

2. Propositional Logic

- Last time:
 - Introduction
 - Testing vs. Formal Specification
 - Examples (Petri Nets, CCS, Alloy, ...)

- Today:
 - Propositional Logic

- The following slides are based on the slides from M. Lawford (<http://www.cas.mcmaster.ca/~lawford/>) and J. Bowen (<http://www.cs.ucl.ac.uk/staff/J.Bowen/GS03/>)

What is Propositional Logic?

Def: A *proposition* is a statement that is either true or false.

E.g. p : “The prof looks tired.”

q : “We’re hungry and not able to eat.”

Propositional logic is a formal mathematical system for reasoning about such statements.

The first statement p is an *atomic proposition*. It cannot be further subdivided.

The 2nd statement q is a *compound proposition* that’s truth depends upon the value of the two atomic propositions:

1. h : “We are hungry.” and 2. e : “We are able eat.”

The *logical connectives* “and” and “not” determine how the atomic proposition affect q .

Restating q in the formal language of propositional logic:

$$q : h \wedge \neg e$$

Logical Connectives

Let T and F represent *true* and *false* respectively.

Define $\mathcal{V} := \{T, F\}$, the set of possible truth values for a proposition. In the following let p, q be propositional variables.

Negation: \neg (NOT)

$\neg : \mathcal{V} \rightarrow \mathcal{V}$

| | |
|-----|----------|
| p | $\neg p$ |
| F | T |
| T | F |

A *truth table* is tabular representation of the truth values of a proposition under all possible assignments. The above is the table for $\neg p$. Clearly it defines a function.

Truth tables define the meaning or interpretation propositions. We call this the *semantics* of the propositional logic.

Computing with truth tables

- Truth tables for logical connectives**

| ϕ | ψ | $\phi \wedge \psi$ | ϕ | ψ | $\phi \vee \psi$ | ϕ | ψ | $\phi \Rightarrow \psi$ | ϕ | $\neg \phi$ |
|--------|--------|--------------------|--------|--------|------------------|--------|--------|-------------------------|--------|-------------|
| T | T | T | T | T | T | T | T | T | T | F |
| T | F | F | T | F | T | T | F | F | F | T |
| F | T | F | F | T | T | F | T | T | F | T |
| F | F | F | F | F | F | F | F | T | F | T |

- Truth tables can be composed**

| ϕ | ψ | $\phi \vee \psi$ | $\neg(\phi \vee \psi)$ |
|--------|--------|------------------|------------------------|
| T | T | T | F |
| T | F | T | F |
| F | T | T | F |
| F | F | F | T |

Example for implication

| F | G | $F \Rightarrow G$ |
|-------|-------|-------------------|
| TRUE | TRUE | TRUE |
| TRUE | FALSE | FALSE |
| FALSE | TRUE | TRUE |
| FALSE | FALSE | TRUE |

 ?

- We expect that if n is **greater** than **3**, then it should be **greater** than **1**, so $n > 3$ should imply $n > 1$.
- Therefore, the formula $(n > 3) \Rightarrow (n > 1)$ should equal true.
- Substituting **4**, **2**, and **0** for n in this formula explains why $F \Rightarrow G$ means F implies G or, equivalently, if F then G .

Truth Tables

In general, a truth table for compound proposition will have 2^n rows, where n = number of unique propositional variables occurring in the expression.

Count in binary with F being 0 and T being 1 to cover all cases.

Precedence of Logical Connectives

We say that operators with a higher order of precedence “have a tighter binding”.

Similarly for logical connectives we define the order of precedence as:

Do 1st <-----> Do last
 \wedge \rightarrow
 \neg , \vee , \leftrightarrow

Thus $((p \wedge \neg(q)) \rightarrow r)$ becomes: $p \wedge \neg q \rightarrow r$

Tautologies and Contradictions

Def: A logical expression is a *tautology* (*contradiction*) if it is true (false) under all possible assignments to its propositional variables.

E.g. $p \vee \neg p$ is a tautology since its truth table results in all T 's while $p \wedge \neg p$ is a contradiction:

| p | $\neg p$ | $p \vee \neg p$ | $p \wedge \neg p$ |
|-----|----------|-----------------|-------------------|
| F | T | T | F |
| T | F | T | F |

The negation of any tautology is a contradiction and vice versa.
Why?

If S is a tautology, then so is any substitution instance of it (i.e. consistently replacing variables with any other formulas results in a tautology!).

E.g $(p \rightarrow q) \vee \neg(p \rightarrow q)$ is a tautology.

Logical (Semantic) Equivalence

Def: Two propositional formulas are *logically equivalent* if they have the same truth table.

This means the propositions define the same function from \mathcal{V}^n to \mathcal{V} where $n :=$ number of propositional variables in the formulas.

E.g. The formulas $\neg(p \wedge q)$ and $\neg p \vee \neg q$ define the same function $f : \mathcal{V}^2 \rightarrow \mathcal{V}$

| p | q | $\neg(p \wedge q)$ | $\neg p \vee \neg q$ |
|-----|-----|--------------------|----------------------|
| F | F | T | T |
| F | T | T | T |
| T | F | T | T |
| T | T | F | F |

Logical (Semantic) Equivalence

Note that $\neg(p \wedge q)$ and $\neg p \vee \neg q$ are logically equivalent iff $\neg(p \wedge q) \leftrightarrow \neg p \vee \neg q$ is a tautology. Why?

| p | q | $\neg(p \wedge q)$ | $\neg p \vee \neg q$ | $\neg(p \wedge q) \leftrightarrow \neg p \vee \neg q$ |
|-----|-----|--------------------|----------------------|---|
| F | F | T | T | |
| F | T | T | T | |
| T | F | T | T | |
| T | T | F | F | |

This is why Rubin refers to logical equivalence as *tautological equivalence* and when ϕ is logically equivalent to ψ writes: $\phi \Leftrightarrow \psi$.

Huth+Ryan refer logical equivalence as *semantic equivalence* and write: $\phi \equiv \psi$. It all means the same thing. The formulas have the same truth table.

Normal Forms

Normal forms in mathematics are canonical representations (i.e. all equivalent objects result in the same representation).

Def: A formula ϕ with p_1, p_2, \dots, p_n propositional variables is in *Disjunctive Normal Form* (DNF) if it has the structure:

$$(x_1^1 \wedge x_2^1 \wedge \dots \wedge x_n^1) \vee \dots \vee (x_1^m \wedge x_2^m \wedge \dots \wedge x_n^m)$$

where $m \leq 2^n$ and for $i = 1, \dots, n$ and $j = 1, \dots, m$, x_i^j is either p_i or $\neg p_i$

E.g. $(\neg p \wedge \neg q \wedge r) \vee (p \wedge \neg q \wedge \neg r)$ is in DNF

$\neg(p \vee q) \wedge r$ is not. Each of the series of conjunctions picks out a row of the truth table where formula is true. DNF ORs together the ANDs for the true rows.

Normal Forms

Consider the truth tables for the formulas

$\neg p \wedge \neg q \wedge r$ and $p \wedge \neg q \wedge \neg r$:

| | p | q | r | $\neg p \wedge \neg q \wedge r$ | $p \wedge \neg q \wedge \neg r$ |
|---|-----|-----|-----|---------------------------------|---------------------------------|
| 0 | F | F | F | F | F |
| 1 | F | F | T | T | F |
| 2 | F | T | F | F | F |
| 3 | F | T | T | F | T |
| 4 | T | F | F | F | F |
| 5 | T | F | T | F | F |
| 6 | T | T | F | F | F |
| 7 | T | T | T | F | F |

For $\neg p \wedge \neg q \wedge r$ only row 1 is true.

Normal Forms

Theorem: Every propositional formula that is not a contradiction is a logically equivalent to a DNF formula.

Corollary: For ϕ, ψ not contradictions, $\phi \equiv \psi$ iff ϕ and ψ have the same DNF representation.

Proof: Two formulas are logically equivalent if and only if they have the same truth table (i.e. same true rows) & thus the same DNF.

Application: Minimizing gate delays

If each input & its negation are available, any logic function can be implemented with one “stage” of multi-input AND gates followed by one “stage” of multi-input OR gates.

Logical Implication, Entailment

Def: We say ϕ *logically implies* ψ if $\phi \rightarrow \psi$ is a tautology. In this case Rubin writes $\phi \Rightarrow \psi$. If ϕ is a conjunction (i.e. ϕ is $\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n$) then we say $\phi_1, \phi_2, \dots, \phi_n$ logically imply ψ .

Huth+Ryan write $\models \phi \rightarrow \psi$ or $\phi \models \psi$.

Premises ϕ_1, \dots, ϕ_n with conclusion ψ is a *sound* or *valid argument*, denoted

$$\phi_1, \phi_2, \dots, \phi_n \models \psi$$

if whenever all the ϕ_i s are true, then ψ is true.

Theorem: $\models \phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n \rightarrow \psi$ if and only if $\phi_1, \phi_2, \dots, \phi_n \models \psi$.

Modus Ponens

Modus Ponens: $p, p \rightarrow q \models q$

| p | q | $p \rightarrow q$ | $p \wedge (p \rightarrow q)$ | q |
|-----|-----|-------------------|------------------------------|-----|
| F | F | T | F | F |
| F | T | T | F | T |
| T | F | F | F | F |
| T | T | T | T | T |

Checking validity (soundness)

- To prove an argument is valid we only have to check that the conclusion (ψ) is true in rows in which all the premises (ϕ_i 's) are true.
- To prove an argument is *invalid* (unsound), we need only find one counter example, a row in which each ϕ_i is true but ψ is false.

Examples: 1. $(p \rightarrow q) \rightarrow r \models p \rightarrow (q \rightarrow r)$ but
 $p \rightarrow (q \rightarrow r) \not\models (p \rightarrow q) \rightarrow r$

Example

$$2. p, p \rightarrow q, \neg r \rightarrow \neg q \models r$$

□ See board

Solution

| | p | q | r | $p \rightarrow q$ | $\neg r \rightarrow \neg q$ | r |
|---|-----|-----|-----|-------------------|-----------------------------|-----|
| 0 | F | F | F | | | |
| 1 | F | F | T | | | |
| 2 | F | T | F | | | |
| 3 | F | T | T | | | |
| 4 | T | F | F | F | | |
| 5 | T | F | T | F | | |
| 6 | T | T | F | T | F | |
| 7 | T | T | T | T | T | T |

Special Cases

1. **No premises:** Premises restrict the cases that we have to consider. No premises means we consider all cases. ψ is a valid argument by itself if it is always true (i.e. it is a tautology). Then we write $\models \psi$ and say that ψ is *valid*.
2. **Premises never all true:** At least one ϕ_i is always false so $\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n$ is a contradiction. Then $\phi_1, \dots, \phi_n \models \psi$.

Contradiction

“If pigs could fly then I’d enjoy brussel sprouts!”

p : Pigs fly; b : Enjoy sprouts

This $(p \models b)$ is an *invalid argument*. Why use it?

The real argument is:

$$p, \neg p \models b$$

which *is* a valid argument.

Why is it valid? There is no counter example where $p \wedge \neg p$ is true and b is false.

Ex falso quod libet! i.e. “From false all things are possible!”

$\neg p$ is an *implicit assumption* in the verbal argument. Implicit assumptions are extremely dangerous in software. Make your assumptions explicit!

How do you make software assumptions explicit? Documentation, using strongly typed languages, dependent typing in PVS, *etc.* ...

Validity & Satisfiability

Let ϕ be some formula of propositional logic. In the case that $\models \phi$, we say that ϕ is *valid*.

In the case that ϕ is **not** valid (i.e., there is some assignment to its variables that makes it false) we will write $\not\models \phi$.

If there is some assignment to the propositional variables that makes ϕ true (i.e., there is one or more T in the final column of ϕ 's truth table), then we say that ϕ is *satisfiable*.

Proposition: ϕ is satisfiable iff $\not\models \neg\phi$.

Conjunctive Normalform

Def: A formula with p_1, p_2, \dots, p_n propositional variables is in *Conjunctive Normal Form* (CNF) if it has the structure:

$$(x_1^1 \vee x_2^1 \vee \dots \vee x_n^1) \wedge \dots \wedge (x_1^m \vee x_2^m \vee \dots \vee x_n^m)$$

where $m \leq 2^n$ and for $i = 1, \dots, n$ and

$j = 1, \dots, m$, x_i^j is either p_i or $\neg p_i$

E.g. $(\neg p \vee \neg q \vee r) \wedge (p \vee \neg q \vee \neg r)$ is in CNF

$\neg(p \wedge q) \vee r$ is not. Each of the series of disjunctions rules out a row of the truth table where formula is false. CNF ANDs together the ORs for the false rows.

One way to obtain the CNF form of a formula ϕ is to write down the DNF for $\neg\phi$ and then negate it and “Demorgan it to death”.

Using CNF to Check $\models \Phi$

Q: CNF seems a little harder to understand than DNF, so why use it?

A: Because it is trivial to check $\models \phi$ if ϕ is in CNF.

Why? Because

$$\models (x_1^1 \vee x_2^1 \vee \dots \vee x_n^1) \wedge (x_1^2 \vee x_2^2 \vee \dots \vee x_n^2) \\ \dots \wedge (x_1^m \vee x_2^m \vee \dots \vee x_n^m)$$

...

if and only if

$$\models (x_1^1 \vee x_2^1 \vee \dots \vee x_n^1)$$

and

$$\models (x_1^2 \vee x_2^2 \vee \dots \vee x_n^2)$$

\vdots

and

$$\models (x_1^m \vee x_2^m \vee \dots \vee x_n^m)$$

If each x_i^j is a *literal* (e.g., p) or its negation (e.g., $\neg p$) then

$\models (x_1^j \vee x_2^j \vee \dots \vee x_n^j)$ iff there exists k, l s.t. $x_k^j = p$ and $x_l^j = \neg p$.

Language of Propositional Calculus

Def: A *propositional formula* is constructed inductively from the symbols for

- propositional variables: p, q, r, \dots or p_1, p_2, \dots
- connectives: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$
- parentheses: $(,)$
- constants: \top, \perp

by the following rules:

1. A propositional variable or constant symbol (\top, \perp) is a formula.
2. If ϕ and ψ are formulas, then so are:

$$(\neg\phi), (\phi \wedge \psi), (\phi \vee \psi), (\phi \rightarrow \psi), (\phi \leftrightarrow \psi)$$

Note: Removing \top, \perp from the above provides the def. of *sentential formulas* in Rubin. Semantically $\top = T$,

BNF and Parse Trees

The Backus Naur form (BNF) for the definition of a propositional formula is:

$$\phi ::= p | \perp | \top | (\neg\phi) | (\phi \wedge \phi) | (\phi \vee \phi) | (\phi \rightarrow \phi)$$

Here p denotes any propositional variable and each occurrence of ϕ to the right of $::=$ represents any formula constructed thus far.

We can apply this inductive definition in reverse to construct a formula's *parse tree*. A parse tree represents a WFF ϕ if

- i) the root is an atomic formula and nothing else (i.e. ϕ is p), or
- ii) the root is \neg and there is only one well formed subtree, or
- iii) the root is $\wedge, \vee, \rightarrow$ or \leftrightarrow and there are two well formed subtrees.

Note: All leaf nodes will be atomic (e.g. p, \perp or \top)

Subformulas and Subtrees

Def: If ϕ is a propositional formula, the *subformulas* of ϕ are given as follows:

- ϕ is a subformula of ϕ ,
- if ϕ is $\neg\psi$ then ψ is a subformula of ϕ ,
- if ϕ is $(\phi_1 \wedge \phi_2)$, $(\phi_1 \vee \phi_2)$, $(\phi_1 \rightarrow \phi_2)$, or $(\phi_1 \leftrightarrow \phi_2)$, then both ϕ_1 and ϕ_2 are subformulas of ϕ
- if ψ is a subformula of ϕ , then all subformulas of ψ are subformulas of ϕ .

The subformulas of ϕ correspond to all of the subtrees of ϕ 's parse tree.

Example: Consider $p \leftrightarrow (q \wedge \neg p \rightarrow q \vee r)$. The fully parenthesized formula is:

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

Motivating Proofs

Limitations of Truth Tables

of rows in truth table = 2^n where n = # of propositional variables in formula

Formula with 10 propositional variables has truth table with $2^{10} = 1024$ rows - too big to do by hand!

Safety critical shutdown system with 3 redundant controllers each with > 20 boolean inputs & majority vote logic on shutdown could have specification of a propositional formula using $> 3 \cdot 20 = 60$ boolean variables. Truth table would have $2^{60} > 1.15 \times 10^{18}$ rows!

Would require $\frac{2^{60}}{8 \times 10^9} = 144,115,188 > 144$ million GB/column!!

Motivating Proofs

Formal proof systems provide a way of examining the structure or *syntax* of formulas to determine the validity of an argument without resorting to truth tables.

E.g. Know $p, p \rightarrow q \models q$ (modus ponens - a.k.a \rightarrow_e). Therefore

$$c \vee \neg d, c \vee \neg d \rightarrow (a \wedge b \leftrightarrow c) \models a \wedge b \leftrightarrow c$$

Formal proof systems can decompose a problem into sub-problems (sub-proofs) that are of a manageable size.

E.g. If $\phi_1, \dots, \phi_n \models \psi_1$ and $\phi_1, \dots, \phi_n \models \psi_2$ then $\phi_1, \dots, \phi_n \models \psi_1 \wedge \psi_2$.

Proof rules and Proofs

These are examples of valid *rules of inference* or *proof rules*. E.g. Knowing that formulas ϕ and $\phi \rightarrow \psi$ are true allows us to *infer* or *deduce* that ψ is true.

An example of an invalid rule of inference would be knowing that ψ is true and $\phi \rightarrow \psi$ is true, we conclude ϕ . Why?

Def: A *proof of ψ from premises ϕ_1, \dots, ϕ_n* is a finite sequence of propositions ending with ψ , such that each member of the sequence is either a premise (ϕ_i), or is derived from previous members of the sequence by a valid rule of inference.

In this case we say that ϕ_1, \dots, ϕ_n *syntactically entails* ψ and write

$$\phi_1, \dots, \phi_n \vdash \psi$$

and say that $\phi_1, \dots, \phi_n \vdash \psi$ is a *valid sequent*.

Proof rules and Natural Deduction

While there exists more than one proof system for propositional (and predicate) logic *Natural Deduction* is one of the most useful systems.

It formalizes rules of mathematical proof you are already familiar with, e.g.,

To prove condition ϕ implies situation ψ , assume ϕ is true and show that ψ follows.

While there are variations in natural deduction proof systems (e.g., see Rubin vs. Huth+Ryan), they all have the same basic elements:

- Rules to eliminate operators, and
- Rules to introduce operators.

Proof rules: Implication Elimination

We know that $\phi, \phi \rightarrow \psi \models \psi$, so we can use this as a proof rule.

If $\phi_1, \dots, \phi_n \vdash \phi$ and $\phi_1, \dots, \phi_n \vdash \phi \rightarrow \psi$, then $\phi_1, \dots, \phi_n \vdash \psi$.

This rule is known as *modus ponens* or “implies (arrow) elimination” which we will abbreviate by $\rightarrow e$ and summarize as follows:

$$\frac{\phi \quad \phi \rightarrow \psi}{\psi} \rightarrow e$$

Proof rules: Implication Introduction

Another useful rule follow from the following useful fact:

$$\phi \models \psi \text{ iff } \models \phi \rightarrow \psi$$

Thus $\phi_1 \dots \phi_n, \phi \vdash \psi$ iff $\phi_1 \dots \phi_n \vdash \phi \rightarrow \psi$.

This rule is a form of the *Deduction Theorem*, a.k.a. conditional premise or “implies (arrow) introduction”, denoted by $\rightarrow i$, and summarized as follows:

$$\frac{\begin{array}{|c|} \hline \phi \\ \vdots \\ \psi \\ \hline \end{array}}{\phi \rightarrow \psi} \rightarrow i$$

A First Formal Proof

Example: Show that $p \rightarrow q, q \rightarrow r \vdash p \rightarrow r$

Proof:

| row | Premises | Deduce | Rule |
|-----|---------------------------------------|--------------------------|----------------------|
| 1 | $p \rightarrow q, q \rightarrow r$ | $\vdash p \rightarrow q$ | Premise |
| 2 | $p \rightarrow q, q \rightarrow r, p$ | $\vdash p$ | Assumption |
| 3 | $p \rightarrow q, q \rightarrow r, p$ | $\vdash q$ | $\rightarrow e1, 2$ |
| 4 | $p \rightarrow q, q \rightarrow r, p$ | $\vdash q \rightarrow r$ | Premise |
| 5 | $p \rightarrow q, q \rightarrow r, p$ | $\vdash r$ | $\rightarrow e3, 4$ |
| 6 | $p \rightarrow q, q \rightarrow r$ | $\vdash p \rightarrow r$ | $\rightarrow i2 - 5$ |

Note: We could show that:

$$p \rightarrow q, q \rightarrow r, r \rightarrow s \vdash p \rightarrow s$$

Remark

- In just 6 rows not $2^3=8$ rows of Truth Table.
- Try the second and check that the numbers now are 8 vs. $2^4=16$.

Proof rules: And Introduction

Its getting pretty tedious writing ϕ_1, \dots, ϕ_n and I'm basically lazy so henceforth I'll write Γ ("Big gamma"), to represent this sequence of premises.

Assume (1) $\Gamma \models \phi$ and (2) $\Gamma \models \psi$.

Then we must have $\Gamma \models \phi \wedge \psi$. (why?)

So if (1) $\Gamma \vdash \phi$ and (2) $\Gamma \vdash \psi$ then $\Gamma \vdash \phi \wedge \psi$.

This result is known as part of the Rules of Adjunction, a.k.a. "conjunction (and) introduction", denoted by $\wedge i$, and summarized as follows:

$$\frac{\phi \quad \psi}{\phi \wedge \psi} \wedge i$$

Proof rules: And Elimination

Consider the following pair of valid arguments:

$$(1) \phi \wedge \psi \models \phi \text{ and } (2) \phi \wedge \psi \models \psi$$

To paraphrase these arguments:

(1) says “When $\phi \wedge \psi$ is true, then ϕ is true.”

(2) says “When $\phi \wedge \psi$ is true, then ψ is true.”

So if $\Gamma \vdash \phi \wedge \psi$ then (1) $\Gamma \vdash \phi$ and (2) $\Gamma \vdash \psi$.

This and the previous result are known as the Rules of Adjunction.

We will call this part “conjunction (and) elimination”, denoted by

$\wedge e_1$ and $\wedge e_2$ respectively, and summarized as follows:

$$\frac{\phi \wedge \psi}{\phi} \wedge e_1 \qquad \frac{\phi \wedge \psi}{\psi} \wedge e_2$$

Proof rules: Negation Elimination/Intr.

It is easy to check that $\neg\neg\phi \equiv \phi$. Hence:

$$(1) \neg\neg\phi \models \phi \text{ and } (2) \phi \models \neg\neg\phi$$

So we know:

(1) if $\Gamma \vdash \neg\neg\phi$ then $\Gamma \vdash \phi$, and

(2) if $\Gamma \vdash \phi$ then $\Gamma \vdash \neg\neg\phi$.

These results are known as “double negation elimination” and

“double negation introduction”, denoted by $\neg\neg e$ and $\neg\neg i$

respectively, and summarized as follows:

$$\frac{\neg\neg\phi}{\phi} \neg\neg e \qquad \frac{\phi}{\neg\neg\phi} \neg\neg i$$

Proof rules: Modus Tollens

Recall the proof rule $\rightarrow e$.

$$\frac{\phi \quad \phi \rightarrow \psi}{\psi} \rightarrow e$$

If $\Gamma \vdash \phi$ and $\Gamma \vdash \phi \rightarrow \psi$, then $\Gamma \vdash \psi$.

Suppose it is still the case that $\Gamma \vdash \phi \rightarrow \psi$, but instead of $\Gamma \vdash \phi$ we know that $\Gamma \vdash \neg\psi$. If ϕ were true, then by $\rightarrow e$ we would have ψ , but since we have $\phi \rightarrow \psi$ and $\neg\psi$, we must have $\neg\phi$.

This reasoning is borne out by the valid argument:

$$\phi \rightarrow \psi, \neg\psi \models \neg\phi$$

The resulting proof rule is known as *modus tollens* or MT and

MT

summarized as:

$$\frac{\phi \rightarrow \psi \quad \neg\psi}{\neg\phi} \text{MT}$$

Example: Show $p \rightarrow q, q \rightarrow r \vdash \neg r \rightarrow \neg p$

Proof rules: Or Introduction

Consider the following pair of valid arguments:

$$(1) \phi \models \phi \vee \psi \text{ and } (2) \psi \models \phi \vee \psi$$

To paraphrase these arguments:

(1) says “When ϕ is true, then ϕ or ψ is true.”

(2) says “When ψ is true, then ϕ or ψ is true.”

So if (1) $\Gamma \vdash \phi$ or (2) $\Gamma \vdash \psi$ then, either way, $\Gamma \vdash \phi \vee \psi$.

We will call this “disjunction (OR) introduction” denoted $\vee i_1$ or $\vee i_2$ respectively and summarized as follows:

$$\frac{\phi}{\phi \vee \psi} \vee i_1 \qquad \frac{\psi}{\phi \vee \psi} \vee i_2$$

Proof rules: Or Elimination

Suppose we want to check if the following argument is valid

$$\phi \vee \psi \models \chi$$

“When ϕ is true or ψ is true, then χ is true”

Then there are two cases that we must consider:

- i) When ϕ is true, then χ is true ($\phi \models \chi$)
- ii) When ψ is true, then χ is true ($\psi \models \chi$)

This covers all possible case when $\phi \vee \psi$ is true, including the case when both are true.

Proof rules: Or Elimination

This rule is a form of the *Rule of Alternative Proof*, a.k.a. “disjunction (or) elimination” denoted \vee_e and summarized as follows:

$$\frac{\phi \vee \psi \quad \begin{array}{|c|} \hline \phi \\ \vdots \\ \chi \\ \hline \end{array} \quad \begin{array}{|c|} \hline \psi \\ \vdots \\ \chi \\ \hline \end{array}}{\chi} \vee_e$$

How can we use it? To show $\Gamma, \phi \vee \psi \vdash \chi$ split proof. Show:

1. $\Gamma, \phi \vdash \chi$
2. $\Gamma, \psi \vdash \chi$

...

Then we are done since $\Gamma, \phi \vee \psi \vdash \phi \vee \psi$. Thus

$$\Gamma, \phi \vee \psi \vdash \chi \text{ iff } \Gamma, \phi \vdash \chi \text{ and } \Gamma, \psi \vdash \chi$$

Proof Rules: $\neg e$

Consider the following semantic equivalence:

$$\phi \wedge \neg\phi \equiv \perp$$

which represents that \perp is itself a contradiction. If both ϕ and $\neg\phi$ are true, then we have an inconsistent set of premises. In fact when ever ϕ is both true and false (since $\neg\phi$ is true), the \perp is also valid.

Although one might think that the following rule should be called “bottom introduction”, to logicians perhaps this is too close to the idea of “introducing inconsistencies” which is a “Bad Thing[™]”, hence they call it “negation elimination” denoted $\neg e$ and summarized as:

$$\frac{\phi \quad \neg\phi}{\perp} \neg e$$

Proof Rules: $\perp e$

Consider the following valid argument:

$$\perp \models \phi$$

\perp is a valid argument for any formula! Why? Because you can't find a counter example in the truth table where \perp is true and ϕ is false.

Ex falso quod libet!

(From false all things are possible!)

This rule is called “bottom elimination”, denoted $\perp e$ and summarized as:

$$\frac{\perp}{\phi} \perp e$$

Thus if $\Gamma \vdash \perp$ then $\Gamma \vdash \phi$ for any ϕ .

Example

Example 1.20 We apply these rules to show that $\neg p \vee q \vdash p \rightarrow q$ is valid:

| | | | | | |
|---|-------------------|---------------------|-------------------|---------------------|--|
| 1 | $\neg p \vee q$ | | | | |
| 2 | $\neg p$ | premise | q | premise | |
| 3 | p | assumption | p | assumption | |
| 4 | \perp | $\neg e$ 3, 2 | q | copy 2 | |
| 5 | q | $\perp e$ 4 | $p \rightarrow q$ | $\rightarrow i$ 3-4 | |
| 6 | $p \rightarrow q$ | $\rightarrow i$ 3-5 | | | |
| 7 | $p \rightarrow q$ | | $p \rightarrow q$ | $\vee e$ 1, 2-6 | |

Proof Rules: $\neg i$

Let us assume that when we add ϕ to our premises Γ

$$\Gamma, \phi \models \perp$$

Then everywhere that all of the premises in Γ are true, ϕ must be false. (Why?) Therefore we must have:

$$\Gamma \models \neg\phi$$

In terms of syntactic entailment this will mean that if $\Gamma, \phi \vdash \perp$ then $\Gamma \vdash \neg\phi$.

This rule is often known as *indirect proof*, “negation (not)”

...

introduction”, denoted by $\neg i$, and summarized as follows:

$$\frac{\begin{array}{|c|} \hline \phi \\ \vdots \\ \perp \\ \hline \end{array}}{\neg\phi} \quad \neg i$$

Copy

This rule is a bit of a kludge, but it will save us extra effort when applied correctly.

Suppose a formula ψ appears on a previous line of our proof where we had a sequence of premises Γ , i.e., $\Gamma \vdash \psi$.

Further suppose that our current sequence of premises (formulas on the left of \vdash) is Γ' .

Q: Can we use ψ on the current line of our proof (i.e., does $\Gamma' \vdash \psi$)?

A: Yes, provided our current sequence of premises Γ' contains all of the formulas appearing in Γ .

E.g., $p \rightarrow q, q \rightarrow r \vdash p \rightarrow r$

Therefore $p \rightarrow q, q \rightarrow r, r \vdash p \rightarrow r$.

LEM

This next rule is based upon the valid argument:

$$\models \phi \vee \neg\phi$$

which say that a formula is either true, or it is false. There is no middle ground.

Since $\phi \vee \neg\phi$ is a tautology, for any set of premises $\Gamma \models \phi \vee \neg\phi$ (why?). Thus in using this rule in proofs we have $\Gamma \vdash \phi \vee \neg\phi$.

It is an example of a *derived rule*, i.e., we can prove it from the rules we have already seen ($\vdash p \vee \neg p$ - try it!).

The “Law of the Excluded Middle” denoted LEM is summarized as:

$$\frac{}{\phi \vee \neg\phi} \text{LEM}$$

Example: Show $\vdash \neg(p \wedge q) \leftrightarrow \neg p \vee \neg q$.

All Rules

| | <i>introduction</i> | <i>elimination</i> | | |
|---------------|---|---|----------------------------|---|
| \wedge | $\frac{\phi \quad \psi}{\phi \wedge \psi} \wedge i$ | $\frac{\phi \wedge \psi}{\phi} \wedge e_1 \quad \frac{\phi \wedge \psi}{\psi} \wedge e_2$ | | |
| \vee | $\frac{\phi}{\phi \vee \psi} \vee i_1 \quad \frac{\psi}{\phi \vee \psi} \vee i_2$ | $\frac{\phi \vee \psi \quad \begin{array}{ c } \chi \\ \chi \end{array}}{\chi} \vee e$ | Some useful derived rules: | |
| \rightarrow | $\frac{\begin{array}{ c } \phi \\ \psi \end{array}}{\phi \rightarrow \psi} \rightarrow i$ | $\frac{\phi \quad \phi \rightarrow \psi}{\psi} \rightarrow e$ | | $\frac{\phi \rightarrow \psi \quad \neg\psi}{\neg\phi} \text{MT} \quad \frac{\phi}{\neg\neg\phi} \neg\neg i$ |
| \neg | $\frac{\begin{array}{ c } \phi \\ \perp \end{array}}{\neg\phi} \neg i$ | $\frac{\phi \quad \neg\phi}{\perp} \neg e$ | | $\frac{\begin{array}{ c } \neg\phi \\ \perp \end{array}}{\phi} \text{PBC} \quad \frac{}{\phi \vee \neg\phi} \text{LEM}$ |
| \perp | (no introduction rule for \perp) | $\frac{\perp}{\phi} \perp e$ | | |
| $\neg\neg$ | | $\frac{\neg\neg\phi}{\phi} \neg\neg e$ | | |

Soundness

Def: A proof system is *sound* (or consistent) if whenever $\Gamma \vdash \psi$, then $\Gamma \models \psi$.

Our system of proof rules given above is sound. We can show this via induction on the length of our proofs. This is an immediate result of the fact that all of our proof rules are based upon valid arguments.

Stating this more formally:

Theorem (Soundness): Let $\phi_1, \phi_2, \dots, \phi_n$ and ψ be a propositional formulas. If $\phi_1, \phi_2, \dots, \phi_n \vdash \psi$ then $\phi_1, \phi_2, \dots, \phi_n \models \psi$.

If we let Γ represent ϕ_1, \dots, ϕ_n then the above says: “If $\Gamma \vdash \psi$ then $\Gamma \models \psi$.”

Completeness

Def: A proof system is *complete* if whenever $\Gamma \models \psi$, then $\Gamma \vdash \psi$.

Our system of proof rules given thus far is complete. We can show this via induction on the height of the parse tree of ψ .

Stating this formally:

Theorem (Completeness): Let $\phi_1, \phi_2, \dots, \phi_n$ and ψ be a propositional formulas. If $\phi_1, \phi_2, \dots, \phi_n \models \psi$ then $\phi_1, \phi_2, \dots, \phi_n \vdash \psi$.

If we let Γ represent ϕ_1, \dots, ϕ_n then the above result together with the previous one gives us the following corollary:

Corollary: $\Gamma \vdash \psi$ iff $\Gamma \models \psi$.

Propositional logic is decidable

- **There is an effective procedure for deciding whether any formula φ is valid**
 1. **convert into conjunctive normal form**
 $(p \vee q \vee \dots) \wedge (r \vee \neg s \vee \dots) \wedge \dots$
 2. **iff each disjunct is valid, the whole formula is valid**
 3. **iff each disjunct contains at least one instance of p and $\neg p$, it is valid**
- **φ is satisfiable iff $\neg\varphi$ is not valid**
- **There are efficient methods of computing satisfiability for any formula φ - SAT solvers**

Summary

- Propositional logic is the simplest formal logic
- It has a sound and complete proof system
- It is decidable and there are efficient procedures for establishing
 - whether a formula is valid
 - whether a formula is satisfiable
- So it is completely mechanizable
 - proofs do not require any human intervention

Conclusion

- This lecture: Propositional logic
 - Natural deduction (proof)
 - Truth tables (model)
- Read Chapter 1 of Huth and Ryan (especially 1.1-1.4)
- Do selected (starred) exercises with online solutions, depending on your previous experience. E.g.:
 - 1.1, 1(a), 2(a); 1.2 1(c), 2(a), 3(d)
 - 1.3 1(d), 2(a); 1.4 1, 2(c); 1.5 2 (b, d)
 - If you know the correct answers do less
 - If you have difficulty do more
- Next lecture: Predicate logic