Throughout this document we are concerned with classical, rather than constructive, propositional logic. That is, the LEM (Law of Excluded Middle) is assumed to hold. However, our meta-proof will be constructive to the extent that it is algorithmically implementable. However, it is not guaranteed to construct the shortest proof.

Meta-Theorem: For any formula \( A \), if \( \models A \) then also \( \not\models A \) in natural deduction.

In words, if \( A \) is a tautology, then there is a natural deduction proof of \( A \).

Proof: We first observe the following, without loss of generality:

We can restrict \( A \) so that the outermost connective, if there is one, is either \( \land \) or \( \neg \).

To justify this, suppose that the outermost connective is \( \lor \), i.e. \( A \) is \( B \lor C \). If there is a derivation of \( \neg(\neg B \land \neg C) \), which has \( \neg \) outermost, then there is a derivation of \( B \lor C \), because \( \neg(\neg B \land \neg C) \not\models B \lor C \), a lemma that we have proved.

On the other hand, suppose that the outermost connective is \( \rightarrow \), i.e. \( A \) is \( B \rightarrow C \). If there is a derivation of \( \neg(B \land \neg C) \), which has \( \neg \) outermost, then there is a derivation of \( B \rightarrow C \), as \( \neg(B \land \neg C) \not\models B \rightarrow C \), a lemma that is also easily proved.

Finally, if the outermost connective is \( \leftrightarrow \), i.e. \( A \) is \( B \leftrightarrow C \), then as we know, this may be regarded as an abbreviation for \( (B \rightarrow C) \land (C \rightarrow B) \), which clearly has \( \land \) outermost.

So the choices of \( A \) that are of concern are now limited to:

Base cases: Formulas with no connectives:
- \( \bot \)
- \( T \)
- \( p \), where \( p \) is a proposition symbol

Inductive cases: Formulas with at least one connective, one of these forms:
- \( B \land C \)
- \( \neg B \)

Obviously, in the first group, only \( T \) can be a tautology, but we retain the others for reasons to be seen.

Recall that a valuation \( \nu \) for a propositional formula \( A \) is an assignment of a truth value (0 or 1) to each proposition symbol in \( A \). For example, in

\[ (p \rightarrow q) \lor (p \rightarrow q) \]

there are 4 possible valuations \( \nu \):
- \( \nu(p, q) = (0, 0); \quad \nu(p, q) = (0, 1); \quad \nu(p, q) = (1, 0); \quad \text{and} \quad \nu(p, q) = (1, 1). \]
We have also seen that each valuation induces a truth value for the entire formula. A tautology is one that always has the induced value 1 regardless of valuation.

**Definition:** Let \( \nu \) be a valuation. By \( \nu^* \) we mean a set of literals (proposition symbols or their negations) for which \( \nu \) induces 1 in each literal. For example, if \( \nu(p, q, r) = (1, 0, 1) \), then \( \nu^* = \{p, \neg q, r\} \).

To prove the completeness theorem, we are first going to construct, for any formula \( A \) (not necessarily a tautology) and any valuation \( \nu \) for \( A \), one of the following cases:

<table>
<thead>
<tr>
<th>Case</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case +</td>
<td>( \nu(A) = 1 ), then there is a proof ( \nu^* \vdash \neg A ).</td>
</tr>
<tr>
<td>Case -</td>
<td>( \nu(A) = 0 ), then there is a proof ( \nu^* \vdash \neg \neg A ).</td>
</tr>
</tbody>
</table>

Case + gives us the main result, but we need Case - for **structural induction** on the form of \( A \). We will effectively deconstruct \( A \), and in the process construct the desired proof.

**Base cases:** Formulas with no connectives:

- \( \bot \): \( \nu(\bot) \) is required to be 0, and there is a proof \( \nu^* \vdash \neg \bot \). (Case -)
- \( T \): \( \nu(T) \) is required to be 1, and there is a proof \( \nu^* \vdash \neg T \). (Case +)

\( p \), where \( p \) is a proposition symbol:

- \( 0^- \): If \( \nu(p) = 0 \), then \( \neg p \in \nu^* \), so there is a proof \( \nu^* \vdash \neg p \).
- \( 0^+ \): If \( \nu(p) = 1 \), then \( p \in \nu^* \), so there is a proof \( \nu^* \vdash p \).

**Inductive cases:** Formulas with at least one connective, one of these forms:

- \( 1^- \): If \( \nu(B \land C) = 0 \), then either \( \nu(B) = 0 \) or \( \nu(C) = 0 \). These are symmetric, so assume \( \nu(B) = 0 \) (case -). By the induction hypothesis, \( \nu^* \vdash \neg B \), which, when extended with \( \neg B \vdash \neg (B \land C) \), gives \( \nu^* \vdash \neg (B \land C) \).

- \( 1^+ \): If \( \nu(B \land C) = 1 \), then both \( \nu(B) = \nu(C) = 1 \). By the induction hypothesis, \( \nu^* \vdash \neg B \) and \( \nu^* \vdash \neg C \) (case +), concatenate these proofs and extend to a proof \( \nu^* \vdash \neg (B \land C) \), by adding one more \( \land \)Introduction step.

- \( 2^- \): If \( \nu(\neg B) = 0 \), then \( \nu(B) = 1 \) (case +). By the induction hypothesis, \( \nu^* \vdash \neg B \), which when extended with \( B \vdash \neg \neg B \), gives \( \nu^* \vdash \neg (\neg B) \).

- \( 2^+ \): If \( \nu(\neg B) = 1 \), then \( \nu(B) = 0 \) (case -). By the induction hypothesis, \( \nu^* \vdash \neg B \), which is exactly what is desired for this case (with no additional step).

**Example:** Suppose \( A \) is \( (p \rightarrow q) \lor (p \rightarrow q) \). We can re-express \( A \) using \( \land \) and \( \neg \) as \( \neg (p \rightarrow q) \land \neg (p \rightarrow q) \) and, looking ahead, as \( \neg ((p \land \neg q) \land (q \land \neg p)) \).

Consider one valuation: \( \nu(p, q) = (1, 0) \). \( \nu^* \) is therefore \( \{p, \neg q\} \).
We have \( v(\neg((p \land \neg q) \land (q \land \neg p))) = 1 \) (case 2+) thus \( v(\neg((p \land q) \land (q \land \neg p))) = 0 \) (case 1-). Here \( v(p \land \neg q) = 1 \), and \( v(q \land \neg p) = 0 \), so we want a proof for the second case: \( v^* \models \neg(q \land \neg p) \).

Since \( v(\neg(q \land \neg p)) = 1 \) (case 2+), then \( v(q \land \neg p) = 0 \) (case 1-).

As both \( v(q) = 0 \) and \( v(\neg p) = 0 \), choose the first one. Then we must prove \( v^* \models \neg q \) (case 0-). This is a base case, and as \( v^* \) includes \( \neg q \), we are done.

Now reconstruct the proof thus derived: \( p, \neg q \models \neg((p \land q) \land (q \land \neg p)) \) working backward:

1. \( p \) Premise (not used)
2. \( \neg q \) Premise
3. \( \neg(q \land \neg p) \) 2, Lemma \( \neg A \models \neg(\neg A \land B) \)
4. \( \neg((p \land q) \land (q \land \neg p)) \) 3, Lemma \( \neg B \models \neg(\neg A \land B) \)

In case it isn’t clear, here is the proof of the first of these lemmas:

1. \( \neg A \) Premise
2. \( A \land B \) Assumption
3. \( A \) 2, \( \land E \)
4. \( \bot \) 1, 3, \( \neg E \)
5. \( \neg(A \land B) \) 2-4, \( \neg I \)

**Continuation of the main proof:** So far, we have shown that for any formula \( A \) and any valuation:

If \( v(A) = 1 \), then there is a proof \( v^* \models \neg A \).

When \( A \) is a tautology, we know that regardless of \( v \), \( v(A) = 1 \), so we have a proof of \( v^* \models \neg A \). But because this proof has premises \( v^* \), it does not suffice as a proof of \( \neg A \). In order to get the latter, we consider all valuations for \( A \). If \( A \) has \( n \) distinct proposition symbols, then there are \( 2^n \) such valuations \( v \), and thus \( 2^n \) proofs \( v^* \models \neg A \).

The remainder of the proof is then to “glue together” each of the \( 2^n \) proofs as subproofs in a larger proof having no premises. This gluing will use two companion constructions: LEM and \( \lor \) Elimination.

This time we use induction on the number \( N \) of proposition symbols in \( A \) to show that said gluing is possible.

**Basis:** If \( N = 0 \), there is no gluing to be done. The single proof of \( v^* \models \neg A \) (where \( v^* \) is thus empty), suffices. (Such an \( A \) might be \( \bot \lor \top \), for example.)
**Induction Step:** Suppose \( N \geq 1 \), and let \( p \) be a proposition symbol in \( A \). By the induction hypothesis, there are proofs \( p \vdash A \) and \( \neg p \vdash A \) each obtained by gluing together \( N-1 \) proofs corresponding to the remaining \( N-1 \) proposition symbols, with \( p \) and \( \neg p \) as respective assumptions.

We combine these proofs as follows:

\[
\begin{array}{c|c}
\hline
& \text{LEM} \\
\hline
p \lor \neg p & \\
\hline
\text{Proof of } p \vdash A & \\
\hline
\text{Proof of } \neg p \vdash A & \\
\hline
A & \lor \text{Elimination}
\end{array}
\]

Our natural deduction proof of tautology \( A \) is thus constructed, and this proves the meta-theorem.

**Example:** Again consider \( A \) as \( \neg((p \land \neg q) \land (q \land \neg p)) \). We already showed how to prove one of the four sub-proofs:

\[
p, \neg q \vdash \neg((p \land \neg q) \land (q \land \neg p)) \quad \text{case } \upsilon(p, q) = (1, 0)
\]

\[
\begin{array}{l|l}
1. & \text{Premise} \\
2. & \text{Premise} \\
3. & \text{Lemma } \neg A \vdash \neg(A \land B) \\
4. & \text{Lemma } \neg B \vdash \neg(A \land B)
\end{array}
\]

By symmetry, we can see a second sub-proof for a different valuation:

\[
\neg p, q \vdash \neg((p \land \neg q) \land (q \land \neg p)) \quad \text{case } \upsilon(p, q) = (0, 1)
\]

\[
\begin{array}{l|l}
1. & \text{Premise} \\
2. & \text{Premise} \\
3. & \text{Lemma } \neg A \vdash \neg(A \land B) \\
4. & \text{Lemma } \neg A \vdash \neg(A \land B)
\end{array}
\]

There are two more sub-proofs:

\[
p, q \vdash \neg((p \land \neg q) \land (q \land \neg p)) \quad \text{case } \upsilon(p, q) = (1, 1)
\]

\[
\upsilon(\neg((p \land \neg q) \land (q \land \neg p))) = 1 \quad \text{(2+)} \text{ so } \upsilon((p \land \neg q) \land (q \land \neg p)) = 0 \quad \text{(1-)} \text{, and } \upsilon(p \land \neg q)) = 0,
\]

so we need a proof of \( p, q \vdash \neg p \) or of \( p, q \vdash \neg(\neg q) \). We have only the latter.
So this subproof is:

1. p  Premise (not used)
2. q  Premise
3. \neg(\neg q)  2, Lemma A \vdash \neg(\neg A)
4. \neg(p \land \neg q)  3, Lemma \neg B \vdash \neg(A \land B)
5. \neg((p \land \neg q) \land (q \land \neg p))  4, Lemma \neg A \vdash \neg(A \land B)

There is one final sub-proof:

\neg p, \neg q \vdash \neg ((p \land \neg q) \land (q \land \neg p))  case \nu(p, q) = (0, 0)

\nu((p \land \neg q) \land (q \land \neg p)) = 1 (2+) so \nu((p \land \neg q) \land (q \land \neg p)) = 0 (1-), and \nu(p \land \neg q) = 0,
calling for a proof of \neg p, \neg q \vdash \neg p or of \neg p, \neg q \vdash \neg(\neg q). We have only the former.

So this final subproof is:

1. \neg p  Premise
2. \neg q  Premise (not used)
3. \neg(p \land \neg q)  1, Lemma \neg A \vdash \neg(A \land B)
4. \neg((p \land \neg q) \land (q \land \neg p))  3, Lemma \neg A \vdash \neg(A \land B)

Now we glue the subproofs together in the prescribed manner:

1. p \lor \neg p  LEM
2. p  Assumption (not used)
3. \neg q  LEM
4. \neg(\neg q)  4, Lemma A \vdash \neg(\neg A)
5. \neg(p \land \neg q)  5, Lemma \neg B \vdash \neg(A \land B)
6. \neg((p \land \neg q) \land (q \land \neg p))  6, Lemma \neg A \vdash \neg(A \land B)
7. \neg q  Assumption
8. \neg(q \land \neg p)  8, Lemma \neg A \vdash \neg(A \land B)
9. \neg((p \land \neg q) \land (q \land \neg p))  9, Lemma \neg B \vdash \neg(A \land B)
10. \neg((p \land \neg q) \land (q \land \neg p))  3, 4-7, 8-10, \lor E
11. \neg(p \land \neg q)  Assumption
12. \neg q  LEM
13. \neg(q \land \neg p)  12, Lemma \neg A \vdash \neg(A \land B)
14. \neg((p \land \neg q) \land (q \land \neg p))  15, Lemma \neg A \vdash \neg(A \land B)
15. \neg(p \land \neg q)  Assumption (not used)
16. \neg((p \land \neg q) \land (q \land \neg p))  12, Lemma \neg A \vdash \neg(A \land B)
17. \neg(q \land \neg p)  18, Lemma \neg A \vdash \neg(A \land B)
18. \neg((p \land \neg q) \land (q \land \neg p))  13, 14-16, 17-19, \lor E
19. \neg((p \land \neg q) \land (q \land \neg p))  1, 2-11, 12-20, \lor E
For comparison, below left is the same proof worked out in JAPE, except that we used P instead of p and R instead of q:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>2</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>¬R</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>¬R</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>¬(p ∧ ¬R)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>(p ∧ ¬R) ∧ (R ∧ ¬P)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>¬(R ∧ ¬P)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>¬(p ∧ ¬R) ∧ (R ∧ ¬P)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>P</td>
<td>¬R</td>
</tr>
<tr>
<td>12</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>R</td>
<td>¬R</td>
</tr>
<tr>
<td>14</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>¬R</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>¬(p ∧ ¬R) ∧ (R ∧ ¬P)</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>¬(p ∧ ¬R)</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>¬(p ∧ ¬R)</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>¬(p ∧ ¬R)</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>¬(p ∧ ¬R)</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>¬(p ∧ ¬R)</td>
<td></td>
</tr>
</tbody>
</table>

Corollary: If $A_1, \ldots, A_n \models B$ then also $A_1, \ldots, A_n \models \neg B$ in natural deduction.

The corollary states that there is a proof corresponding to any entailment. It is not difficult to show that the corollary reduces to the theorem.

“Living” Meta-Proof: Using the technique outlined here, it is possible to construct a program that will automatically produce a natural deduction proof for any tautology. I call this program a living proof, because examples can be tried by supplying them to the program. Below is the output for the current example, using a program based on a similar technique.

Proof for tautology: or(implies(p,q),implies(q,p));

1: or(p,not(p)) [lem]
   | 2: p [assumption(or-elim)]
   | 3: or(q,not(q)) [lem]
   |   | 4: q [assumption(or-elim)]
   |   |   | 5: implies(p,q) [implies-intro(4)]
   |   |   | 6: or(implies(p,q),implies(q,p)) [or-intro(5)]
   |   |   |   |   |   | 7: not(q) [assumption(or-elim)]
   |   |   |   |   |   |   | 8: implies(q,p) [derived(7)]
   |   |   |   |   |   |   | 9: or(implies(p,q),implies(q,p)) [or-intro(8)]
   |   |   |   |   |   |   |   | 10: or(implies(p,q),implies(q,p)) [or-elim(3,4,6,7,9)]
   |   |   |   |   |   |   |   |   |   |   | 11: not(p) [assumption(or-elim)]
   |   |   |   |   |   |   |   |   |   |   | 12: or(q,not(q)) [lem]
   |   |   |   |   |   |   |   |   |   |   |   | 13: q [assumption(or-elim)]
   |   |   |   |   |   |   |   |   |   |   |   | 14: implies(p,q) [derived(11)]
   |   |   |   |   |   |   |   |   |   |   |   | 15: or(implies(p,q),implies(q,p)) [or-intro(14)]
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | 16: not(q) [assumption(or-elim)]
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | 17: implies(p,q) [derived(11)]
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | 18: or(implies(p,q),implies(q,p)) [or-intro(17)]
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | 19: or(implies(p,q),implies(q,p)) [or-elim(12,13,15,16,18)]
   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | 20: or(implies(p,q),implies(q,p)) [or-elim(1,2,10,11,19)]