Soundness and Completeness for Propositional Natural Deduction

Robert Keller
February 2014
Proof vs. Truth for Propositional Formulas

• We began by defining truth (wrt a valuation)

• We have also seen proof (using natural deduction)

• Now we want to connect these. Ideally:

  • Our proofs are proving only true statements. (soundness)

  • There is nothing lacking in our proof system. (completeness)
Soundness vs. Completeness

• Given a definition of truth, and a proof system,
  • The system is **sound** if every formula the system proves is true.
  • The system is **complete** if every true formula is provable.

• In order to make this more precise, we need a little more notation (surprise!).
Syntactic **Provability**: Single Turnstile

- Let $A_1, \ldots, A_n, B$ be formulas.
- Recall the meaning of

\[ A_1, \ldots, A_n \vdash B \]

Formula B is **derivable** from formulas $A_1, \ldots, A_n$ using the **proof rules** of our system.
Semantic **Entailment**: Double Turnstile

- Let $A_1, \ldots, A_n, B$ be formulas. The meaning of

$$A_1, \ldots, A_n \models B$$

(read $A_1, \ldots, A_n$ entails $B$) is:

For every **valuation** $\nu$ such that

if $\nu$ satisfies each of $A_1, \ldots, A_n$

then $\nu$ also satisfies $B$.

(If some $\nu$ fails to satisfy one of $A_1, \ldots, A_n$ it doesn’t matter.)

$A_1, \ldots, A_n \models B$ is said to be a **valid entailment** in this case.
Example of a Valid Entailment  |=

- Determine whether $p \lor q, \neg q \lor r \models p \lor r$
- One way is to examine at most 8 valuations $\nu$, one for each possible value of $\nu(p), \nu(q), \nu(r)$.

<table>
<thead>
<tr>
<th>$\nu(p)$</th>
<th>$\nu(q)$</th>
<th>$\nu(r)$</th>
<th>$\nu(p \lor q)$</th>
<th>$\nu(\neg q \lor r)$</th>
<th>LHS holds</th>
<th>RHS holds</th>
<th>$\nu(p \lor r)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>$\checkmark$</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td>$\checkmark$</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$\checkmark$</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>$\checkmark$</td>
<td>$\checkmark$</td>
<td>1</td>
</tr>
</tbody>
</table>
Alternative Example of $|=\$

- Determine whether or not $p \lor q, \neg q \lor r |= p \lor r$

- We could reason as follows:
  - $\nu(q) = 0$ or $1$.
    - If $\nu(q) = 0$, then all LHS holds iff $\nu(p) = 1$, and in that case $\nu(p \lor r) = 1$, i.e. RHS holds.
    - If $\nu(q) = 1$, then LHS holds iff $\nu(r) = 1$, and in that case $\nu(p \lor r) = 1$, i.e. RHS holds.
  - As RHS holds whenever all LHS holds, we have entailment.
Claim (prove for yourself)

- These statements are equivalent:
  - $A_1, \ldots, A_n \models B$
  - $\models (A_1 \land \ldots \land A_n) \rightarrow B$
  - $\models A_1 \rightarrow \ldots \rightarrow A_n \rightarrow B$ (grouping is to the right)
Validity and Tautology

• $\models B$ is the special case for $n = 0$, and we say $B$ is valid.

  Every valuation must induce 1 for $B$, because every valuation vacuously satisfies the LHS.

• In the propositional case, being valid and being a tautology are the same, but this is not true in predicate logic.

• $\vdash B$ is the special case for $n = 0$, meaning that $B$ is provable from the empty set of premises.
Validity and Provability

• $\models B$ is valid.

• $\vdash B$ is provable from the empty set of premises.
More general *Sets* of Formulas, Models

- Generally, $\Gamma$ is a (possibly-infinite) set of formulas.

- A valuation $\nu$ satisfies $\Gamma$

  iff $\nu$ satisfies *each* formula in $\Gamma$.

Such a valuation is also called a *model* for $\Gamma$. 
Validity and Provability

• More generally, $\Gamma$ is a (possibly-infinite) set of formulas.

• $\Gamma \models B$ means: Every valuation that satisfies $\Gamma$ (i.e. satisfies every formula in $\Gamma$) also satisfies $B$.

Put another way, every model for $\Gamma$ is also a model for \{B\}.

• $\Gamma \vdash B$ means $B$ is provable from formulas $\Gamma$. 
Special Cases

• $\Gamma \models \bot$ means: Every valuation that satisfies $\Gamma$ (i.e. satisfies every formula in $\Gamma$) also satisfies $\bot$.

  But no valuation satisfies $\bot$.

• Therefore $\Gamma \models \bot$ iff no valuation satisfies $\Gamma$.

• Therefore $\Gamma \models \bot$ iff $\Gamma$ is unsatisfiable.

• Therefore $\text{not}(\Gamma \models \bot)$ iff $\Gamma$ is satisfiable (meaning that some valuation satisfies $\Gamma$).
Consistency

- $\Gamma \vdash \bot$ means: $\bot$ is provable from formulas in $\Gamma$.

- If $\Gamma \vdash \bot$, we say $\Gamma$ is inconsistent.

- Otherwise, $\Gamma$ is consistent.

- Compare:
  - $\Gamma \models \bot$ means $\Gamma$ is unsatisfiable
  - $\Gamma \models \neg \bot$ means $\Gamma$ is inconsistent.
Soundness vs. Completeness of a Logical System (such as ND)

- **Soundness**: Every provable sequent is a valid entailment:

  (for every set $\Gamma$ and formula $B$):

  $$\Gamma \vdash B \implies \Gamma \models B$$

- **Completeness**: Every valid entailment is provable:

  (for every set $\Gamma$ and formula $B$):

  $$\Gamma \models B \implies \Gamma \vdash B$$
Proof of Soundness for ND

- **Soundness** says: Every sequent of Natural Deduction is an entailment:

  for every \( \Gamma, B \):

  \[
  \Gamma \vdash B \implies \Gamma \models B
  \]

- Assume that \( \Gamma \vdash B \), to show \( \Gamma \models B \).

- This will be by **structural induction** on the **proof tree** of \( B \) from formulas in \( \Gamma \).

- The next table summarizes the ND rules.
Structural Induction?

- Structural Induction generalizes the usual mathematical induction.

- It proves properties of all elements of a set where the set is defined inductively.
Inductive Definition Example

• **Binary Trees** over a set $S$.

• A single element of $S$ is a binary tree (called a “leaf”).

• If $T_1$ and $T_2$ are binary trees, so is the pair $(T_1, T_2)$ (called a “non-leaf”).
Binary Tree Example

- Let the set $S$ be \{a, b\}.
- Then the following are binary trees:
  - a
  - b
  - (a, a)
  - (a, b)
  - (b, a)
  - (b, b)
  - (a, (a, a))
  - (a, (a, b))
  - (a, (b, a))
  - (a, (b, b))
  - (b, (a, a)) etc.
Binary Tree Example

- Let the set S be \{a, b\}.
- Then the following are binary trees:
  - a
    - 1 leaf, 0 non-leaves
  - b
  - (a, a)
    - 2 leaves, 1 non-leaf
  - (a, b)
  - (b, a)
  - (b, b)
  - (a, (a, a))
    - 3 leaves, 2 non-leaves
  - (a, (a, b))
  - (a, (b, a))
  - (a, (b, b))
  - (b, (a, a))
    - etc.
Structural Induction Proof

- In any binary tree T:
  \[ \#\text{leaves}(T) = 1 + \#\text{non-leaves}(T) \]

- Basis: If T is a leaf, then there is 1 leaf and 0 non-leaves. So the equation holds.
Structural Induction Proof

- Induction Step: If $T$ is a non-leaf, then $T$ is $(T_1, T_2)$ for some $T_1$ and $T_2$.

- By the **induction hypothesis**
  \[ \#\text{leaves}(T_1) = 1 + \#\text{non-leaves}(T_1) \]
  \[ \#\text{leaves}(T_2) = 1 + \#\text{non-leaves}(T_2) \]

- The **inductive conclusion** is
  \[ \#\text{leaves}(T) = 1 + \#\text{non-leaves}(T) \]
Structural Induction Proof

But \#leaves(T) = \#leaves(T_1) + \#leaves(T_2) =

by the induction hypothesis

1 + \#non-leaves(T_1) + 1 + \#non-leaves(T_2) =

1 + (1 + \#non-leaves(T_1) + \#non-leaves(T_2)) =

1 + \#non-leaves(T), as T adds one non-leaf.

This is the desired inductive conclusion.
### Natural Deduction Rules with Contexts

<table>
<thead>
<tr>
<th>Introduction</th>
<th>Elimination</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>∧</strong></td>
<td><strong>∧</strong></td>
</tr>
</tbody>
</table>
| \[ \Gamma \vdash A \quad \Gamma \vdash B \]
| \[ \Gamma \vdash A \wedge B \] | \[ \Gamma \vdash A \wedge B \] |
| \[ \Gamma \vdash A \wedge B \] | \[ \Gamma \vdash A \wedge B \] | \[ \Gamma \vdash A \wedge B \] |
| **∨** | **∨** | **∨** |
| \[ \Gamma \vdash A \quad \Gamma \vdash B \]
| \[ \Gamma \vdash A \vee B \quad \Gamma \vdash A \vee B \] | \[ \Gamma \vdash A \vee B \] | \[ \Gamma \vdash A \vee B \] |
| \[ \Gamma \vdash A \vee B \] | \[ \Gamma \vdash A \vee B \] | \[ \Gamma \vdash A \vee B \] |
| **→** | **→** | **→** |
| \[ \Gamma \vdash A \rightarrow B \] | \[ \Gamma \vdash A \rightarrow B \] | \[ \Gamma \vdash A \rightarrow B \] |
| \[ \Gamma \vdash A \rightarrow B \] | \[ \Gamma \vdash A \rightarrow B \] | \[ \Gamma \vdash A \rightarrow B \] |
| **¬** | **¬** | **¬** |
| \[ \Gamma \vdash \lnot A \] | \[ \Gamma \vdash \lnot A \] | \[ \Gamma \vdash \lnot A \] |
| \[ \Gamma \vdash \lnot A \] | \[ \Gamma \vdash \lnot A \] | \[ \Gamma \vdash \lnot A \] |
| **RAA** | **RAA** | **RAA** |
| \[ \Gamma \vdash \lnot A \] | \[ \Gamma \vdash \lnot A \] | \[ \Gamma \vdash \lnot A \] |
| \[ \Gamma \vdash \perp \] | \[ \Gamma \vdash \perp \] | \[ \Gamma \vdash \perp \] |
| **Premise** | **Premise** | **Premise** |
| \[ \Gamma \vdash \{A\} \] | \[ \Gamma \vdash \{A\} \] | \[ \Gamma \vdash \{A\} \] |

Here \( \Gamma \) represents the **context** (set of formulas in force when rule is applied).
Proof of Soundness

- In order to establish soundness, we argue that each of the rules is valid.

- That is, in each rule, we can replace $\vdash$ with $|=\ $ and show that if the entailments above the line are valid, then so are the entailments below the line.
### Entailments Corresponding to Natural Deduction Rules

<table>
<thead>
<tr>
<th></th>
<th>Introduction</th>
<th>Elimination</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\land$</td>
<td>$\Gamma \models A$ $\Gamma \models B$ $\Gamma \models A \land B$</td>
<td>$\Gamma \models A \land B$ $\Gamma \models A$ $\Gamma \models B$</td>
</tr>
<tr>
<td>$\lor$</td>
<td>$\Gamma \models A$ $\Gamma \models B$ $\Gamma \models A \lor B$</td>
<td>$\Gamma \cup {A} \models C$ $\Gamma \cup {B} \models C$ $\Gamma \cup {A \lor B} \models C$</td>
</tr>
<tr>
<td>$\rightarrow$</td>
<td>$\Gamma \cup {A} \models B$ $\Gamma \models A \rightarrow B$</td>
<td>$\Gamma \models A$ $\Gamma \models A \rightarrow B$ $\Gamma \models B$</td>
</tr>
<tr>
<td>$\neg$</td>
<td>$\Gamma \cup {A} \models \bot$ $\Gamma \models \neg A$</td>
<td>$\Gamma \models A$ $\Gamma \models \neg A$ $\Gamma \models \bot$</td>
</tr>
<tr>
<td>RAA</td>
<td>$\Gamma \cup {\neg A} \models \bot$ $\Gamma \models \bot$</td>
<td>$\bot E$ $\Gamma \models \bot$ $\Gamma \models A$</td>
</tr>
<tr>
<td>Premise</td>
<td>$\Gamma \cup {A} \models A$</td>
<td></td>
</tr>
</tbody>
</table>

Here $\Gamma$ represents the context (set of formulas in force when rule is applied).
Basis for the Induction

The basis is the set of rules with nothing above the line. The only one is the

Premise rule $\Gamma \cup \{A\} \models A$,

which has to be shown with no assumptions.

But this is obvious, for if $\nu$ satisfies $\Gamma \cup \{A\}$ then it must satisfy $A$, which is one of its elements.
Example Proof Step for $\land I$

\[
\begin{array}{c}
\Gamma \models A \\
\Gamma \models B
\end{array}
\]
\[\Gamma \models A \land B\]

Assume $\Gamma \models A$ and $\Gamma \models B$, to show $\Gamma \models A \land B$.

Suppose $\nu$ satisfies $\Gamma$. Then by our assumption, $\nu(A) = 1$ and $\nu(B) = 1$. Therefore $\nu(A \land B) = 1$. 
Example Proof Step for $\lor$: 

\[ \Gamma \cup \{ A \} \models C \quad \Gamma \cup \{ B \} \models C \]
\[ \Gamma \cup \{ A \lor B \} \models C \]

Assume $\Gamma \cup \{ A \} \models C$ and $\Gamma \cup \{ B \} \models C$, to show $\Gamma \cup \{ A \lor B \} \models C$.

Suppose $\nu$ satisfies $\Gamma \cup \{ A \lor B \}$. Then $\nu$ satisfies $\Gamma$ and $A \lor B$. From the truth table for $\lor$, either $\nu(A) = 1$ or $\nu(B) = 1$. Then $\nu$ satisfies either $\Gamma \cup \{ A \}$ or $\Gamma \cup \{ B \}$. So by the assumption, satisfies $C$. 
Example Proof Step for →I

\[ \Gamma \cup \{A\} \models B \]
\[ \Gamma \models A \rightarrow B \]

Assume \( \Gamma \cup \{A\} \models B \), to show \( \Gamma \models A \rightarrow B \).

Suppose \( \nu \) satisfies \( \Gamma \), to show \( \nu \) satisfies \( A \rightarrow B \).

Either \( \nu(A) = 0 \) or \( \nu(A) = 1 \). If \( \nu(A) = 0 \), then
\( \nu(A \rightarrow B) = 1 \) by the truth table for \( \rightarrow \).

If \( \nu(A) = 1 \), then \( \nu \) satisfies \( \Gamma \cup \{A\} \), so by the assumption, \( \nu \) satisfies \( B \), i.e. \( \nu(B) = 1 \). So by the truth table \( \nu(A \rightarrow B) = 1 \) again.

Hence \( \nu \) satisfies \( A \rightarrow B \).
Exercise

Verify the remaining rules for yourself, completing the proof of soundness for natural deduction.
Uses of Soundness

- As we know, there are algorithms for determining whether or not
  \[ A_1, \ldots, A_n \models B \]

- Thus, one can compute a necessary condition of whether there is a proof of
  \[ A_1, \ldots, A_n \vdash B \]

- In other words, before embarking on trying to find a proof of a formula, we could check whether the formula follows on semantic grounds first.
Completeness

- Completeness says

\[
\Gamma \models B \text{ implies } \Gamma \vdash B
\]

- The general case (where \( \Gamma \) could be infinite) will require a “non-constructive” proof.

- The case of \( \Gamma \) finite is special, and admits a constructive, algorithmic, proof.
Finite Completeness

• Finite completeness says (for all $A_1, \ldots, A_n, B$)

$$A_1, \ldots, A_n \models B$$

implies

$$A_1, \ldots, A_n \vdash B$$

• If this could be established, then the algorithm mentioned for soundness would be a necessary and sufficient condition for the existence of a proof. Thus provability could be testable algorithmically.

• As our proof will use LEM, i.e. it applies to a classical rather than an intuitionistic system.
Proof of Finite Completeness

Three steps are used to show
\[ A_1, \ldots, A_n \models B \implies A_1, \ldots, A_n \vdash B : \]

1. \( A_1, \ldots, A_n \models B \implies \models (A_1 \rightarrow (A_2 \rightarrow \ldots (A_n \rightarrow B )) \ldots) \)

2. **For any formula C, \( \models C \implies \vdash C. \)**

   \[ [C \text{ could be } (A_1 \rightarrow (A_2 \rightarrow \ldots (A_n \rightarrow B )) \ldots), \text{ for example.}] \]

3. \( \vdash (A_1 \rightarrow (A_2 \rightarrow \ldots (A_n \rightarrow B )) \ldots) \implies A_1, \ldots, A_n \vdash B \)

**Step 2 is the key one**, as only it bridges the gap between \( \models \) and \( \vdash \). The other two are simplifying steps, showing that we don’t need to worry about the LHS of the turnstiles.

You can prove 1 and 3 by induction on n.
Proof that for all $C$

$$\models C \text{ implies } \vdash C$$

• Assume $\models C$. Let $p_1, p_2, \ldots, p_k$ be the set of all proposition symbols that occur in $C$.

• **For each combination** of proposition symbols with and without negation, we show that there is a sequent with that combination on the left and the formula of interest on the right:
  • $p_1, p_2, \ldots, p_k \vdash C$
  • $\neg p_1, p_2, \ldots, p_k \vdash C$
  • $p_1, \neg p_2, \ldots, p_k \vdash C$
  • $\neg p_1, \neg p_2, \ldots, p_k \vdash C$
  • etc.

• Those sequents will be combined into a single sequent of the required form using LEM and $\lor E$. 
The Combination Process

- Because this constructs a derivation that is of length exponential in \( k \), we will show it by example, for \( k = 2 \).

- Given that we have:

  - \( p_1, p_2 \mid \vdash C \)
  - \( \neg p_1, p_2 \mid \vdash C \)
  - \( p_1, \neg p_2 \mid \vdash C \)
  - \( \neg p_1, \neg p_2 \mid \vdash C \)

- The proof constructed for the single sequent is shown on the next page.
Proof Constructed for the Single Sequent

1. \( p_1 \lor \neg p_1 \quad \text{LEM} \)
2. \( \vdash  p_1 \quad \text{Assumption} \)
3. \( p_2 \lor \neg p_2 \quad \text{LEM} \)
4. \( \vdash  p_2 \quad \text{Assumption} \)
   \[ \ldots \text{steps in the proof of } p_1, p_2 \vdash C \]
5. \( C \)
6. \( \vdash  \neg p_2 \quad \text{Assumption} \)
   \[ \ldots \text{steps in the proof of } p_1, \neg p_2 \vdash C \]
7. \( C \)
8. \( \vdash  C \quad \text{vE } 3, 4-5, 6-7 \)
9. \( \vdash  \neg p_1 \quad \text{Assumption} \)
10. \( \vdash  p_2 \lor \neg p_2 \quad \text{LEM} \)
11. \( \vdash  p_2 \quad \text{Assumption} \)
   \[ \ldots \text{steps in the proof of } \neg p_1, p_2 \vdash C \]
12. \( C \)
13. \( \vdash  \neg p_2 \quad \text{Assumption} \)
   \[ \ldots \text{steps in the proof of } \neg p_1, \neg p_2 \vdash C \]
14. \( C \)
15. \( \vdash  C \quad \text{vE } 10, 11-12, 13-14 \)
16. \( \vdash  C \quad \text{vE } 1, 2-8, 9-15 \)
Proofs for the Individual Sequents

We want to show that for any formula $C$, if $| = C$ then each of the individual sequents below has a proof:

- $p_1, p_2, \ldots, p_k \vdash C$
- $\neg p_1, p_2, \ldots, p_k \vdash C$
- $p_1, \neg p_2, \ldots, p_k \vdash C$
- $\neg p_1, \neg p_2, \ldots, p_k \vdash C$

etc. (for every combination of symbols and their negations)

where $p_1, p_2, \ldots, p_k$ are the proposition symbols in $C$.

Approach: Use structural induction on the structure of the formula $C$ (rather than on the proof tree as before).
Correspondence between a collection of literals and a valuation

- Let \( \{p^*_1, p^*_2, \ldots, p^*_k\} \) be a set of literals containing each variable in the formula of interest in either negated or unnegated form.

- By the corresponding valuation, we mean the valuation that makes each literal 1.
Example

• Formula:
  \[(p \lor \neg q) \rightarrow r\]

• A relevant set of literals is
  \[\{\neg p, q, \neg r\} \].

• The corresponding valuation \(\nu\) is
  \[\nu(p) = 0, \nu(q) = 1, \nu(r) = 0.\]
Proofs for the Individual Sequents

The key step in the induction is this

**Lemma**: Let $C$ be any formula. For any set of literals $p^*_1, p^*_2, \ldots, p^*_k$ containing all variables in $C$, let $\nu$ be the corresponding valuation as defined previously. Then

$$A(C): \text{If } \nu(C) = 1, \text{ then } p^*_1, p^*_2, \ldots, p^*_k \vdash C.$$  

$$\quad \text{else } p^*_1, p^*_2, \ldots, p^*_k \vdash (\neg C).$$

This is done by **structural induction** on the **structure** of the **formula** $C$. 
Proving \( A(C) \): If \( \nu(C) = 1 \) then \( p^*_1, p^*_2, \ldots, p^*_k \models C. \)

else \( p^*_1, p^*_2, \ldots, p^*_k \models (\neg C). \)

- This is done by **structural induction** on the **structure** of the formula \( C \).

**Basis:** If \( C \) is a **single proposition symbol** \( p \), then:

- If \( \nu(p) = 1 \), then \( p^* \) must be \( p \), and we certainly have \( p \models p \) (case A).
- If \( \nu(p) = 0 \), then \( p^* \) must be \( \neg p \), and we have \( \neg p \models (\neg p) \) (case B).

- If \( C \) is the proposition symbol \( \perp \), then \( \nu(\perp) = 0 \) always, but also \( \models \neg \perp \) (using \( \neg I \)) (case B).
**Proving** \( A(C) \): If \( \nu(C) = 1 \) then \( p^*_1, p^*_2, \ldots, p^*_k \models C \).

else \( p^*_1, p^*_2, \ldots, p^*_k \models (\neg C) \).

- **Induction Step**: We have to show that the inductive hypothesis that property \( A \) for sub-formulas implies \( A \) for the overall formula, for each operator: \( \neg \land \lor \rightarrow \) forming \( C \) at the top level.

- This involves a case analysis for each operator.
Case where C is of form D \land E:

- Suppose \( \nu(C) = 0 \). Then we must have \( \nu(D) = 0 \) or \( \nu(E) = 0 \). WLOG, assume the first and appeal to symmetry.

- By the induction hypothesis, we have proof
  \[
  p^*_1, p^*_2, \ldots, p^*_k \vdash \neg D
  \]

- We want a proof
  \[
  p^*_1, p^*_2, \ldots, p^*_k \vdash \neg(D \land E)
  \]

Construct one by extending the given one as follows (in tree form):

\[
\begin{array}{c}
[D \land E]_1 \\
\neg D & D \land E \\
\bot \\
\neg E \\
\neg(D \land E) & \neg I_1
\end{array}
\]
Case where $C$ is of form $D \land E$:

- Suppose $\nu(C) = 1$. Then we must have $\nu(D) = \nu(E) = 1$.

- By the induction hypothesis, we have proof
  
  $\begin{align*}
  p^*_1, p^*_2, \ldots, p^*_k &\vdash D \\
  p^*_1, p^*_2, \ldots, p^*_k &\vdash E
  \end{align*}$

- We want a proof
  
  $\begin{align*}
  p^*_1, p^*_2, \ldots, p^*_k &\vdash D \land E
  \end{align*}$

How would you construct one?
Case where C is of form \( \neg D \) for some D:

- **Case 1:** \( \nu(C) = 0 \)
  - Then \( \nu(D) = 1 \).
  - By the induction hypothesis:
    \[ p^*_1, p^*_2, \ldots, p^*_k \vdash D, \]
  
  How to get a proof
  \[ p^*_1, p^*_2, \ldots, p^*_k \vdash \neg C? \]
  
  (Recall that C is \( \neg D \), so we need a proof of \( \neg \neg D \)).
Case where $C$ is of form $\neg D$ for some $D$

- **Case 2**: $\nu(C) = 1$
- Then $\nu(D) = 0$.
- By the induction hypothesis:
  
  \[
  p^*_1, p^*_2, \ldots, p^*_k \vdash (\neg D),
  \]
  but that is the same
  
  \[
  p^*_1, p^*_2, \ldots, p^*_k \vdash C
  \]
Exercise

- Prove the remaining cases needed to establish finite completeness.
Algorithm-Based Proof

• The proof just outlined is sufficiently constructive that we can create an algorithm from it:

• Given a valid formula C, generate a natural deduction proof of C.

• In some sense, such an algorithmic proof is useful, in that it can be live-tested by computer for various examples, unlike an ordinary proof.
Example of Algorithmically-Generated Proof by prover.pro

DNE: \( \neg\neg p \rightarrow p \)

?- testTautology(implies(not(not(p)), p)).

Proof for tautology: implies(not(not(p)), p):

<table>
<thead>
<tr>
<th>1: or(p, not(p)) [lem]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2: p [assumption(or-elim)]</td>
</tr>
<tr>
<td>3: implies(not(not(p)), p) [implies-intro(2)]</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>4: not(p) [assumption(or-elim)]</td>
</tr>
<tr>
<td>5: not(not(not(p))) [not-not-intro(4)]</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>6: not(not(p)) [assumption(implies-intro)]</td>
</tr>
<tr>
<td>7: bottom [not-elim(5, 6)]</td>
</tr>
<tr>
<td>8: p [bottom(7)]</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>9: implies(not(not(p)), p) [implies-intro(6-8)]</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>10: implies(not(not(p)), p) [or-elim(1, 2-3, 4-9)]</td>
</tr>
</tbody>
</table>
Example of Algorithmically-Generated Proof by prover.pro

**Peirce’s law:** \(((p \rightarrow q) \rightarrow p) \rightarrow p\)
(can be proved by a human using RAA rather than LEM in 12 steps)

Proof for tautology: \(\text{implies(implies(implies(p, q), p), p)}\):

<table>
<thead>
<tr>
<th>1: or(p, not(p)) [lem]</th>
</tr>
</thead>
<tbody>
<tr>
<td>2: p [assumption(or-el)]</td>
</tr>
<tr>
<td>3: or(q, not(q)) [lem]</td>
</tr>
<tr>
<td>4: q [assumption(or-el)]</td>
</tr>
<tr>
<td>5: implies(implies(p, q), p, p) [implies-intro(2)]</td>
</tr>
<tr>
<td>6: not(q) [assumption(or-el)]</td>
</tr>
<tr>
<td>7: implies(implies(p, q), p, p) [implies-intro(2)]</td>
</tr>
<tr>
<td>8: implies(implies(p, q), p, p) [or-el(3, 4-5, 6-7)]</td>
</tr>
</tbody>
</table>

| 9: not(p) [assumption(or-el)] |
| 10: or(q, not(q)) [lem] |
| 11: q [assumption(or-el)] |
| 12: p [assumption(implies-intro)] |
| 13: bottom [not-el(9, 12)] |
| 14: q [bottom(13)] |
| 15: implies(p, q) [implies-intro(12-14)] |
| 16: implies(implies(p, q), p) [assumption(not-intro)] |
| 17: p [implies-elim(15, 16)] |
| 18: bottom [not-el(9, 17)] |
| 19: not(implies(implies(p, q), p)) [not-intro(16-18)] |

| 20: implies(implies(p, q), p) [assumption(implies-intro)] |
| 21: bottom [not-el(19, 20)] |
| 22: p [bottom(21)] |
| 23: implies(implies(implies(p, q), p), p) [implies-intro(20-22)] |
| 24: not(q) [assumption(or-el)] |
| 25: p [assumption(implies-intro)] |
| 26: bottom [not-el(9, 25)] |
| 27: q [bottom(26)] |
| 28: implies(p, q) [implies-intro(25-27)] |
| 29: implies(implies(p, q), p) [assumption(not-intro)] |
| 30: p [implies-elim(28, 29)] |
| 31: bottom [not-el(9, 30)] |
| 32: not(implies(implies(p, q), p)) [not-intro(29-31)] |
| 33: implies(implies(p, q), p) [assumption(implies-intro)] |
| 34: bottom [not-el(32, 33)] |
| 35: p [bottom(34)] |
| 36: implies(implies(implies(p, q), p), p) [implies-intro(33-35)] |
| 37: implies(implies(implies(p, q), p), p) [or-el(10, 11-23, 24-36)] |
| 38: implies(implies(implies(p, q), p), p) [or-el(1, 2-8, 9-37)] |
Sketch of Completeness for the General (not-necessarily finite) Propositional Case

• This sketch follows van Dalen, *Logic and Structure*.

• **Definition**: A set of formulas \( \Gamma \) is **consistent** provided
  \[
  \text{not } \Gamma \vdash \bot.
  \]

• Note the parallel:
  
  • **Consistency** of \( \Gamma \): Not \( \Gamma \vdash \bot \).
  
  • **Satisfiability** of \( \Gamma \): Not \( \Gamma \models \bot \).
Lemma A

• For any $\Gamma$, $B$

\[ \Gamma \models B \quad \text{iff} \quad \Gamma \cup \{\neg B\} \models \bot \]

• Proof: Homework
Lemma B

• For any $\Gamma$, $B$
  
  $\Gamma \models B$ iff $\Gamma \cup \{\neg B\} \models \bot$.

• Proof: Homework
Lemma C

The following are equivalent:

a) Completeness.

b) For all $\Gamma, \varphi$, $\Gamma \models \varphi$ implies $\Gamma \not\vdash \varphi$.

c) For all $\Gamma$, $\Gamma \models \bot$ implies $\Gamma \not\vdash \bot$.

d) For all $\Gamma$, not $\Gamma \not\vdash \bot$ implies not $\Gamma \models \bot$.

e) For all $\Gamma$, if $\Gamma$ is consistent then $\Gamma$ has a model (i.e. $\Gamma$ is satisfiable by some valuation).

Proof:

• (b) restates (a).

• (c) iff (b) is by Lemmas A and B.

• (d) is the contrapositive of (c).

• (e) is a restatement of (d).
General Completeness Theorem

• We have shown that completeness is equivalent to saying:

• Every consistent set of formulas has a model.

• Sketch of the proof of the above statement:

  Start with a $\Gamma_0$ that is consistent, then show there exists a model for $\Gamma_0$, based on ND rules.
Sketch, continued

- First extend $\Gamma_0$ to a **maximally consistent** set $\Gamma_{\text{max}}$:
  - Let $A_0, A_1, A_2, \ldots$ be an enumeration of every possible propositional formula and define sets $\Gamma_0, \Gamma_1, \Gamma_2, \ldots$ as follows:
    - If $\Gamma_i \cup \{A_i\}$ is consistent, $\Gamma_{i+1}$ is defined as $\Gamma_i \cup \{A_i\}$.
    - Otherwise $\Gamma_{i+1}$ is defined as $\Gamma_i$.
  - (The Axiom of Choice is being used here.)
  - The **limit** of this process is $\bigcup \{\Gamma_0, \Gamma_1, \Gamma_2, \ldots \} = \Gamma_{\text{max}}$.
  - Then show that $\Gamma_{\text{max}}$ is consistent, and in fact, maximally consistent.
Sketch, continued

- \( \Gamma_{\text{max}} \) is **consistent**, because at no step is a formula added that would destroy its consistency.

- It is **maximally** consistent because it can be shown to be **closed under derivability**:
  
  If \( \Gamma_{\text{max}} \models \varphi \), then in fact \( \varphi \in \Gamma_{\text{max}} \).

- Any maximally consistent set has a model \( \nu \) as follows:
  
  - For each proposition symbol \( p \), if \( p \in \Gamma_{\text{max}} \) then \( \nu(p) = 1 \), otherwise \( \nu(p) = 0 \).

- Then argue that \( \nu \) **satisfies** \( \Gamma_{\text{max}} \) using closure under derivability (using a soundness-like argument).

- Finally, \( \nu \) also satisfies \( \Gamma_0 \), since \( \Gamma_0 \subseteq \Gamma_{\text{max}} \). So \( \Gamma_0 \) has a model.