 98)

Hilbert Systems

Also called axiomatic systems or Frege systems. Axiomatic systems are forward reasoning.

They consist of

- a set of axioms
- a set of rules of inference (also called rules of derivation).

Presuming the truth of the axioms, apply the rules of inference to derive new propositions; continue until the desired formula is reached.

The following discussion is general for all Hilbert systems, not just those for propositional logic. (Fitting [Fit96] and Kelly [Kel97]).

Derivations

Definition. A derivation in a Hilbert system from a set S of formulas is a finite sequence $X_1, X_2, \ldots X_n$ of formulas such that each term is either an axiom, or is a member of S, or follows from earlier terms by one of the rules of inference.

We write:

$$\boxed{S \vdash_{\text{ph}} X}$$

to say that X has a derivation from S in the propositional Hilbert system.

Proofs

Definition. A proof in a Hilbert system is a finite sequence $X_1, X_2, \ldots X_n$ of formulas such that each term is either an axiom or follows from earlier terms by one of the rules of inference. A proof is a derivation from an empty set of formulas, i.e.,

$$\vdash_{\text{ph}} X$$

We will write proofs as a list of formulas, each on its own line, and refer to the line of a proof in the justification for steps.

Definition. X is a theorem of a Hilbert system if X is the last line of a proof. X is a consequence of a set S if X is the last line of a derivation from S.

Hilbert System for Propositional Logic

In a Hilbert System, every axiom is a tautology.

It's not very interesting (or useful) to take all the tautologies as axioms, rather we need a finite number of axioms, or at least a finite number of forms that axioms can take. We call these forms axiom schemes.

For example, $p \Rightarrow p$, $(p \wedge q) \Rightarrow (p \wedge q)$, and $\neg q \Rightarrow \neg q$ all have the form $X \Rightarrow X$.

We adopt the convention of using capital letters to represent formulae in axiom schemes.

An Axiomatic System for Prop. Logic

Src: Kelly [Kel97]

Three axiom (schemes):

1. $A \Rightarrow (B \Rightarrow A)$
2. $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$
3. $(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$

One rule of inference:

1. *modus ponens (MP)* - From A and $A \Rightarrow B$, we derive B, where A and B are any well-formed formulas.

Simple Example of a Proof

Show $\vdash_{\text{ph}} ((x \Rightarrow y) \Rightarrow (x \Rightarrow x))$:

1. $x \Rightarrow (y \Rightarrow x)$
   Ax1 where $A \equiv x, B \equiv y$
2. $(x \Rightarrow (y \Rightarrow x)) \Rightarrow ((x \Rightarrow y) \Rightarrow (x \Rightarrow x))$
   Ax2 where $A \equiv x, B \equiv y, C \equiv x$
3. $(x \Rightarrow y) \Rightarrow (x \Rightarrow x)$
   MP on lines 1 and 2

Example

Rather than constructing particular proofs, we can actually construct “meta-theorems” (theorem schemes).

Example: Show $\vdash_{\text{ph}} X \Rightarrow X$

Examples to Try

Show the following:

1. $\vdash_{\text{ph}} \neg X \Rightarrow (X \Rightarrow Y)$
2. $\{X \Rightarrow Y, Y \Rightarrow Z\} \vdash_{\text{ph}} X \Rightarrow Z$
3. $\vdash_{\text{ph}} (Y \Rightarrow Z) \Rightarrow ((X \Rightarrow Y) \Rightarrow (X \Rightarrow Z))$

Note: You can reuse previous results in these proofs.

Deduction Theorem

Theorem. In any Hilbert System with at least Axiom Schemes 1 and 2, and with *modus ponens* as the only rule of inference, $S \cup \{X\} \vdash_{\text{ph}} Y$ iff $S \vdash_{\text{ph}} (X \Rightarrow Y)$.

This result was proven by both Tarski and Herbrand.

Use of the Deduction Theorem

Show $\{A \Rightarrow B\} \vdash_{ph} A \Rightarrow (C \Rightarrow B)$:

Set out to show: $A \Rightarrow B, A \vdash_{ph} C \Rightarrow B$:

1	A	premise
2	$A \Rightarrow B$	premise
3	B	MP on 1 and 2
4	$B \Rightarrow (C \Rightarrow B)$	Ax1
5	$C \Rightarrow B$	MP on 3 and 4

Now that we've proven $\{A \Rightarrow B, A\} \vdash_{ph} C \Rightarrow B$, using the deduction theorem we can conclude: $\{A \Rightarrow B\} \vdash_{ph} A \Rightarrow (C \Rightarrow B)$.

Sound. and Complete. of Axiom. Logic

Soundness: Every theorem A in Axiomatic Logic is a tautology:

$$\vdash_{ph} A \quad \Rightarrow \quad \models A$$

Completeness: If A is a tautology then it is a theorem of Axiomatic Logic:

$$\models A \quad \Rightarrow \quad \vdash_{ph} A$$

Axiomatic Logic is consistent.

An Aside on Monotonicity

Definition. A monotonic logic is one where a valid proof cannot be invalidated by the addition of extra premises.

We will only be studying monotonic logics.

Non-monotonic logics are often useful for reasoning about knowledge.

Propositional Logic

1. syntax (well-formed formulas)
2. semantics (truth tables)
3. proof theory
   - axiom systems
   - natural deduction
   - sequent calculus

Natural Deduction

Gerhard Gentzen (1909–1945). Natural deduction was introduced in his paper *Investigations into Logical Deduction*, 1935.

Src: http://www-groups.dcs.st-andrews.ac.uk/ history/Mathematicians/Gentzen.html

Natural Deduction

$$\boxed{p_1, p_2, p_3, \ldots \vdash_{\text{ND}} q}$$

The notation above means that there is a proof using natural deduction that the argument with premises $p_1, p_2, p_3, \ldots$ and conclusion q is valid.

Logical formulas ψ such that $\vdash_{\text{ND}} \psi$ are called tautologies.

Again, there are multiple natural deduction systems for propositional logic. We will (mostly) be following the presentation of Huth and Ryan [HR00].

Natural Deduction

- Forward proof with hierarchical boxed structure
- No axioms
- Inference rules
  - Eliminate or introduce logical connective ($\wedge, \vee, \neg\neg$)
  - Modus ponens, modus tollen (implication elimination)
  - Conditional proof (implication introduction)
  - Proof by contradiction

Inference Rules

Definition. An inference rule is a primitive valid argument form. Each inference rule enables the elimination or the introduction of a logical connective.

Most inference rules have names that consists of:

1. a logical connective,
2. a letter:
   - “i” indicates that the rule introduces the connective
   - “e” indicates that the rule eliminates the connective

Examples: $\wedge$i, $\Rightarrow$e

Rules for Conjunction

and-introduction

$$\frac{\begin{array}{c}p\\ q\end{array}}{p \wedge q}\ \wedge\text{i}$$

and-elimination

$$\frac{p \wedge q}{p}\ \wedge\text{e}$$

$$\frac{p \wedge q}{q}\ \wedge\text{e}$$

Above the line are the premises of the rule. Below the line is the conclusion. To the right of the line is the name of the rule.

p and q may be larger formulas than proposition letters.

Example

Show $p \wedge q, r \vdash_{\text{ND}} q \wedge r$

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

We present proofs in the linear format, but a tree format could be used.

Rules for Eliminating Implication

Implies-elimination (*modus ponens*)

$$\frac{\begin{array}{c}p\\ p \Rightarrow q\end{array}}{q}\ \Rightarrow\text{e}$$

A related rule is *modus tollens:*

$$\frac{\begin{array}{c}p \Rightarrow q\\ \neg q\end{array}}{\neg p}\ \text{MT}$$

Example: If it is raining, then I have my umbrella up. I do not have my umbrella up. Therefore it is not raining.

Subordinate Proofs

More complicated proofs require intermediate subordinate proofs. We make assumptions, and then discharge the assumptions.

Subordinate proofs are indented/boxed with the first line in the box being the assumption made in that subordinate proof. The first line below the indentation/box is the result of discharging the assumption.

Implies Introduction

$$\dfrac{\left[\begin{array}{l} r \quad \text{assumption} \\ \vdots \\ q \end{array}\right.}{r \implies q}\ \Rightarrow\text{i}$$

Within a new box, we assume r and prove q. The box marks the scope of the temporary assumption. Lines in the box depend on the assumption. The line after the box introduces an implication, whose premise is the temporary assumption and whose conclusion is the last derived proposition in the box.

- The resulting implication does not depend on the assumption.
- Boxes may be nested.
- At any stage in the proof, the active formulas are those occurring in boxes that haven't yet been closed. We can use only active formulas to derive new formulas.

Rules for Disjunction

or-introduction

$$\dfrac{p}{p \vee q}\ \vee\text{i} \qquad \dfrac{q}{p \vee q}\ \vee\text{i}$$

or-elimination
(case analysis)

$$\dfrac{p \vee r \quad \boxed{\begin{array}{c} p \\ \vdots \\ q \end{array}} \quad \boxed{\begin{array}{c} r \\ \vdots \\ q \end{array}}}{q}\ \vee\text{e}$$

Rules for Negation

false-elimination

$$\dfrac{\textbf{false}}{q}\ \textbf{false}\text{e}$$

not-elimination

$$\dfrac{\begin{array}{c} p \\ \neg p \end{array}}{\textbf{false}}\ \neg\text{e} \qquad \dfrac{\boxed{\begin{array}{c} r \\ \vdots \\ \textbf{false} \end{array}}}{\neg r}\ \neg\text{i}$$

From a contradiction, we can prove anything.

$$\dfrac{\begin{array}{c} p \\ \neg p \end{array}}{q}\ \neg\text{e}$$

Derived Rule: Proof by Contradiction

Also called RAA (reduction to absurdity).

$$\dfrac{\left[\begin{array}{l} \neg r \quad \text{assumption} \\ \vdots \\ \textbf{false} \end{array}\right.}{r}\ \text{RAA}$$

1	$\neg r \Rightarrow$ **false**	premise
2	$\neg r$	assumption
3	**false**	$\Rightarrow$e $1, 2$
4	$\neg\neg r$	$\neg$i $2 - 3$
5	r	$\neg\neg$e 4

Examples to Try

Show:

1. $p \wedge q \Rightarrow r \vdash_{\text{ND}} p \Rightarrow (q \Rightarrow r)$
2. $p \wedge (q \Rightarrow r) \vdash_{\text{ND}} p \wedge q \Rightarrow r$
3. $p \Rightarrow q \vdash_{\text{ND}} (p \wedge r) \vdash_{\text{ND}} (q \wedge r)$

Summary of Natural Deduction

Natural deduction for propositional logic is sound and complete.

A summary of the rules can be found on an additional handout.

What to Remember

Propositional Logic

- syntax (well-formed formulae)
- semantics (truth tables)

Axiomatic proofs

- 3 axioms
- Modus Ponens as only inference rule
- sound and complete

Natural deduction

- no axioms
- inference rules eliminate or introduce logical operator
- sound and complete

Summary

- What is logic? (completeness, soundness)
- Propositional Logic
  - syntax
  - semantics
  - proof procedures
    - axiom system
    - natural deduction
    - ~~next class: sequent calculus~~

References

[Fit96] Melvin Fitting. First-Order Logic and Automated Theorem Proving. Springer, 1996. Second Edition.

[FrO] Free on-line dictionary of computing. http://foldoc.doc.ic.ac.uk/foldoc/index.html.

[HR00] Michael R. A. Huth and Mark D. Ryan. Logic in Computer Science. Cambridge University Press, Cambridge, 2000. DC Library: QA76.9.L63 H88 2000.

[Kel97] John Kelly. The Essence of Logic. Prentice Hall, 1997.

[Sub02] Peter Suber. Philosophy 340, Earlham College, 2002. http://www.earlham.edu/ peters/courses/logsys/math-ind.htm.