02—Propositional Logic II

CS 5209: Foundation in Logic and AI

Martin Henz

January 21, 2010
1 Natural Deduction

2 Propositional Logic as a Formal Language

3 Semantics of Propositional Logic

4 Soundness and Completeness

5 Conjunctive Normal Form
1. Natural Deduction
   - Recap: Propositions
   - Recap: Rules for Conjunction and Disjunction
   - Recap: Rules for Double Negation and Implication
   - Recap: Rules for Disjunction
   - Rules for Negation
   - Basic and Derived Rules

2. Propositional Logic as a Formal Language

3. Semantics of Propositional Logic

4. Soundness and Completeness
Propositions

Propositions are Declarative Sentences
Sentences which one can—in principle—argue as being true or false.

Examples

1. The sum of the numbers 3 and 5 equals 8.
2. Jane reacted violently to Jack’s accusations.
3. Every natural number > 2 is the sum of two prime numbers.
Sequents as Logical Arguments

A Sequent in English

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.*
Focus on Structure, not Content

We are primarily concerned about the structure of arguments in this class, not the validity of statements in a particular domain.

We therefore simply abbreviate sentences by letters such as $p$, $q$, $r$, $p_1$, $p_2$ etc.

Instead of English words such as “if...then”, “and”, “not”, it is more convenient to use symbols such as $\rightarrow$, $\land$, $\neg$. 
Sequents in Symbolic Notation

A Sequent in English

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.

The same sequent using letters and symbols

\[ p \land \neg q \rightarrow r, \neg r, p \vdash q \]

Remaining task

Develop proof rules that allows us to derive such sequents
Rules for Conjunction

Introduction of Conjunction

\[ \begin{array}{c}
\phi \\
\psi \\
\hline \\
\phi \land \psi \\
\end{array} \]

 Elimination of Conjunction

\[ \begin{array}{c}
\phi \land \psi \\
\hline \\
\phi \\
\hline \\
[\land e_1] \\
\phi \\
\hline \\
[\land e_2] \\
\psi \\
\end{array} \]
### Rules of Double Negation

#### Introduction of Double Negation

\[
\phi \\
\hline
\neg \neg \phi \\
\]

#### Elimination of Double Negation

\[
\neg \neg \phi \\
\hline
\phi \\
\]

\[
\neg \neg \\
\phi \\
\hline
\phi \\
\]

\[
\phi \\
\hline
\neg \neg \phi \\
\]

\[
\neg \neg \phi \\
\hline
\phi \\
\]

\[
\neg \neg \phi \\
\hline
\phi \\
\]

\[
\neg \neg \phi \\
\hline
\phi \\
\]
Rules for Eliminating Implication

\[
\begin{align*}
\phi & \quad \phi \rightarrow \psi \\
\hline & \quad [\rightarrow e] \\
& \quad \psi \\
\phi \rightarrow \psi & \quad \neg \psi \\
\hline & \quad [MT] \\
& \quad \neg \phi
\end{align*}
\]
Rule for Introduction of Implication

\[ \phi \rightarrow \psi \]
Rules for Introduction of Disjunction

\[
\begin{align*}
\phi &
\quad [\lor i_1] \\
\phi \lor \psi &

\psi &
\quad [\lor i_2] \\
\phi \lor \psi &
\end{align*}
\]
Rule for Elimination of Disjunction

\[ \phi \vee \psi \]

\[ \vdots \]

\[ \chi \]

\[ \psi \]

\[ \vdots \]

\[ \chi \]

\[ \chi \]

\[ [\vee e] \]
Proof of \( p \land (q \lor r) \vdash (p \land q) \lor (p \land r) \)

<table>
<thead>
<tr>
<th>Step</th>
<th>Formula</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( p \land (q \lor r) )</td>
<td>premise</td>
</tr>
<tr>
<td>2</td>
<td>( p )</td>
<td>( \land e_1 \ 1 )</td>
</tr>
<tr>
<td>3</td>
<td>( q \lor r )</td>
<td>( \land e_2 \ 1 )</td>
</tr>
<tr>
<td>4</td>
<td>( q )</td>
<td>assumption</td>
</tr>
<tr>
<td>5</td>
<td>( p \land q )</td>
<td>( \land i \ 2,4 )</td>
</tr>
<tr>
<td>6</td>
<td>( (p \land q) \lor (p \land r) )</td>
<td>( \lor i_1 \ 5 )</td>
</tr>
<tr>
<td>7</td>
<td>( r )</td>
<td>assumption</td>
</tr>
<tr>
<td>8</td>
<td>( p \land r )</td>
<td>( \land i \ 2,7 )</td>
</tr>
<tr>
<td>9</td>
<td>( (p \land q) \lor (p \land r) )</td>
<td>( \lor i_2 \ 8 )</td>
</tr>
<tr>
<td>10</td>
<td>( (p \land q) \lor (p \land r) )</td>
<td>( \lor e \ 3, 4–6, 7–9 )</td>
</tr>
</tbody>
</table>
A Special Proposition

- Recall: We are only interested in the truth value of propositions, not the subject matter that they refer to (Martian pizzas or whatever).
- Therefore, all propositions that we all agree must be true are the same!
- Example: \( p \rightarrow p \)
- We denote the proposition that is always true using the symbol \( \top \).
Another Special Proposition

- We denote the proposition that is always true using the symbol \( \top \).
- Similarly, we denote the proposition that is always false using the symbol \( \bot \).
- Example: \( p \land \neg p \)
Elimination of Negation

\[
\phi \quad \neg \phi \\
\hline
[\neg e] \\
\bot
\]
Introduction of Negation

\[
\phi \\
\vdots \\
\bot \\
\hline
\neg i \\
\neg \phi
\]
Elimination of $\bot$

$\bot$  

$\bot$  

$\phi$  

$[\bot e]$
Basic Rules (conjunction and disjunction)

\[
\phi \quad \psi
\]

\[
\frac{}{\phi \land \psi}[\land i]
\]

\[
\phi \land \psi
\]

\[
\frac{}{\phi}[\land e_1]
\]

\[
\phi
\]

\[
\frac{}{\psi}[\land e_2]
\]

\[
\phi \lor \psi
\]

\[
\frac{}{\phi}[\lor i_1]
\]

\[
\phi \lor \psi
\]

\[
\frac{}{\psi}[\lor i_2]
\]

\[
\phi \lor \psi
\]

\[
\frac{\phi \lor \psi}{\chi}[\lor e]
\]

\[
\phi
\]

\[
\vdots
\]

\[
\chi
\]

\[
\psi
\]

\[
\vdots
\]

\[
\chi
\]
Basic Rules (implication)

\[
\begin{array}{c}
\phi \\
\vdots \\
\psi \\
\hline
\phi \rightarrow \psi
\end{array}
\]

\[
\phi \quad \phi \rightarrow \psi \\
\hline
\psi
\]

\[
\phi \rightarrow \psi
\]

\[
\phi \rightarrow \psi
\]

\[
\phi \rightarrow \psi
\]

\[
\phi \rightarrow \psi
\]
Basic Rules (negation)

\[
\begin{array}{c}
\phi \\
\vdots \\
\bot \\
\end{array}
\quad \left[\neg i\right] \\
\neg \phi
\]

\[
\begin{array}{c}
\phi \\
\neg \phi
\end{array}
\quad \left[\neg e\right] \\
\bot
Basic Rules ($\bot$ and double negation)

\[
\begin{align*}
\bot & \quad [\bot e] \\
\phi & \\
\neg\neg\phi & \quad [\neg\neg e] \\
\phi &
\end{align*}
\]
Some Derived Rules: Introduction of Double Negation

\[ \phi \]

\[ \frac{\phi}{\neg \neg \phi} \mid \neg \neg i \]
Example: Deriving \([\neg \neg i]\) from \([\neg i]\) and \([\neg e]\)

1. \(\phi\) premise
2. \(\neg \phi\) assumption
3. \(\bot\) \(\neg \neg \phi\) \(\neg e\) 1,2
4. \(\neg \neg \phi\) \(\neg i\) 2–3
Some Derived Rules: Modus Tollens

\[
\phi \rightarrow \psi \quad \neg \psi \\
\hline \\
\neg \phi
\]
Some Derived Rules: Proof By Contradiction

\[
\neg \phi \\
\vdots \\
\bot \\
\hline
\phi
\]

[PBC]
Some Derived Rules: Law of Excluded Middle

$$\phi \lor \neg \phi$$

[LEM]
1. Natural Deduction

2. Propositional Logic as a Formal Language

3. Semantics of Propositional Logic

4. Soundness and Completeness

5. Conjunctive Normal Form
Recap: Logical Connectives

\(\neg\): negation of \(p\) is denoted by \(\neg p\)

\(\lor\): disjunction of \(p\) and \(r\) is denoted by \(p \lor r\), meaning at least one of the two statements is true.

\(\land\): conjunction of \(p\) and \(r\) is denoted by \(p \land r\), meaning both are true.

\(\rightarrow\): implication between \(p\) and \(r\) is denoted by \(p \rightarrow r\), meaning that \(r\) is a logical consequence of \(p\).
Formal Description Required

Use of Meta-Language

When we describe rules such as \( \text{LEM} \)

\[
\phi \lor \neg \phi
\]

we mean that letters such as \( \phi \) can be replaced by \textit{any} formula.

But what exactly is the set of formulas that can be used for \( \phi \)?

Allowed

\[(p \land (\neg q))\]

Not allowed

\[] \land p \quad q\neg(\]

\(\text{CS 5209: Foundation in Logic and AI} \quad 02—\text{Propositional Logic II} \quad 31\)
Definition of Well-formed Formulas

Definition

- Every propositional atom $p, q, r, \ldots$ and $p_1, p_2, p_3, \ldots$ is a well-formed formula.
- If $\phi$ is a well-formed formula, then so is $(\neg \phi)$.
- If $\phi$ and $\psi$ are well-formed formulas, then so is $(\phi \land \psi)$.
- If $\phi$ and $\psi$ are well-formed formulas, then so is $(\phi \lor \psi)$.
- If $\phi$ and $\psi$ are well-formed formulas, then so is $(\phi \rightarrow \psi)$.
Definition very restrictive

How about this formula?

\[ p \land \neg q \lor r \]

Usually, this is understood to mean

\[ ((p \land (\neg q)) \lor r) \]

...but for the formal treatment of this section and the first homework, we insist on the strict definition, and exclude such formulas.
Backus Naur Form: A more compact definition

Backus Naur Form for propositional formulas

\[ \phi ::= p | (\neg \phi) | (\phi \land \phi) | (\phi \lor \phi) | (\phi \rightarrow \phi) \]

where \( p \) stands for any atomic proposition.
Inversion principle

How can we show that a formula such as

\[ (((\neg p) \land q) \to (p \land (q \lor (\neg r)))) \]

is well-formed?

Answer: We look for the only applicable rule in the definition (the last rule in this case), and proceed on the parts.
Parse trees

A formula

\[ (((\neg p) \land q) \rightarrow (p \land (q \lor (\neg r)))) \]

...and its parse tree:
Meaning of propositional formula

Meaning as mathematical object
We define the meaning of formulas as a function that maps formulas and valuations to truth values.

Approach
We define this mapping based on the structure of the formula, using the meaning of their logical connectives.
Truth values and valuations

Definition

The set of truth values contains two elements $T$ and $F$, where $T$ represents “true” and $F$ represents “false”.

Definition

A *valuation* or *model* of a formula $\phi$ is an assignment of each propositional atom in $\phi$ to a truth value.
### Meaning of logical connectives

The meaning of a connective is defined as a truth table that gives the truth value of a formula, whose root symbol is the connective, based on the truth values of its components.

<table>
<thead>
<tr>
<th>$\phi$</th>
<th>$\psi$</th>
<th>$\phi \land \psi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
Truth tables of formulas

Truth tables use placeholders of formulas such as $\phi$:

<table>
<thead>
<tr>
<th>$\phi$</th>
<th>$\psi$</th>
<th>$\phi \land \psi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

Build the truth table for given formula:

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$r$</th>
<th>$(p \land q)$</th>
<th>$((p \land q) \land r)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>

...
### Truth tables of other connectives

<table>
<thead>
<tr>
<th>$\phi$</th>
<th>$\psi$</th>
<th>$\phi \lor \psi$</th>
<th>$\phi$</th>
<th>$\psi$</th>
<th>$\phi \rightarrow \psi$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\phi$</th>
<th>$\neg \phi$</th>
<th>$\top$</th>
<th>$\bot$</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>
Constructing the truth table of a formula

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>(¬p)</th>
<th>¬q</th>
<th>p → ¬q</th>
<th>q ∨ ¬p</th>
<th>(p → ¬q) → (q ∨ ¬p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
Validity and Satisfiability

Validity
A formula is valid if it computes $\top$ for all its valuations.

Satisfiability
A formula is satisfiable if it computes $\top$ for at least one of its valuations.
Semantic Entailment

Definition

If, for all valuations in which all $\phi_1, \phi_2, \ldots, \phi_n$ evaluate to $\top$, the formula $\psi$ evaluates to $\top$ as well, we say that $\phi_1, \phi_2, \ldots, \phi_n$ semantically entail $\psi$, written:

$$\phi_1, \phi_2, \ldots, \phi_n \models \psi$$
Soundness of Propositional Logic

Soundness

Let $\phi_1, \phi_2, \ldots, \phi_n$ and $\psi$ be propositional formulas. If $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$, then $\phi_1, \phi_2, \ldots, \phi_n \models \psi$. 
Completeness

Let $\phi_1, \phi_2, \ldots, \phi_n$ and $\psi$ be propositional formulas. If $\phi_1, \phi_2, \ldots, \phi_n \models \psi$, then $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$. 
Natural Deduction

Propositional Logic as a Formal Language

Semantics of Propositional Logic

Soundness and Completeness

Conjunctive Normal Form
Provability

Definition
If there is a natural deduction proof of $\psi$ using the premises $\phi_1, \phi_2, \ldots, \phi_n$, we say that $\psi$ is provable from $\phi_1, \phi_2, \ldots, \phi_n$ and write

$$\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$$
Semantic Entailment

Definition
If, for all valuations in which all $\phi_1, \phi_2, \ldots, \phi_n$ evaluate to $\top$, the formula $\psi$ evaluates to $\top$ as well, we say that $\phi_1, \phi_2, \ldots, \phi_n$ semantically entail $\psi$, written:

$$\phi_1, \phi_2, \ldots, \phi_n \models \psi$$
Some More Definitions

Semantic equivalence
Let $\phi$ and $\psi$ be formulas of propositional logic. We say that $\phi$ and $\psi$ are semantically equivalent iff $\phi \models \psi$ and $\psi \models \phi$ hold. We write $\psi \equiv \phi$.

Validity
If $\models \phi$ holds, we call $\phi$ valid.
Soundness of Natural Deduction

Soundness

Let $\phi_1, \phi_2, \ldots, \phi_n$ and $\psi$ be propositional formulas. If $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$, then $\phi_1, \phi_2, \ldots, \phi_n \models \psi$. 


Proof outline

Proof by structural induction on the proof (as a graph) for $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$.

1. **Base case:** Trivial proof. It must be a premise. Thus the sequent is $\phi \vdash \phi$. Whenever $\phi$ evaluates to $T$, $\phi$ evaluates to $T$. Thus $\phi \models \phi$.

2. **Inductive step:** Analyze every possible application of a proof rule, and use hypothesis to show $\phi_1, \phi_2, \ldots, \phi_n \models \psi$. 
Completeness of Propositional Logic

Completeness
Let $\phi_1, \phi_2, \ldots, \phi_n$ and $\psi$ be propositional formulas. If $\phi_1, \phi_2, \ldots, \phi_n \models \psi$, then $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$. 
Proof outline

To show: If $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$, then $\phi_1, \phi_2, \ldots, \phi_n \models \psi$.

Step 1: Show $\models \phi_1 \rightarrow (\phi_2 \rightarrow (\phi_3 \rightarrow (\cdots (\phi_n \rightarrow \psi) \cdots )))$.

Step 2: Show $\vdash \phi_1 \rightarrow (\phi_2 \rightarrow (\phi_3 \rightarrow (\cdots (\phi_n \rightarrow \psi) \cdots )))$.

Step 3: Show $\phi_1, \phi_2, \ldots, \phi_n \vdash \psi$.

In Step 2, construct proof for each line of the truth table, and assemble the proofs into a single proof. This proof uses structural induction on the formula $\psi$. 
Questions about Propositional Formula

- Is a given formula valid?
- Is a given formula satisfiable?
- Is a given formula invalid?
- Is a given formula unsatisfiable?
- Are two formulas equivalent?
Decision Problems

Definition

A decision problem is a question in some formal system with a yes-or-no answer.

Examples

The question whether a given propositional formula is satisfiable (unsatisfiable, valid, invalid) is a decision problem.

The question whether two given propositional formulas are equivalent is also a decision problem.
How to Solve the Decision Problem?

Question
How do you decide whether a given propositional formula is satisfiable/valid?

The good news
We can construct a truth table for the formula and check if some/all rows have $\top$ in the last column.
Satisfiability is Decidable

An algorithm for satisfiability

Using a truth table, we can implement an algorithm that returns “yes” if the formula is satisfiable, and that returns “no” if the formula is unsatisfiable.

Decidability

Decision problems for which there is an algorithm computing “yes” whenever the answer is “yes”, and “no” whenever the answer is “no”, are called decidable.

Decidability of satisfiability

The question, whether a given propositional formula is satisfiable, is decidable.
The Bad News

Concern

In practice, propositional formulas can be large. Example:
http://www.comp.nus.edu.sg/~cs5209/prop.txt

Techniques so far inadequate

Proving satisfiability/validity using truth tables or natural deduction is impractical for large formulas.
Is there a *practical* way of deciding satisfiability?

**Question**

Is there an *efficient* algorithm that decides whether a given formula is satisfiable?

More precisely...

Is there a *polynomial-time* algorithm that decides whether a given formula is satisfiable?

**Answer**

We do not know!
What *do* we know about satisfiability?

Truth assignment as witness
If the answer is “yes”, then a satisfying truth assignment can serve as a proof that the answer is indeed “yes”.

Witness for satisfiability
Such a proof is called a *witness*.

Checking the witness
We can quickly check whether indeed the witness assignment makes the formula true. This can be done in time proportional to the size of the formula.
The Complexity Class NP

Definition
Decision problems for which the “yes” answer has a proof that can be checked in polynomial time, are called \textit{NP}.

Origin of name
NP stands for “\textbf{N}on-deterministic \textbf{P}olynomial time”.

Original definition
NP is the set of decision problems solvable in polynomial time by a non-deterministic Turing machine.
Some History

- The concept of NP-completeness was introduced by Stephen Cook in 1971 at the 3rd Annual ACM Symposium on Theory of Computing.

- At the conference, there was a fierce debate whether there could be a polynomial time algorithm to solve such problems.

- John Hopcroft convinced the delegates that the problem should be put off to be solved at some later date.

- In 1972, Richard Karp presented 21 NP-completed problems.

- Cook and Leonid Levin proved independently that propositional satisfiability is in this class.
Algorithms for Proving Satisfiability of $\psi$

- Transform $\neg \psi$ into Conjunctive Normal Form $\text{ncnf}$ and prove validity (non-validity) of $\text{ncnf}$
- Transform $\psi$ into Conjunctive Normal Form $\text{cnf}$ and search for a satisfying valuation
  - Complete algorithms: guaranteed to terminate with correct answer
    - example: DPLL
  - Incomplete algorithms: Return “yes” for some satisfiable formulas, and run forever for other satisfiable formulas and all unsatisfiable formulas; example: WalkSAT
- Transform $\psi$ into DAG; return “yes” for some satisfiable formulas, return “no” for some unsatisfiable formulas, return “don’t know” otherwise; example: linear solver (1.6.1)
Natural Deduction

Propositional Logic as a Formal Language

Semantics of Propositional Logic

Soundness and Completeness

Conjunctive Normal Form
Conjunctive Normal Form

Definition
A literal $L$ is either an atom $p$ or the negation of an atom $\neg p$. A formula $C$ is in *conjunctive normal form* (CNF) if it is a conjunction of clauses, where each clause is a disjunction of literals:

\[
\begin{align*}
L & ::= p | \neg p \\
D & ::= L | L \lor D \\
C & ::= D | D \land C
\end{align*}
\]
Examples

\((\neg p \lor q \lor r) \land (\neg q \lor r) \land (\neg r)\) is in CNF.

\((\neg p \lor q \lor r) \land ((p \land \neg q) \lor r) \land (\neg r)\) is not in CNF.

\((\neg p \lor q \lor r) \land \neg (\neg q \lor r) \land (\neg r)\) is not in CNF.
Usefulness of CNF

Lemma
A disjunction of literals $L_1 \lor L_2 \lor \cdots \lor L_m$ is valid iff there are $1 \leq i, j \leq m$ such that $L_i$ is $\neg L_j$.

How to disprove

\[
\models (\neg q \lor p \lor r) \land (\neg p \lor r) \land q
\]

Disprove any of:

\[
\models (\neg q \lor p \lor r) \quad \models (\neg p \lor r) \quad \models q
\]
Usefulness of CNF

Lemma
A disjunction of literals \( L_1 \lor L_2 \lor \cdots \lor L_m \) is valid iff there are \( 1 \leq i, j \leq m \) such that \( L_i \) is \( \neg L_j \).

How to prove

\[ \models (\neg q \lor p \lor q) \land (p \lor r \neg p) \land (r \lor \neg r) \]

Prove all of:

\[ \models (\neg q \lor p \lor q) \quad \models (p \lor r \neg p) \quad \models (r \lor \neg r) \]
Usefulness of CNF

Proposition

Let $\phi$ be a formula of propositional logic. Then $\phi$ is satisfiable iff $\neg \phi$ is not valid.

Satisfiability test

We can test satisfiability of $\phi$ by transforming $\neg \phi$ into CNF, and show that some clause is not valid.
Theorem

Every formula in the propositional calculus can be transformed into an equivalent formula in CNF.
Algorithm for CNF Transformation

1. Eliminate implication using:
   \[ A \rightarrow B \equiv \neg A \lor B \]

2. Push all negations inward using De Morgan’s laws:
   \[ \neg (A \land B) \equiv (\neg A \lor \neg B) \]
   \[ \neg (A \lor B) \equiv (\neg A \land \neg B) \]

3. Eliminate double negations using the equivalence \[ \neg \neg A \equiv A \]

4. The formula now consists of disjunctions and conjunctions of literals. Use the distributive laws
   \[ A \lor (B \land C) \equiv (A \lor B) \land (A \lor C) \]
   \[ (A \land B) \lor C \equiv (A \lor C) \land (B \lor C) \]
   to eliminate conjunctions within disjunctions.
Example

\[
(\neg p \rightarrow \neg q) \rightarrow (p \rightarrow q) \equiv \neg(\neg\neg p \lor \neg q) \lor (\neg p \lor q) \\
\equiv (\neg\neg\neg p \land q) \lor (\neg p \lor q) \\
\equiv (\neg p \land q) \lor (\neg p \lor q) \\
\equiv (\neg p \lor \neg p \lor q) \land (q \lor \neg p \lor q)
\]