3 Propositional Logic

3.1 Syntax

3.2 Semantics

3.3 Equivalence and Normal Forms

3.4 Proof Procedures

3.5 Properties

3.1 Syntax

- **Definition 3.0** An alphabet Σ consists of a finite or countably infinite set. The elements of this set are calles symbols. Without loss of generality (Wlog) we assume that $\Lambda \notin \Sigma$.

  The set of words (or strings) over Σ is denoted by Σ^* and is defined as follows:

  1. $\Lambda \in \Sigma^*$.
  2. If $w \in \Sigma^*$ and $a \in \Sigma$, then $aw \in \Sigma^*$.
  3. There are no other words in Σ^*.

  Λ is called empty word or empty string.

- **Example** Let $\Sigma = \{1, 2\}$. We can form the following words:

$$\Lambda,\ 1\Lambda,\ 2\Lambda,\ 11\Lambda,\ 12\Lambda,\ 21\Lambda,\ 22\Lambda,\ 111\Lambda,\ \ldots\ .$$

3.1 Alphabet of Propositional Logic

- **Definition 3.1** An **alphabet of propositional logic** consists of
  - a (countably) infinite set $\mathcal{R} = \{p_1, p_2, \ldots\}$ of propositional variables,
  - the set $\{\neg/1, \wedge/2, \vee/2, \rightarrow/2, \leftrightarrow/2\}$ of connectives, and
  - the special characters "(" and ")".
- Indices are sometimes omitted.
- We will occasionally use other letters to denote variables.
- Propositional variables are also called **nullary relation symbols**.
- Different alphabets of propositional logic differ in $\mathcal{R}$.

Propositional Formulas

- **Definition 3.2**
  A (propositional) atomic formula, briefly called atom, is a propositional variable.

- **Definition 3.3** The set of propositional formulas is the smallest set $\mathcal{L}(\mathcal{R})$ with the following properties:

  1. If F is an atomic formula, then $F \in \mathcal{L}(\mathcal{R})$.
  2. If $F \in \mathcal{L}(\mathcal{R})$, then $\neg F \in \mathcal{L}(\mathcal{R})$.
  3. If $\circ/2$ is a binary connective, $F,\ G \in \mathcal{L}(\mathcal{R})$, then $(F \circ G) \in \mathcal{L}(\mathcal{R})$.

- Does such a smallest set exist?
  - There are sets, which satisfy all properties.
  - The intersection of such sets satisfies the properties too.
  - The smallest set is the intersection of all sets satisfying the properties.

- **Notation** A (possibly indexed) denotes an atom,
  F, G, H (possibly indexed) denote propositional formulas.

Structural Induction

- How can we prove properties of sets of formulas?
- **Theorem 3.4 (Principle of structural induction)**
  Every propositional formula has a property E, if the following conditions are met:
  1. **Induction Base**: Every atom has property E.
  2. **Induction Steps**:
     If F has property E, then $\neg F$ has property E as well.
     If $\circ/2$ is a binary connective, and F and G have property E,
     then $(F \circ G)$ has property E as well.
- **Proof** Let $\mathcal{E}$ be the set of all formulas satisfying E.
  - $\mathcal{E} \subseteq \mathcal{L}(\mathcal{R})$.
  - $\mathcal{E}$ satisfies the three conditions of definition 3.3.
  - Because $\mathcal{L}(\mathcal{R})$ is the smallest set satisfying these conditions,
    $\mathcal{L}(\mathcal{R}) \subseteq \mathcal{E}$ must hold. qed

Structural Recursion

- **How can we define functions acting on formulas?**
- **Theorem 3.5 (Principle of structural recursion)**
  Let M be an arbitrary set
  and let the following functions be given:
  - $\mathit{foo}_{\mathcal{R}} : \mathcal{R} \longrightarrow M$,
  - $\mathit{foo}_{\neg} : M \longrightarrow M$,
  - $\mathit{foo}_{\circ} : M \times M \longrightarrow M$ **for** $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.

  Then there is exactly one function $\mathit{foo} : \mathcal{L}(\mathcal{R}) \longrightarrow M$
  satisfying the following conditions:
  1. **Recursion base** $\mathit{foo}(A) = \mathit{foo}_{\mathcal{R}}(A)$ **for all** $A \in \mathcal{R}$.
  2. **Recursion steps**
     $\mathit{foo}(\neg G) = \mathit{foo}_{\neg}(\mathit{foo}(G))$ **for all** $G \in \mathcal{L}(\mathcal{R})$.
     $\mathit{foo}((G_1 \circ G_2)) = \mathit{foo}_{\circ}(\mathit{foo}(G_1), \mathit{foo}(G_2))$ **for all** $G_1, G_2 \in \mathcal{L}(\mathcal{R})$.
- **Proof** $\rightsquigarrow$ **Exercise**

Structural Recursion – Example

- $M = \mathbb{N}$.
- $\mathit{foo}_{1\mathcal{R}}(A) = 0$ for all $A \in \mathcal{R}$.
- $\mathit{foo}_{1\neg}(X) = X$ for all $X \in M$.
- $\mathit{foo}_{1\circ}(X, Y) = X + Y + 1$ for all $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ and all $X, Y \in M$.
- We obtain:

$$\mathit{foo}_1(F) = \begin{cases} 0 & \text{if } F \text{ is an atom,} \\ \mathit{foo}_1(G) & \text{if } F \text{ is of the form } \neg G, \\ \mathit{foo}_1(G_1) + \mathit{foo}_1(G_2) + 1 & \text{if } F \text{ is of the form } (G_1 \circ G_2). \end{cases}$$

- Often $\mathit{foo}_{1\mathcal{R}}$, $\mathit{foo}_{1\neg}$ and $\mathit{foo}_{1\circ}$ are not explicitely specified.
- *foo* is a function symbol af a meta language.

Structural Recursion – More Examples

- $\mathit{foo}_2(F) = \begin{cases} 1 & \text{if } F \text{ is an atom,} \\ \mathit{foo}_2(G) & \text{if } F \text{ is of the form } \neg G, \\ \mathit{foo}_2(G_1) + \mathit{foo}_2(G_2) & \text{if } F \text{ is of the form } (G_1 \circ G_2). \end{cases}$

  counts the number of occurrences of atoms in F.

- $\mathit{foo}_3(F) = \begin{cases} F & \text{if } F \text{ is an atom,} \\ \neg\mathit{foo}_3(G) & \text{if } F \text{ is of the form } \neg G, \\ \mathit{foo}_3(G_1) \circ \mathit{foo}_3(G_2) & \text{if } F \text{ is of the form } (G_1 \circ G_2). \end{cases}$

  eliminates parentheses in F.

- $\mathit{foo}_4(F) = \{\Lambda\} \cup \begin{cases} \emptyset & \text{if } F \text{ is an atom,} \\ \{1\pi \mid \pi \in \mathit{foo}_4(G)\} & \text{if } F \text{ is of the form } \neg G, \\ \{1\pi_1 \mid \pi_1 \in \mathit{foo}_4(G_1)\} \cup \{2\pi_2 \mid \pi_2 \in \mathit{foo}_4(G_2)\} & \\ & \text{if } F \text{ is of the form } (G_1 \circ G_2) \end{cases}$

  computes the set of positions in F.

Subformulas

- **Definition 3.6** Let F be a propositional formula. The **set of subformulas** of F is the smallest set of formulas $\mathcal{S}(F)$ satisfying the following conditions:
  1. $F \in \mathcal{S}(F)$.
  2. If $\neg G \in \mathcal{S}(F)$, then $G \in \mathcal{S}(F)$.
  3. If $(G_1 \circ G_2) \in \mathcal{S}(F)$, then $G_1,\ G_2 \in \mathcal{S}(F)$.

- **Example**

$$\begin{aligned}&\mathcal{S}(\neg((p_1 \longrightarrow p_2) \vee p_1))\\ &= \{\neg((p_1 \longrightarrow p_2) \vee p_1),\ ((p_1 \longrightarrow p_2) \vee p_1),\ (p_1 \longrightarrow p_2),\ p_1,\ p_2\}.\end{aligned}$$

3.2 Semantics

- What is the meaning of $\neg((p_1 \longrightarrow p_2) \vee p_1)$?
- Here: formulas may be true or false $\rightsquigarrow$ binary logic.

3.2.1 The Structure of the Truth Values

- A structure consists of a set of individuals and a set of functions defined on it.
- The set of truth values $\mathcal{W}$ is the set $\{\top, \bot\}$.
- Unary functions of type $\mathcal{W} \to \mathcal{W}$:
  - Negation $\neg^*$: $\neg^*(\top) = \bot$ and $\neg^*(\bot) = \top$.
  - Identity.
  - Projections onto $\bot$ and $\top$.
- Binary functions of type $\mathcal{W} \times \mathcal{W} \to \mathcal{W}$:
  - Conjunction $\wedge^*/2$.
  - Disjunction $\vee^*/2$.
  - Implication $\to^* /2$.
  - NAND $\uparrow^*/2$.
  - NOR $\downarrow^* /2$.
  - Equivalence $\leftrightarrow^* /2$. Etc.

The Functions over the Set of Truth Values

		$\wedge^*$	$\vee^*$	$\rightarrow^*$	$\leftarrow^*$	$\uparrow^*$	$\downarrow^*$	$\not\rightarrow^*$	$\not\leftarrow^*$
$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\bot$	$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\bot$	$\top$	$\bot$	$\top$	$\top$	$\bot$	$\top$	$\bot$
$\bot$	$\top$	$\bot$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\top$	$\top$	$\bot$	$\bot$

		$\leftrightarrow^*$	$\not\leftrightarrow^*$						
$\top$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\top$	$\bot$
$\top$	$\bot$	$\bot$	$\top$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\top$
$\bot$	$\top$	$\bot$	$\top$	$\top$	$\bot$	$\bot$	$\top$	$\top$	$\bot$
$\bot$	$\bot$	$\top$	$\bot$	$\top$	$\bot$	$\bot$	$\top$	$\bot$	$\top$

▷ **Here** **negation, conjunction, disjunction, implication, equivalence.**

3.2.2 Interpretations and Models

- **Definition 3.7** A (propositional) interpretation $I = (\mathcal{W}, \cdot^I)$ consists of the set $\mathcal{W}$ and a mapping $\cdot^I : \mathcal{L}(\mathcal{R}) \to \mathcal{W}$ with:

$$[F]^I = \begin{cases} w \in \mathcal{W} & \text{if } F \in \mathcal{R}, \\ \neg^*[G]^I & \text{if } F \text{ is of the form } \neg G, \\ ([G_1]^I \circ^* [G_2]^I) & \text{if } F \text{ is of the form } (G_1 \circ G_2). \end{cases}$$

- An interpretation $I = (\mathcal{W}, \cdot^I)$ is uniquely defined by specifying how $\cdot^I$ acts on propositional variables.
- Recall that A denotes an atom.
- We will sometimes identify an interpretation $I = (\mathcal{W}, \cdot^I)$ with $\{A \mid [A]^I = \top\}$.
- We often write F^I instead of $[F]^I$.
- I (possibly indexed) denotes an interpretation.

Satisfiability of Formulas

- **Definition 3.10** Let $F \in \mathcal{L}(\mathcal{R})$.
  - F is called **satisfiable**
    if there is an interpretation $I = (\mathcal{W}, \cdot^I)$ with $F^I = \top$.
  - F is called **valid** (or F is a **tautology**)
    if for all interpretations $I = (\mathcal{W}, \cdot^I)$ we have: $F^I = \top$.
  - F is called **falsifiable**
    if there is an interpretation $I = (\mathcal{W}, \cdot^I)$ with $F^I = \bot$.
  - F is called **unsatisfiable**
    if for all interpretations $I = (\mathcal{W}, \cdot^I)$ we have: $F^I = \bot$.

Models

- **Definition 3.11** An interpretation $I = (\mathcal{W}, \cdot^I)$ is called **model** for a propositional formula F, in symbols $I \models F$, if $F^I = \top$.
- **Abbreviations**
  - If I is not a model for the formula F, i. e. if $F^I = \bot$, then we write $I \not\models F$.
  - If F is valid, then we write $\models F$.

Validity and Unsatisfiability

- **Theorem 3.12** A propositional formula F is valid ($\models F$), iff $\neg F$ is unsatisfiable.
- **Proof** $\models F$ iff all interpretations are models for F
  iff no interpretation is a model for $\neg F$
  iff $\neg F$ is unsatisfiable. qed

Satisfiability of a Set of Formulas

- **Definition 3.13** Let $\mathcal{G}$ be a set of formulas.
  - $\mathcal{G}$ is **satisfiable**,
    if there is an interpretation mapping each element $F \in \mathcal{G}$ to $\top$.
  - An interpretation I is called **model** for $\mathcal{G}$, in symbols $I \models \mathcal{G}$,
    if I is model for all $F \in \mathcal{G}$.

Logical Consequence

- **Definition 3.14** A propositional formula F is a (propositional) consequence of a set of propositional formulas $\mathcal{G}$, in symbols $\mathcal{G} \models F$, iff for every interpretation I we have: if $I \models \mathcal{G}$, then $I \models F$ as well.

- **Exercises**
  - Does $\{p, (p \rightarrow q)\} \models q$ hold?
  - Does $\{(p \vee q)\} \models q$ hold?
  - Does $\{(p \wedge \neg p)\} \models q$ hold?

- **Theorem 3.15** Let $F, F_1, \ldots, F_n$ be propositional formulas.
  $\{F_1, \ldots, F_n\} \models F$ holds iff $\models ((\ldots(F_1 \wedge F_2) \wedge \ldots \wedge F_n) \rightarrow F)$ holds.

- **Proof** $\rightsquigarrow$ **Exercise**

Truth Tabling

- How can we compute the value of a formula F under all possible interpretations?
- Computing a truth table:
  1. Let $m = |\mathcal{S}(F)|$ be the number of subformulas of F.
  2. Let $n = |\{p \in \mathcal{R} \mid p \in \mathcal{S}(F)\}|$ be the number of propositional variables occurring in F.
  3. Form a table with 2^n rows and m columns, where the first n columns are marked by the n propositional variables occurring in F, the last column is marked by F, and the remaining columns are marked by the other subformulas of F.
  4. Fill in the first n columns with $\top$ und $\bot$ as follows: In the first column fill in alternating downwards $\top\bot\top\bot\ldots$, in the second column $\top\top\bot\bot\ldots$, etc.
  5. Calculate the values in the remaining columns using the known functions on the set of truth values.

Working with Truth Tables

- Let F be a formula and T a corresponding truth table. Then:
  - F is satisfiable **iff** T contains a row with $\top$ in the last column.
  - F is valid **iff** all rows in T have $\top$ in the last column.
  - F is falsifiable **iff** T contains a row with $\bot$ in the final column.
  - F is unsatisfiable **iff** all rows T have $\bot$ in the final column.
- We can apply truth tabling to decide whether $\mathcal{G} \models F$ holds for finite $\mathcal{G}$
  $\rightsquigarrow$ **Exercise**.
- **Exercise**
  Write a (Prolog) program which generates a truth table for a given formula F.

3.3 Equivalence and Normal Forms

3.3.1 Semantic Equivalence

- **Definition 3.16** Two propositional formulas F and G are **semantically equivalent**, in symbols $F \equiv G$, if for all interpretations I we have: $I \models F$ iff $I \models G$.

- **Observe** $\equiv$ is an equivalence relation:
  - $\equiv$ is reflexive: $F \equiv F$.
  - $\equiv$ is symmetric: If $F \equiv G$, then $G \equiv F$.
  - $\equiv$ is transitive: If $F \equiv G$ and $G \equiv H$, then $F \equiv H$.

  This follows immediately from Definition 3.16 $\rightsquigarrow$ **Exercise**.

Well-Known Semantic Equivalences

- **Theorem 3.17** **The following equivalences hold:**

$(F \wedge F)$	$\equiv$	F	
$(F \vee F)$	$\equiv$	F	**idempotency**
$(F \wedge G)$	$\equiv$	$(G \wedge F)$	
$(F \vee G)$	$\equiv$	$(G \vee F)$	**commutativity**
$((F \wedge G) \wedge H)$	$\equiv$	$(F \wedge (G \wedge H))$	
$((F \vee G) \vee H)$	$\equiv$	$(F \vee (G \vee H))$	**associativity**
$((F \wedge G) \vee F)$	$\equiv$	F	
$((F \vee G) \wedge F)$	$\equiv$	F	**absorption**
$(F \wedge (G \vee H))$	$\equiv$	$((F \wedge G) \vee (F \wedge H))$	
$(F \vee (G \wedge H))$	$\equiv$	$((F \vee G) \wedge (F \vee H))$	**distributivity**

Theorem 3.17 (continued)

- **Theorem 3.17 (continued)** The following equivalences hold:

$\neg\neg F$	$\equiv$	F	double negation
$\neg(F \wedge G)$	$\equiv$	$(\neg F \vee \neg G)$	
$\neg(F \vee G)$	$\equiv$	$(\neg F \wedge \neg G)$	de Morgan
$(F \vee G)$	$\equiv$	F, if F is valid	
$(F \wedge G)$	$\equiv$	G, if F is valid	tautology
$(F \vee G)$	$\equiv$	G, if F is unsatisfiable	
$(F \wedge G)$	$\equiv$	F, if F is unsatisfiable	unsatisfiability
$(F \leftrightarrow G)$	$\equiv$	$((F \wedge G) \vee (\neg G \wedge \neg F))$	equivalence
$(F \rightarrow G)$	$\equiv$	$(\neg F \vee G)$	implication

- **Proof** $\rightsquigarrow$ **Exercise**

Positions

- **Recall**

$$\mathit{foo}_4(F) = \{\Lambda\} \cup \begin{cases} \emptyset & \text{if } F \text{ is an atom,} \\ \{1\pi \mid \pi \in \mathit{foo}_4(G)\} & \text{if } F \text{ is of form } \neg G, \\ \{1\pi_1 \mid \pi_1 \in \mathit{foo}_4(G_1)\} \cup \{2\pi_2 \mid \pi_2 \in \mathit{foo}_4(G_2)\} & \\ & \text{if } F \text{ is of the form } (G_1 \circ G_2). \end{cases}$$

- $\mathit{foo}_4(F)$ is the set of **positions** in a propositional logic formula F.
- **Notation** We denote this set also with $\mathcal{P}(F)$.
- Positions are words over $\{1, 2\}$, where Λ denotes the empty word.

Replacements

- The subformula of F at position $\pi \in \mathcal{P}(F)$, in symbols $F\lceil\pi\rceil$, is defined by:
  1. $F\lceil\Lambda\rceil = F$.
  2. $F\lceil 1\pi\rceil = G\lceil\pi\rceil$, if F is of the form $\neg G$.
  3. $F\lceil i\pi\rceil = G_i\lceil\pi\rceil$, if F is of the form $(G_1 \circ G_2)$ and $i \in \{1, 2\}$.
- The replacement of the subformula of F at position $\pi \in \mathcal{P}(F)$ by H, in symbols $F\lceil\pi \mapsto H\rceil$, is defined by:
  1. $F\lceil\Lambda \mapsto H\rceil = H$,
  2. $F\lceil 1\pi \mapsto H\rceil = \neg(G\lceil\pi \mapsto H\rceil)$, if F is of the form $\neg G$.
  3. $F\lceil 1\pi \mapsto H\rceil = (G_1\lceil\pi \mapsto H\rceil \circ G_2)$, if F is otf $(G_1 \circ G_2)$.
  4. $F\lceil 2\pi \mapsto H\rceil = (G_1 \circ G_2\lceil\pi \mapsto H\rceil)$, if F is otf $(G_1 \circ G_2)$.

The Replacement Theorem

- **Theorem 3.18 (Replacement Theorem)**
  Let $F \in \mathcal{L}(\mathcal{R})$, $\pi \in \mathcal{P}(F)$, $F\lceil\pi\rceil = G$ and $G \equiv H$.
  Then, $F \equiv F\lceil\pi \mapsto H\rceil$.

- **Proof** Structural induction on π.

- **Induction basis** Let $\pi = \Lambda$.

  - $F = F\lceil\Lambda\rceil = G \equiv H = F\lceil\Lambda \mapsto H\rceil$. ✓

- **Induction hypothesis (IH)** The theorem holds for π'.

Proof Replacement Theorem (Continuation)

- **Induction step** Let $\pi = i\pi'$. We distinguish two cases:

  $i = 1$ $\pi = 1\pi' \in \mathcal{P}(F)$, hence F must be otf $\neg F'$ or $(G_1 \circ G_2)$.

  $\neg F'$ $F\lceil 1\pi' \mapsto H\rceil = (\neg F')\lceil 1\pi' \mapsto H\rceil = \neg(F'\lceil \pi' \mapsto H\rceil)$.

  From IH we conclude: $F' \equiv F'\lceil \pi' \mapsto H\rceil$

  Hence, for all interpretations I we find:

$$
\begin{array}{rcll}
[F]^I & = & [\neg F']^I & \text{Assumption} \\
 & = & \neg^*[F']^I & \text{Def } I \\
 & = & \neg^*[F'\lceil \pi' \mapsto H\rceil]^I & \text{IH and Def } \equiv \\
 & = & [\neg(F'\lceil \pi' \mapsto H\rceil)]^I & \text{Def } I \\
 & = & [(\neg F')\lceil 1\pi' \mapsto H\rceil]^I & \text{Def replacement} \\
 & = & [F\lceil 1\pi' \mapsto H\rceil]^I & \text{Assumption}
\end{array}
$$

  Hence, $F \equiv F\lceil 1\pi' \mapsto H\rceil$ ✓

  $(G_1 \circ G_2)$ analogously

  $i = 2$ analogously

qed

Agreement

- If $\pi \notin \mathcal{P}(F)$ then $F\lceil \pi \mapsto H \rceil := F$.
- Let $F\lceil \pi \rceil = G$.
  If it is obvious from the context which subformula G is to be replaced, then we write $F\lceil G \mapsto H \rceil$ instead of $F\lceil \pi \mapsto H \rceil$.
- Theorem 3.18 allows to replace all occurrences of the connectives $\leftrightarrow$ and $\rightarrow$.
  - Wlog we consider only formulas over $\mathcal{R} \cup \{\neg, \wedge, \vee, (,)\}$.

3.3.2 Negation Normal Form

- **Definition 3.19** A propositional formula F is in negation normal form if all negation signs $\neg$ occurring in F appear directly in front of propositional variables.
- **Proposition 3.20** There is an algorithm which transforms any formula into a semantically equivalent formula in negation normal form.
- **Algorithm** Given F. While F is not in negation normal form do:
  - Select a subformula of F having the form $\neg G$ with $G \notin \mathcal{R}$.
  - If $\neg G = \neg\neg H$, then replace $\neg\neg H$ by H.
  - If $\neg G = \neg(G_1 \wedge G_2)$, then replace $\neg(G_1 \wedge G_2)$ by $(\neg G_1 \vee \neg G_2)$.
  - If $\neg G = \neg(G_1 \vee G_2)$, then replace $\neg(G_1 \vee G_2)$ by $(\neg G_1 \wedge \neg G_2)$.

The Rules of the Algorithm

- **The rules will be abbreviated by:**

$$\frac{\neg\neg H}{H} \qquad \frac{\neg(G_1 \wedge G_2)}{(\neg G_1 \vee \neg G_2)} \qquad \frac{\neg(G_1 \vee G_2)}{(\neg G_1 \wedge \neg G_2)}$$

- **The algorithm is (don't care) non-deterministic!**
- **Example**

$$\begin{aligned} \neg(r \vee \neg(\neg p \wedge q)) &\equiv (\neg r \wedge \neg\neg(\neg p \wedge q)) && \textbf{de Morgan} \\ &\equiv (\neg r \wedge (\neg p \wedge q)) && \textbf{double negation} \end{aligned}$$

3.3.3 Clause Form

- **Definition 3.21**
  A **literal** is a propositional variable, or a negated propositional variable.

- **Notation**
  - L (possible indexed) denotes a literal.
  - **Generalized disjunction**

$$[F_1, \ldots, F_n] = (\ldots((F_1 \vee F_2) \vee F_3) \vee \ldots \vee F_n)$$

  - **Generalized conjunction**

$$\langle F_1, \ldots, F_n \rangle = (\ldots((F_1 \wedge F_2) \wedge F_3) \wedge \ldots \wedge F_n)$$

  - **Observe** $[F] = F$ and $\langle F \rangle = F$. Hence, $[F] \equiv F$ and $\langle F \rangle \equiv F$.
  - **Empty generalized disjunction**: $[\,]$ with $[\,]^I = \bot$ for all I.
  - **Empty generalized conjunction**: $\langle\rangle$ with $\langle\rangle^I = \top$ for all I.
  - **Observe** $(G \vee [\,]) \equiv G$ and $(G \wedge \langle\rangle) \equiv G$.

Clauses

- **Definition 3.22**
  - A **clause** is a generalized disjunction $[L_1, \ldots, L_n]$, $n \geq 0$, where every L_i, $1 \leq i \leq n$, is a literal.
  - A **dual clause** is a generalized conjunction $\langle L_1, \ldots, L_n \rangle$, $n \geq 0$, where every L_i, $1 \leq i \leq n$, is a literal.
  - A formula is in **conjunctive normal form**, or **clause form**, **iff** it is of the form $\langle C_1, \ldots, C_m \rangle$, $m \geq 0$, and if every C_j, $1 \leq j \leq m$, is a clause.
  - A formula is in **disjunctive normal form**, or **dual clause form**, **iff** it is of the form $[C_1, \ldots, C_m]$, $m \geq 0$, and if every C_j, $1 \leq j \leq m$, is a dual clause.
- **Notation** C (possibly indexed) denotes a clause.

An Observation

- Let F be a formula in clause form. The following holds:
  - F is unsatisfiable **if** a clause occurring in F is unsatisfiable.
  - A clause is unsatisfiable **iff** it is of the form $[\,]$.

Transformation into Clause Form

- **Theorem 3.23**
  1. There is an algorithm which transforms any formula into a semantically equivalent formula in clause form.
  2. There is an algorithm which transforms any formula into a semantically equivalent formula in dual clause form.

- **Proof of 1**
  - We have to specify the algorithm.
  - We have to show that the algorithm is correct or sound,
    i.e., if it outputs G given F,
    then G must be in clause form and $F \equiv G$ must hold.
  - We have to show that the algorithm terminates,
    i.e., it stops in finite time given F.

An Algorithm for the Transformation into Clause Form

- **Input**: A propositional formula F.
  Output: A formula, which is in conjunctive normal form and is equivalent to F.
  $G := \langle [F] \rangle$. (G is a conjunction of disjunctions.)
  While G is not in conjunctive normal form do:
  Select a non-clausal element H from G.
  Select a non-literal element K from H.
  Apply the rule among the following ones which is applicable.

$$\frac{\neg\neg D}{D} \qquad \frac{(D_1 \wedge D_2)}{D_1 \mid D_2} \qquad \frac{\neg(D_1 \wedge D_2)}{\neg D_1, \neg D_2} \qquad \frac{(D_1 \vee D_2)}{D_1, D_2} \qquad \frac{\neg(D_1 \vee D_2)}{\neg D_1 \mid \neg D_2}$$

- A rule $\frac{D}{D'}$ is applicable to K if K is of the form D.
  If applied, then K is replaced by D'.

- A rule $\frac{D}{D_1 | D_2}$ is applicable to K if K is of the form D.
  If applied, H is replaced by two disjunctions:
  The first one is obtained from H by replacing the occurrence of D by D_1.
  The second one is obtained from H by replacing the occurrence of D by D_2.

An Example and a Lemma

- **Example**

$$\langle[\neg(p \vee (\neg(p \wedge q) \wedge \neg r))]\rangle$$
$$\langle[\neg p],\ [\neg(\neg(p \wedge q) \wedge \neg r)]\rangle$$
$$\langle[\neg p],\ [\neg\neg(p \wedge q), \neg\neg r]\rangle$$
$$\langle[\neg p],\ [(p \wedge q),\ \neg\neg r]\rangle$$
$$\langle[\neg p],\ [(p \wedge q),\ r]\rangle$$
$$\langle[\neg p],\ [p,\ r],\ [q,\ r]\rangle$$

- **Lemma 3.24** **If a propositional formula G is a conjunction of disjunctions and G' is obtained from G by applying one of the rules of the algorithm, then G' is a conjunction of disjunctions and $G \equiv G'$.**

- **Proof $\rightsquigarrow$ Exercise.**

Induction Principle for While-Loops

- A **loop invariant** for a while-loop W is a statement E having the following property: If E holds, then E still holds after one execution of the body of W.
- **Theorem 3.25 (Induction Principle for While-Loops)**
  If a statement E holds before entering a loop W, and is a loop invariant for W, then E holds after termination of W as well (if W terminates).
- **Proof** $\rightsquigarrow$ Literature.

Soundness of the Algorithm

- Let F be the given propositional formula and let E be the statement: G ist a conjunction of disjunctions and $F \equiv G$ holds.
  - Initially G is defined to be $\langle [F] \rangle$.
  - Because $F \equiv \langle [F] \rangle$ holds, E holds before entering the while-loop.
  - From Lemma 3.24 we learn that E is a loop invariant for the while-loop.
  - Hence, by Theorem 3.25 E holds when the loop has terminated.
  - The loop terminates only when G is in conjunctive normal form.
  - $\rightsquigarrow$ The algorithm is sound!

Termination of the Algorithm

- When does a non-deterministic algorithm terminate?
  - **Weak Termination**
    There is at least one way to execute the algorithm such that it terminates.
  - **Strong Termination** The algorithm terminates independently of which choice we make in the non-deterministic situations.

- **Definition 3.26** The function *r*, called **rank** of a propositional formula, is defined over $\mathcal{L}(\mathcal{R})$ as follows:

$$r(H) = \begin{cases} 0 & \text{if } H \text{ is a literal,} \\ r(F) + 1 & \text{if } H \text{ is of the form } \neg\neg F, \\ r(F) + r(G) + 1 & \text{if } H \text{ is of the form } (F \wedge G) \text{ or } (F \vee G), \\ r(\neg F) + r(\neg G) + 1 & \text{if } H \text{ is of the form } \neg(F \wedge G) \text{ or } \neg(F \vee G). \end{cases}$$

Multisets

- A **multiset** is comparable to a set,
  but the individual elements may occur multiple times in it.

- **Here** multisets of natural numbers.
  - $\dot{\{}1, 1, 2, 2, 2, 3\dot{\}}$.
  - $M_1 \succ M_2$ holds **iff** M_2 is obtained from M_1 by deleting a number n from M_1 and replacing it by finitely many numbers which are all less than n.
  - $\dot{\{}1, 1, 2, 2, 2, 3\dot{\}} \succ \dot{\{}1, 1, 1, 1, 2, 2, 2, 2\dot{\}}$.
  - **Observation** there cannot be infinitely long sequences $(M_i \mid i \geq 0)$ of finite multisets of natural numbers such that $M_0 \succ M_1 \succ \ldots$.
    Proof see Dershowitz, Manna: Proving Termination with Multiset Orderings, CACM 22, 465-475 (1979).

Proof of Termination

- Let $G = \langle [H_{11}, \ldots, H_{1,n_1}], \ldots, [H_{m1}, \ldots, H_{mn_m}] \rangle$. We define:

$$f(G) \;=\; \dot{\{} \sum_{j=1}^{n_1} r(H_{1j}), \ldots, \sum_{j=1}^{n_m} r(H_{mj}) \dot{\}}$$

- **Observation** With every execution of the while-loop, the associated multisets become smaller with respect to $\succ/2$ (Proof $\rightsquigarrow$ Exercise) $\rightsquigarrow$ Termination. qed

- **Example**

$$\begin{array}{ll}
\langle [\neg(p \vee (\neg(p \wedge q) \wedge \neg r))] \rangle & \dot{\{}5\dot{\}} \\
\langle [\neg p],\ [\neg(\neg(p \wedge q) \wedge \neg r)] \rangle & \dot{\{}0, 4\dot{\}} \\
\langle [\neg p],\ [\neg\neg(p \wedge q), \neg\neg r] \rangle & \dot{\{}0, 3\dot{\}} \\
\langle [\neg p],\ [(p \wedge q),\ \neg\neg r] \rangle & \dot{\{}0, 2\dot{\}} \\
\langle [\neg p],\ [(p \wedge q),\ r] \rangle & \dot{\{}0, 1\dot{\}} \\
\langle [\neg p],\ [p,\ r],\ [q,\ r] \rangle & \dot{\{}0, 0, 0\dot{\}}
\end{array}$$

- **Observation** If G is in clause form, then $f(G) = \dot{\{}0, \ldots, 0\dot{\}}$ and the number of clauses occurring in G is equal to the number of occurrences of 0 in $f(G)$.

3.4 Proof Methods

- $\{F_1, \ldots, F_n\} \models F$

  iff $(\langle F_1, \ldots, F_n \rangle \longrightarrow F)$ is valid

  iff $\neg(\langle F_1, \ldots, F_n \rangle \longrightarrow F)$ is unsatisfiable

  iff $\langle F_1, \ldots, F_n, \neg F \rangle$ is unsatisfiable.

- Truth tabling
- Resolution
- Semantic tableaus
- Calculus of natural deduction
- Sequent calculus
- Connection method
- Hilbert systems

3.4.1 Resolution

- Resolution is a method to show unsatisfiability of formulas.
- Here: Wlog we assume that formulas are in clause form.
- A formula in clause form in unsatisfiable if
  - it contains the empty clause, or
  - it can be transformed into a semantically equivalent formula which contains the empty clause.
- Resolution is search for the empty clause using an appropriate inference rule.

Example

- **Some Facts**

  K_1 If it is hot and humid, then it will rain.
  K_2 If it is humid, then it is hot.
  K_3 It is humid now.

- **Question**

  K_4 Will it rain?

- **Introducing propositional variables**

  p it is hot
  q it is humid
  r it will rain

- **Formalization**

  F_1 $((p \wedge q) \longrightarrow r)$
  F_2 $(q \longrightarrow p)$
  F_3 q
  F_4 r

Example (continued)

$$\{((p \wedge q) \rightarrow r),\ (q \rightarrow p),\ q\} \models r$$

iff

$$(\langle((p \wedge q) \rightarrow r),\ (q \rightarrow p),\ q\rangle \rightarrow r) \text{ is valid}$$

iff

$$\neg(\langle((p \wedge q) \rightarrow r),\ (q \rightarrow p),\ q\rangle \rightarrow r) \text{ is unsatisfiable}$$

iff

$$\langle((p \wedge q) \rightarrow r),\ (q \rightarrow p),\ q,\ \neg r\rangle \text{ is unsatisfiable}$$

iff

$$\langle[\neg p, \neg q, r],\ [\neg q, p],\ [q],\ [\neg r]\rangle \text{ is unsatisfiable}$$

How to Show Unsatisfiability?

- **Recall**

$$\langle [\neg p, \neg q, r],\ [\neg q, p],\ [q],\ [\neg r] \rangle$$

- **Observation**

$$\{[\neg p,\ \neg q, r],\ [q]\} \models [\neg p, r]$$

- **Hence**

$$\langle [\neg p, \neg q, r],\ [\neg q, p],\ [q],\ [\neg r] \rangle \ \equiv\ \langle [\neg p, \neg q, r],\ [\neg q, p],\ [q],\ [\neg r],\ [\neg p, r] \rangle$$

Resolution Rule

- **Definition 3.27** Let C_1 be a clause where the atom A occurs, and C_2 a clause where the negated atom $\neg A$ occurs. Let C be the result of performing the following steps:

  1. Remove all occurrences of A from C_1.
  2. Remove all occurrences of $\neg A$ from C_2.
  3. Combine the clauses obtained in this way disjunctively.

  C was generated applying the (propositional) resolution rule to C_1 and C_2, where A is the literal that was resolved upon.
  We also call C the resolvent of C_1 and C_2 wrt A.

- C may be the resolvent of C_1 and C_1.

Derivations and Refutations

- **Definition 3.28** Let $F = \langle C_1, \ldots, C_n \rangle$ be a formula in clause form.
  1. The sequence $(C_i \mid 1 \leq i \leq n)$ is a resolution derivation for F.
  2. If $(C_i \mid 1 \leq i \leq m)$ is a resolution derivation for F, and C_{m+1} is obtained by applying the resolution rule to two elements from $(C_i \mid 1 \leq i \leq m)$, then $(C_i \mid 1 \leq i \leq m+1)$ is a resolution derivation for F.
  3. A resolution derivation for F which contains the empty clause $[\,]$ is called resolution refutation for F.
- F may contain $[\,]$.
- It suffices to consider refutations in which $[\,]$ occurs once.
- We may assume that $[\,]$ is the last clause in a refutation.

Example (continued)

1	$[\neg p, \neg q, r]$	
2	$[\neg q, p]$	
3	$[q]$	
4	$[\neg r]$	
5	$[\neg q, \neg q, r]$	*res*$(1, 2)$
6	$[r]$	*res*$(3, 5)$
7	$[\,]$	*res*$(4, 6)$

Resolution Proofs

- **Definition 3.29** Let F be a propositional formula, and G a formula in clause form equivalent to $\neg F$.
  - A **propositional resolution proof** for F is a resolution refutation for G.
  - F is called **theorem** of the resolution calculus, if there is a resolution proof for F.
  - We denote by $\vdash_r F$, that F has a resolution proof.
- **Requirements**
  - **Soundness** If $\vdash_r F$, then $\models F$.
  - **Completeness** If $\models F$, then $\vdash_r F$.
- **Observation**
  - Let F be a formula with n propositional variables.
  - $(F \wedge (p \wedge \neg p))$ is unsatisfiable.
  - The proof consists of a single resolution step!

3.4.4 Further Proof Methods and Calculi

- **Definition 3.35** A **calculus** consists of
  - an alphabet,
  - a language,
  - a set of axioms, i. e., a set of formulas of the language, and
  - a set of inference rules defining the provability relation.
- **Definition 3.36** Let $\mathcal{L}$ be a logic and $\mathcal{C}$ a calculus, both defined over the same alphabet and the same language. Let $\models$ be the logical consequence relation of $\mathcal{L}$ and $\vdash$ the provability relation of $\mathcal{C}$.
  - $\mathcal{C}$ is **sound** wrt $\mathcal{L}$, if for every formula F of the language we find: if $\vdash F$ holds then so does $\models F$.
  - $\mathcal{C}$ is **complete** wrt $\mathcal{L}$, if for every formula F of the language we find: if $\models F$ holds then so does $\vdash F$.

A Characterization of Calculi

- **Definition 3.37** **A calculus is**
  - **positive, if its axioms are valid;**
  - **negative, if its axioms are unsatisfiable;**
  - **generating, if it operates by proving a formula starting out from the axioms;**
  - **analyzing, if it operates by reducing the formula to be proved to the axioms.**

Additional Calculi and Methods

- Calculi
  - Semantic Tableaus
  - Hilbert Systems
  - Calculus of Natural Deduction
  - Sequent Calculus
  - Davis-Putman-Logeman-Loveland (DPLL) Procedure
  - Connection Method
- Model Generation
  - Greedy Satisfiability Testing
  - Binary Decision Diagrams

Natural Deduction

- Gentzen 1935: How to represent logical reasoning in mathematics?
  - Calculus of natural deduction.
- It's alphabet is the alphabet of propositional logic.
- Remember $[\,]$ denotes some unsatisfiable formula.
- It's language is the language of propositional logic.

An Introductory Example

- Suppose we want to show the validity of

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

- A proof could look like the following:

Assume that $(p \vee (q \wedge r))$ holds. We distinguish two cases:
(i) Assume that p holds.
Hence, $(p \vee q)$ must hold.
(ii) Assume that $(q \wedge r)$ holds.
Hence, q and, consequently, $(p \vee q)$ must hold.
Thus, $(p \vee q)$ must hold in any case. (1)
Likewise, we conclude that $(p \vee r)$ must hold. (2)
From (1) and (2) we conclude that $((p \vee q) \wedge (p \vee r))$ holds.
We can now cancel the assumption and, thus, obtain
a proof of $((p \vee (q \wedge r)) \rightarrow ((p \vee q) \wedge (p \vee r)))$.

Inference Rules – Example

- **Elimination of the implication**

$$\frac{G \qquad (G \rightarrow F)}{F}\,(\rightarrow E)$$

- **Schema which needs to be instantiated, e.g.,**

$$\frac{(p \wedge q) \qquad ((p \wedge q) \rightarrow (q \wedge p))}{(q \wedge p)}\,(\rightarrow E)$$

Derivations – Some Notation

- Derivations have the form of trees, where the nodes are labeled by formulas.
- Derivations will be denoted by

$$\nabla$$

- A derivation where the root is labeled by F is denoted by

$$\begin{array}{c}\nabla\\ F\end{array}$$

  This will be called a derivation of F.
- A formula labeling a leaf node in a derivation is called hypothesis or assumption.
- If we want to single out a hypothesis G in a derivation of F, then we will write

$$\begin{array}{c}G\\ \nabla\\ F\end{array}$$

  This will be called a derivation of F from G.

Derivations – More Notation

- Some inference rule will cancel hypotheses.
- If a hypothesis has been cancelled, then this will be denoted by marking the hypothesis with $\lfloor\ \rfloor$.
- The hypotheses cancelled by the application a of a rule as well as a itself will be marked by a unique index.

Natural Deduction - Inference Rules

- **Negation**

$$\frac{[\,]}{G}\,(f) \qquad \frac{\begin{array}{c}\lfloor \neg F \rfloor \\ \nabla \\ [\,]\end{array}}{F}\,\textit{(raa)} \qquad \frac{\begin{array}{c}\lfloor F \rfloor \\ \nabla \\ [\,]\end{array}}{\neg F}\,(\neg I) \qquad \frac{F \qquad \neg F}{[\,]}\,(\neg E)$$

- **Conjunction**

$$\frac{F \qquad G}{(F \wedge G)}\,(\wedge I) \qquad \frac{(F \wedge G)}{F}\,(\wedge E) \qquad \frac{(F \wedge G)}{G}\,(\wedge E)$$

- **Disjunction**

$$\frac{F}{(F \vee G)}\,(\vee I) \qquad \frac{G}{(F \vee G)}\,(\vee I) \qquad \frac{(F \vee G) \qquad \begin{array}{c}\lfloor F \rfloor \\ \nabla \\ H\end{array} \qquad \begin{array}{c}\lfloor G \rfloor \\ \nabla \\ H\end{array}}{H}\,(\vee E)$$

Natural Deduction – Inference Rules (Continued)

- **Implication**

$$\frac{\begin{array}{c}\lfloor G \rfloor \\ \nabla \\ F\end{array}}{(G \rightarrow F)}\,(\rightarrow I) \qquad \frac{\begin{array}{c}\lfloor \neg F \rfloor \\ \nabla \\ \neg G\end{array}}{(G \rightarrow F)}\,(\rightarrow I) \qquad \frac{G \qquad (G \rightarrow F)}{F}\,(\rightarrow E)$$

- **Equivalence**

$$\frac{\begin{array}{c}\lfloor G \rfloor \\ \nabla \\ F\end{array} \qquad \begin{array}{c}\lfloor F \rfloor \\ \nabla \\ G\end{array}}{(G \leftrightarrow F)}\,(\leftrightarrow I) \qquad \frac{G \qquad (G \leftrightarrow F)}{F}\,(\leftrightarrow E) \qquad \frac{F \qquad (G \leftrightarrow F)}{G}\,(\leftrightarrow E)$$

Derivations (1)

- The set of derivations in the calculus of natural deduction is the smallest set $\mathcal{X}$ with the following properties

  1. $\mathcal{L}(R) \subseteq \mathcal{X}$.
  2. If

     $$\begin{array}{c} \nabla \\ H_1 \end{array}$$

     is a derivation in $\mathcal{X}$ and

     $$\frac{H_1}{H_2}$$

     is the instance of a rule $r \in \{(f),\ (\wedge E),\ (\vee I)\}$, then

     $$\frac{\begin{array}{c} \nabla \\ H_1 \end{array}}{H_2}\ r$$

     is a derivation in $\mathcal{X}$.

Derivations (2)

3 If

$$\begin{array}{c} H_1 \\ \bigtriangledown \\ H_2 \end{array}$$

is a derivation in $\mathcal{X}$, j is a new index, and

$$\dfrac{\begin{array}{c} \lfloor H_1 \rfloor \\ \bigtriangledown \\ H_2 \end{array}}{H_3}$$

is an instance of a rule $r \in \{$*(raa)*, $(\neg I)$, $(\rightarrow I)\}$, then

$$\dfrac{\begin{array}{c} \lfloor H_1 \rfloor^j \\ \bigtriangledown \\ H_2 \end{array}}{H_3}\, r^j$$

is a derivation in $\mathcal{X}$.
The hypothesis H_1 was cancelled by the application of the rule.

Derivations (3)

4 If

$$\begin{array}{c}\nabla_1\\ H_1\end{array} \quad , \quad \begin{array}{c}\nabla_2\\ H_2\end{array}$$

are derivations in $\mathcal{X}$ and if

$$\dfrac{\begin{array}{c}\nabla_1\\ H_1\end{array} \qquad \begin{array}{c}\nabla_2\\ H_2\end{array}}{H_3}$$

is an instance of a rule $r \in \{(\neg E),\ (\rightarrow E),\ (\leftrightarrow E)\}$, then

$$\dfrac{\begin{array}{c}\nabla_1\\ H_1\end{array} \qquad \begin{array}{c}\nabla_2\\ H_2\end{array}}{H_3}\ r$$

is a derivation in $\mathcal{X}$.

Derivations (4)

5 If

$$\begin{array}{ccccc} & & F & & G \\ \nabla_1 & & \nabla_2 & & \nabla_3 \\ (F \vee G) & , & H & , & H \end{array}$$

are derivations in $\mathcal{X}$ and j is a new index, then

$$\dfrac{\begin{array}{c} \nabla_1 \\ (F \vee G) \end{array} \quad \begin{array}{c} \lfloor F \rfloor^j \\ \nabla_2 \\ H \end{array} \quad \begin{array}{c} \lfloor G \rfloor^j \\ \nabla_3 \\ H \end{array}}{H} \, (\vee E)^j$$

is a derivation in $\mathcal{X}$.
The hyotheses F and G are cancelled by the application of $(\vee E)$.

Derivations (5)

6 If

$$\begin{array}{c} G \\ \nabla_1 \\ F \end{array} \quad , \quad \begin{array}{c} F \\ \nabla_2 \\ G \end{array}$$

are derivations in $\mathcal{X}$ and j is a new index, then

$$\dfrac{\begin{array}{c} \lfloor G \rfloor^j \\ \nabla_1 \\ F \end{array} \qquad \begin{array}{c} \lfloor F \rfloor^j \\ \nabla_2 \\ G \end{array}}{(G \leftrightarrow F)} \; (\leftrightarrow I)^j$$

is a derivation in $\mathcal{X}$.
The hypotheses G and F are cancelled by the application of $(\leftrightarrow I)$.

Proofs

- An element of $\mathcal{X}$ is called **derivation of F in the calculus of natural deduction** if in its tree representation the root node is labeled by F.
- A **proof of F in the calculus of natural deduction** is a derivation of F in the calculus of natural deduction, where each hypothesis is cancelled. This will be denoted by $\vdash_n F$.
- **A Simple Example** A proof of $(p \longrightarrow p)$

$$\frac{\lfloor p \rfloor^1}{(p \longrightarrow p)}\,(\longrightarrow I)^1$$

The Introductory Example Revisited

$$
\dfrac{\dfrac{\dfrac{\lfloor (p \vee (q \wedge r)) \rfloor^1 \quad \dfrac{\lfloor p \rfloor^2}{(p \vee q)}(\vee I) \quad \dfrac{\dfrac{\lfloor (q \wedge r) \rfloor^2}{q}(\wedge E)}{(p \vee q)}(\vee I)}{(p \vee q)}(\vee E)^2 \quad \dfrac{\lfloor (p \vee (q \wedge r)) \rfloor^1 \quad \dfrac{\lfloor p \rfloor^3}{(p \vee r)}(\vee I) \quad \dfrac{\dfrac{\lfloor (q \wedge r) \rfloor^3}{r}(\wedge E)}{(p \vee r)}(\vee I)}{(p \vee r)}(\vee E)^3}{((p \vee q) \wedge (p \vee r))}(\wedge I)}{((p \vee (q \wedge r)) \rightarrow ((p \vee q) \wedge (p \vee r)))}(\rightarrow I)^1
$$

▷ **Please compare this to the informal proof given at the beginning.**

Lemmas (1)

- Subderivations may occur over and over again, e.g.,

$$\dfrac{\dfrac{\neg\neg F \qquad \lfloor \neg F \rfloor^1}{[\,]}\,(\neg E)}{F}\,(raa)^1$$

- Idea Generate such derivations once and for all, specify corresponding inference rules called lemmas, and apply the lemmas in addition to the usual inference rules.
- In the example we obtain

$$\dfrac{\neg\neg F}{F}\,(l1)$$

Lemmas (2)

- **The lemma**

$$\frac{\neg(F \wedge G) \qquad F}{\neg G}\,(l2)$$

- **and its derivation**

$$\frac{\dfrac{\neg(F \wedge G) \qquad \dfrac{F \qquad \lfloor G \rfloor^1}{(F \wedge G)}\,(\wedge I)}{[\,]}\,(\neg E)}{\neg G}\,(\neg I)^1$$

Lemmas (3)

- **The lemma**

$$\frac{\neg(F \wedge G) \qquad G}{\neg F}\,(l3)$$

- **and its derivation**

$$\frac{\dfrac{\neg(F \wedge G) \qquad \dfrac{\lfloor F \rfloor^1 \qquad G}{(F \wedge G)}\,(\wedge I)}{[\,]}\,(\neg E)}{\neg F}\,(\neg I)^1$$

Lemmas (4)

- **The lemma**

$$\frac{\neg(F \vee G)}{(\neg F \wedge \neg G)}\,(l4)$$

- **and its derivation**

$$\dfrac{\dfrac{\dfrac{\neg(F \vee G) \qquad \dfrac{\lfloor F \rfloor^1}{(F \vee G)}\,(\vee I)}{[\,]}\,(\neg E)}{\neg F}\,(\neg I)^1 \qquad \dfrac{\dfrac{\neg(F \vee G) \qquad \dfrac{\lfloor G \rfloor^2}{(F \vee G)}\,(\vee I)}{[\,]}\,(\neg E)}{\neg G}\,(\neg I)^2}{(\neg F \wedge \neg G)}\,(\wedge I)$$

Another Example Demonstrating the Use of Lemmas

- **A proof of** $(p \vee \neg p)$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\lfloor \neg(p \vee \neg p) \rfloor^1}{(\neg p \wedge \neg\neg p)}\,(l4)}{\neg\neg p}\,(\wedge E)}{p}\,(l1) \qquad \dfrac{\dfrac{\lfloor \neg(p \vee \neg p) \rfloor^1}{(\neg p \wedge \neg\neg p)}\,(l4)}{\neg p}\,(\wedge E)}{[\,]}\,(\neg E)}{(p \vee \neg p)}\,(\textit{raa})^1
$$

Some Remarks on Natural Deduction

- Lemmas shorten proofs, but enlarge the search space.
- The calculus of natural deduction is sound and complete.
- Human readable proofs versus machine generated proofs.
- Sequent calculus.
- Proof theory.

Sequent Calculus

- A **sequent** is an expression of the form $\mathcal{F} \vdash \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are multisets of formulas.
- **Notation** We use the following abbreviations:

$$\begin{aligned} \dot{\{}F_1, \ldots, F_n\dot{\}} &\rightsquigarrow F_1, \ldots, F_n. \\ \mathcal{F} \dot{\cup} \dot{\{}G\dot{\}} &\rightsquigarrow \mathcal{F}, G. \end{aligned}$$

- The **inference rules** are of the form

$$\frac{S_1 \ldots S_n}{S}\,\mathsf{r}$$

or

$$\frac{}{S}\,\mathsf{r}$$

Sequent Calculus: Axiom, Cut and Structural Rules

Axiom

$$\frac{}{\boldsymbol{H}, \mathcal{F} \vdash \mathcal{G}, \boldsymbol{H}}\ \mathsf{ax}$$

Cut

$$\frac{\mathcal{F} \vdash \mathcal{G}, \boldsymbol{H} \quad \boldsymbol{H}, \mathcal{F} \vdash \mathcal{G}}{\mathcal{F} \vdash \mathcal{G}}\ \mathsf{cut}$$

Structural rules

$$\frac{\boldsymbol{H}, \boldsymbol{H}, \mathcal{F} \vdash \mathcal{G}}{\boldsymbol{H}, \mathcal{F} \vdash \mathcal{G}}\ \mathsf{cl}$$

$$\frac{\mathcal{F} \vdash \mathcal{G}, \boldsymbol{H}, \boldsymbol{H}}{\mathcal{F} \vdash \mathcal{G}, \boldsymbol{H}}\ \mathsf{cr}$$

Sequent Calculus: Logical Rules

Rules for the left hand side

$$\frac{\mathcal{F} \vdash \mathcal{G}, H_1 \quad H_2, \mathcal{F} \vdash \mathcal{G}}{(H_1 \rightarrow H_2), \mathcal{F} \vdash \mathcal{G}} \rightarrow\mathsf{l}$$

$$\frac{H_1, H_2, \mathcal{F} \vdash \mathcal{G}}{(H_1 \wedge H_2), \mathcal{F} \vdash \mathcal{G}} \wedge\mathsf{l}$$

$$\frac{H_1, \mathcal{F} \vdash \mathcal{G} \quad H_2, \mathcal{F} \vdash \mathcal{G}}{(H_1 \vee H_2), \mathcal{F} \vdash \mathcal{G}} \vee\mathsf{l}$$

$$\frac{\mathcal{F} \vdash \mathcal{G}, H}{\neg H, \mathcal{F} \vdash \mathcal{G}} \neg\mathsf{l}$$

Rules for the right hand side

$$\frac{H_1, \mathcal{F} \vdash \mathcal{G}, H_2}{\mathcal{F} \vdash \mathcal{G}, (H_1 \rightarrow H_2)} \rightarrow\mathsf{r}$$

$$\frac{\mathcal{F} \vdash \mathcal{G}, H_1 \quad \mathcal{F} \vdash \mathcal{G}, H_2}{\mathcal{F} \vdash \mathcal{G}, (H_1 \wedge H_2)} \wedge\mathsf{r}$$

$$\frac{\mathcal{F} \vdash \mathcal{G}, H_1, H_2}{\mathcal{F} \vdash \mathcal{G}, (H_1 \vee H_2)} \vee\mathsf{r}$$

$$\frac{H, \mathcal{F} \vdash \mathcal{G}}{\mathcal{F} \vdash \mathcal{G}, \neg H} \neg\mathsf{r}$$

Sequent Calculus: Soundness and Completeness

- A **proof** of the sequent S in the sequent calculus is defined as follows:
  - An axiom of the form $\dfrac{}{S}\,\mathsf{r}$ is a proof of S.
  - If $T_1, \ldots, T_n$ are proofs of $S_1, \ldots, S_n$ respectively and the sequent calculus contains a rule of the form $\dfrac{S_1 \ldots S_n}{S}\,\mathsf{r}$, then $\dfrac{T_1 \ldots T_n}{S}\,\mathsf{r}$ is a proof of S.
- The propositional sequent calculus is sound and complete.
- **Subformula property**: all formulas occurring in the condition of a rule occur as subformulas in the conclusion of the rule.
  - Only the cut rule violates this property.
  - **Gentzen's "Hauptsatz"**: Cuts can be removed in sequent calculus proofs.

3.5 Properties

- **Compactness Theorem**
- **Soundness and Completeness Theorems**

3.5.1 Compactness Theorem – Introduction

- How can we decide whether a possibly infinite set of formulas is satisfiable?
- Idea We test all finite subsets.
- Remember $\equiv$ is an equivalence relation on $\mathcal{L}(\mathcal{R})$.
- Let $\mathcal{L}(\mathcal{R}, n) \subseteq \mathcal{L}(\mathcal{R})$ be the set of formulas in which only the variables $p_1, \ldots, p_n$ occur.
- There are 2^n different interpretations for $\mathcal{L}(\mathcal{R}, n)$.
- How many different equivalence classes are defined by $\equiv$ on $\mathcal{L}(\mathcal{R}, n)$?
  - 2^{2^n}.

Compactness Theorem

- **Theorem 3.39 (Compactness Theorem)**
  A set $\mathcal{F}$ of propositional formulas is satisfiable
  iff every finite subset $\mathcal{G} \subseteq \mathcal{F}$ is satisfiable.
- **Proof** Let $\mathcal{F}$ be a (possibly infinite) set of formulas. We have to show:

  (a) If $\mathcal{F}$ is satisfiable, so is each finite $\mathcal{G} \subseteq \mathcal{F}$.

  (b) If each $\mathcal{G} \subseteq \mathcal{F}$ is satisfiable, so is $\mathcal{F}$.

  - (a) follows immediately from the definition of satisfiability for sets.
  - To show (b) we have to prove the existence of a model for $\mathcal{F}$ starting from the models for each finite $\mathcal{G} \subseteq \mathcal{F}$.

Proof of Part (b) of the Compactness Theorem

- Suppose, each finite $\mathcal{G} \subseteq \mathcal{F}$ is satisfiable.
- Let $\mathcal{G}_n = \mathcal{F} \cap \mathcal{L}(\mathcal{R}, n)$.
- There are at most $k \leq 2^{2^n}$ different equivalence classes on $\mathcal{G}_n$ defined by $\equiv$.
- Let $G_1, \ldots, G_k$ be representatives of these equivalence classes.
- Hence, for each $G \in \mathcal{G}_n$ there is an i, $1 \leq i \leq k$, with $G \equiv G_i$.
- Hence, each model for $\{G_1, \ldots, G_k\}$ is also a model for $\mathcal{G}_n$.
- Because $\{G_1, \ldots, G_k\}$ is a finite subset of $\mathcal{F}$, it has a model.
- Let I_n be this model.
- Because $\mathcal{G}_1 \subseteq \mathcal{G}_2 \subseteq \ldots \subseteq \mathcal{G}_n$, I_n is also a model for all $\mathcal{G}_j$, $1 \leq j \leq n$.

Proof of Part (b) of the Compactness Theorem (continued)

- We construct a model I for $\mathcal{F}$ as follows:

  Set $I := \emptyset$, $K_0 := \mathbb{N}$ and $n = 1$.

  Do the following:

  If there are infinitely many $j \in K_{n-1}$ with $[p_n]^{I_j} = \top$, then set

  $$I := I \cup \{p_n\} \text{ and } K_n := K_{n-1} \setminus \{j \mid [p_n]^{I_j} = \bot\}, \qquad (\top)$$

  else set

  $$K_n := K_{n-1} \setminus \{j \mid [p_n]^{I_j} = \top\}. \qquad (\bot)$$

  Set $n := n + 1$.

  Goto "Do the following".

- I is an interpretation.
- By induction the following proposition $E(n)$ can be proven:

  K_n contains infinitely many elements and
  for all $m \in K_n$ with $m > n$ we find: $[p_i]^{I_m} = [p_i]^I$ for all $1 \leq i \leq n$.

  Proof $\rightsquigarrow$ Exercise.

Proof of Part (b) of the Compactness Theorem (continued)

- To show: I is a model for $\mathcal{F}$.
- Let F be an arbitrary but fixed element of $\mathcal{F}$.
- There are at most finitely many propositional variables occurring in F.
- Let p_n be the variable with the highest index.
- Then, $F \in \mathcal{G}_n \subseteq \mathcal{G}_{n+1} \subseteq \ldots$
  and each interpretation $I_n,\ I_{n+1},\ \ldots$ is a model for F.
- Because of $E(n)$ we find:
  - K_n contains infinitely many elements.
  - For all $m \in K$ with $m > n$ we find: $[p_i]^{I_m} = [p_i]^I$ for all $1 \leq i \leq n$.
- Select $m \in K_n$ with $m > n$.
- Because I_m is a model for F and F contains only the variables $p_1,\ \ldots,\ p_n$, I must be a model for F as well.
- Because F is an arbitrary element from $\mathcal{F}$, I must also be a model for $\mathcal{F}$. qed

Implications of the Compactness Theorem

- The proof is not constructive.
- We may negate both sides of the compactness theorem.
- **Corollary 3.40** A set of propositional formulas is unsatisfiable **iff** one of its finite subsets is unsatisfiable.
- Procedure to prove unsatisfiability of $\mathcal{F}$:
  - We generate all finite subsets of $\mathcal{F}$ in a systematic way and test them for unsatisfiability.

3.5.2 Soundness and Completeness Theorems

- **Lemma 3.41 (Propositional Resolution Lemma)** Let $F = \langle C_1, \ldots, C_n \rangle$ be a propositional formula in clause form with the clauses C_i, $1 \leq i \leq n$, and let C be a resolvent of two clauses from F. Then $F \equiv (F \wedge C)$ holds.
- **Proof** To show $I \models (F \wedge C)$ iff $I \models F$.

$\Rightarrow$ immediate.

$\Leftarrow$ Suppose $I \models F$. Let C be the resolvent of C_1 and C_2 wrt A.

- Let $C_1' = C_1$ "without" A and $C_2' = C_2$ "without" $\neg A$.
- Then $C = (C_1' \vee C_2')$.
- **Case 1** $I \models A$.

  Then, $I \not\models \neg A$.

  Because $I \models C_2$ we conclude $I \models C_2'$.

  Hence, $I \models C$.
- **Case 2** $I \not\models A$.

  Because $I \models C_1$ we find $I \models C_1'$.

  Hence $I \models C$. qed

An Implication of the Resolution Lemma

- **Corollary 3.41(a)** Let $F = \langle C_1, \ldots, C_n \rangle$ be a propositional logic formula in clause form with clauses C_i, $1 \leq i \leq n$, and let $D_1, \ldots, D_m$ be the computed resolvents in a resolution derivation from F. Then, $F \equiv \langle F, D_1, \ldots, D_n \rangle$.
- **Proof** Induction over n $\rightsquigarrow$ **Exercise**.

Resolution Method

- **Theorem 3.42 (Soundness and Completeness of the Prop. Resolution Method)**
  Let F be a propositional formula. $\models F$ iff $\vdash_r F$.
- **Proof Soundness To show** If $\vdash_r F$, then $\models F$.
  - Suppose $\vdash_r F$.
  - By Definition 3.29 we find a resolution refutation for $\neg F$,
    i.e. by Definition 3.28, a resolution derivation for $\neg F$ containing $[\,]$.
  - By Theorem 3.23 we may assume wlog that $\neg F$ is in clause form.
  - Let $C_1, \ldots, C_n, [\,]$ be all resolvents computed in the refutation.
  - By the Corollary 3.41(a) we find

$$\neg F \equiv \langle \neg F, C_1, \ldots, C_n, [\,] \rangle.$$

  - Because $[\,]$ is unsatisfiable we conlude by Theorem 3.17 that $\neg F \equiv [\,]$,
    i.e. $\neg F$ ist unsatisfiable.
  - Applying Theorem 3.12 we find $\models F$.

Completeness of the Resolution Method

- **Proof Completeness** **To show** If $\models F$, then $\vdash_r F$.
  - Suppose $\models F$.
  - By Theorem 3.23 we find a formula G in clause form with $G \equiv \neg F$.
  - By Theorem 3.12 we find that G is unsatisfiable.
  - To show: there is a resolution refutation for G
  - The proof is by induction on the number n of propositional variables occurring in G.
- $n = 0$ Then $G = \langle [\,], \ldots, [\,] \rangle$ and we are done.
- $n \Rightarrow n + 1$ Assume that the proposition holds for n, i.e. if G is unsatisfiable and in G occur only the variables $p_1, \ldots, p_n$, then there is a resolution refutation for G.
  - Wlog we may assume that $p_1, \ldots, p_n$ are the n variables occurring in G $\rightsquigarrow$ Exercise.

Completeness of the Resolution Method (continued)

- Let G' be a formula in clause form in which the variables $p_1, \ldots, p_{n+1}$ occur.
  - Let $G'_\top$ be the formula obtained from G' by
    - deleting each clause, in which p_{n+1} occurs, and
    - deleting each occurrence of $\neg p_{n+1}$.
  - Let $G'_\bot$ be the formula obtained from G' by
    - deleting each clause, in which $\neg p_{n+1}$ occurs, and
    - deleting each occurrence of p_{n+1}.
  - We find: $G'_\top$ and $G'_\bot$ are unsatisfiable if G' is unsatisfiable
    $\rightsquigarrow$ Exercise.

Completeness of the Resolution Method (continued)

- Applying the induction assumption we find refutations for $G'_\bot$ and $G'_\top$.
- resolvents $G'_\top$ $\rightsquigarrow$ resolvents G'.
  - $\rightsquigarrow$ **Case 1** resolution refutation for G'.
  - $\rightsquigarrow$ **Case 2** resolution derivation with $[\neg p_{n+1}, \ldots \neg p_{n+1}]$ as last entry.
- resolvents $G'_\bot$ $\rightsquigarrow$ resolvents G'.
  - $\rightsquigarrow$ **Case 1** resolution refutation for G'.
  - $\rightsquigarrow$ **Case 2** resolution derivation with $[p_{n+1}, \ldots, p_{n+1}]$ as last entry.
- If case 2 was selected twice, then compute the resolvent of $[p_{n+1}, \ldots, p_{n+1}]$ and $[\neg p_{n+1}, \ldots, \neg p_{n+1}]$
  $\rightsquigarrow$ resolution refutation for G'.
- An application of the induction principle yields the desired result. qed

Satisfiability

- **Up to now** Does $\models F$ hold?
- **Now** Is F satisfiable?
  - **Goal** To find a model for F.
  - **Applications** optimization problems, planning problems, etc.
  - **Method** "greedy satisfiability testing" (gsat)

Some Definitions

- Let F be a formula in clause form.
- $C \in F$ iff clause C occurs in F.
- $F_u(I) = \{C \mid C \in F$ and $I \not\models C\}$.
- *flip*(I, p) denotes the interpretation obtained from I by reversing the truth value assigned to p.
- *rank*$(I, F, p) = |F_u(I)| - |F_u(\textit{flip}(I, p))|$ denotes the rank of F wrt I and p.

Satisfiability Testing

- `maxtries`, `maxloops` $\in \mathbb{N}$.
- `initial` **selects an arbitrary interpretation** I **for** F**.**
- `pick` **selects an arbitrary element from a set of propositional variables.**
- `flip` **implements** ***flip*****.**
- `rank` **impements** ***rank*****.**

A Generic Algorithm

`Input`: **A propositional formula** `F` **in clause form.**
`Output`: **A model for** `F`**, if one is found,**
or "`no model found`".

```
procedure gensat(F)
  for J := 1 to maxtries do
    I := initial(F);
    for K := 1 to maxloops do
      if I ⊨ F then return I;
      I := flip(I,pick(hillclimb(F,I)));
    end for;
  end for;
  return 'no model found';
end gensat;
```

Hillclimbing

`Input`: **A propositional formula** `F` **in clause form and an interpretation** `I`**.**
`Output`: **A set of variables occurring in** `F`**.**

```
procedure hillclimb(F,I);
  r := rnd(0,1);
  return {p | select(F,I,p,r) = ⊤};
end hillclimb;
```

gsat

$$\mathtt{select}(\mathtt{F},\mathtt{I},\mathtt{p},\mathtt{r}) = \begin{cases} \top & \text{if } \mathtt{rank}(\mathtt{F},\mathtt{I},\mathtt{p}) = \max\{\mathtt{rank}(\mathtt{F},\mathtt{I},\mathtt{p}') \mid \mathtt{p}' \in \mathcal{R}\}, \\ \bot & \text{otherwise.} \end{cases}$$

gwsat

$$\mathtt{select}(\mathtt{F},\mathtt{I},\mathtt{p},\mathtt{r}) = \begin{cases} \top & \text{if } \mathtt{r} \geq \mathtt{r}' \\ & \text{and } \mathtt{rank}(\mathtt{F},\mathtt{I},\mathtt{p}) = \max\{\mathtt{rank}(\mathtt{F},\mathtt{I},\mathtt{p}') \mid \mathtt{p}' \in \mathcal{R}\}, \\ \top & \text{if } \mathtt{r} < \mathtt{r}' \text{ and } \boldsymbol{p} \text{ occurs in } \boldsymbol{F_u}(\mathtt{I}), \\ \bot & \text{otherwise.} \end{cases}$$

where `r'` **is some fixed value in** $[0, 1]$**.**

Some Results

- **3-CNF formulas**
  (formulas in conjunctive normal form, each clause containing 3 literals).
- $\texttt{maxtries} = 1$, $\texttt{maxloops} = N \times 100$,
  where N is the number of variables occurring in F.

N	variant	forced	unforced
100	gsat	76.1%	23.5%
	gwsat	100%	82.2%
200	gsat	75.3%	8.4%
	gwsat	99%	48.7%
300	gsat	77.0%	2.7%
	gwsat	100%	26.3%
500	gsat	70.2%	—
	gwsat	100%	—

- $\texttt{maxtries} = 15: \quad 26.3\% \rightsquigarrow 98.8\%.$