In computer:

\[
\text{light1\_broken} \leftarrow \text{sw\_up} \\
\wedge \text{power} \wedge \text{unlit\_light1}.
\]

\[
\text{sw\_up}.
\]

\[
\text{power} \leftarrow \text{lit\_light2}.
\]

\[
\text{unlit\_light1}.
\]

\[
\text{lit\_light2}.
\]

In user’s mind:

- \text{light1\_broken}: light #1 is broken
- \text{sw\_up}: switch is up
- \text{power}: there is power in the building
- \text{unlit\_light1}: light #1 isn’t lit
- \text{lit\_light2}: light #2 is lit

Conclusion: \text{light1\_broken}

- The computer doesn’t know the meaning of the symbols
- The user can interpret the symbol using their meaning

This will be used next class.
CNF and DNF

How to convert a propositional formula to CNF and DNF.

Two main techniques.

One involves using the substitution theorem and a number of equivalences, including:

1. The definition of implication \( P \rightarrow Q \equiv \neg P \lor Q \)

2. The definition of equivalence \( P \leftrightarrow Q \equiv (P \rightarrow Q) \land (Q \rightarrow P) \)
3. Idempotency \((F \land F) \equiv F\), \((F \lor F) \equiv F\)

4. Commutativity \((F \land G) \equiv (G \land F)\), \((F \lor G) \equiv (G \lor F)\)

5. Associativity \(((F \land G) \land H) \equiv (F \land (G \land H)) \equiv (F \land G \land H)\) \((F \lor G) \lor H) \equiv (F \lor (G \lor H)) \equiv (F \lor G \lor H)\)

6. Absorption \((F \land (F \lor G)) \equiv F\), \((F \lor (F \land G)) \equiv F\)

7. Distributivity \((F \land (G \lor H)) \equiv ((F \land G) \lor (F \land H))\) \((F \lor (G \land H)) \equiv ((F \lor G) \land (F \lor H))\)
8. Double Negation \( \neg\neg F \equiv F \)

9. De Morgan's Laws \( \neg(F \lor G) \equiv (\neg F \land \neg G) \)
\( \neg(F \land G) \equiv (\neg F \lor \neg G) \)

10. Tautology laws \( (F \lor G) \equiv F \) if \( F \) is a tautology
\( (F \land G) \equiv G \) if \( F \) is a tautology

11. Unsatisfiability laws \( (F \lor G) \equiv G \) if \( F \) is unsatisfiable
\( (F \land G) \equiv F \) if \( F \) is unsatisfiable
To convert $\mathcal{F}$ to CNF,

1. Push negation inwards, only

\[
\neg \mathcal{A} \equiv \neg \neg \mathcal{A}
\]

\[
\neg (\mathcal{A} \lor \mathcal{B}) = (\neg \mathcal{A} \land \neg \mathcal{B})
\]

\[
\neg (\mathcal{A} \land \mathcal{B}) = (\neg \mathcal{A} \lor \neg \mathcal{B})
\]

until no subformulas of the form on the LHS of these rules occur.
2. Substitute in F each occurrence of a subformula of the form

\[(F \lor (\neg H)) \Rightarrow ((F \lor G) \land (F \lor H))\]

\[(((F \land G) \lor H)) \Rightarrow (((F \lor H) \land (G \lor H))\]

Until no such subformulas occur.

The resulting formula is a CNF (except for possible occurrences of tautology).

(End of method)
Another technique is based on truth tables. For formula $F$, to convert to DNF, take each row of its truth table and build a conjunction, with a positive literal of the corresponding variable is assigned 1 ($t$), and a negative literal of the corresponding variable is assigned 0.

To convert $F$ to CNF, interchange the roles of 0
and 1, and construct a disjunction for each row of the truth table.

Example. Convert to DNF and CNF the following formula:

\[ F = ((\neg A \rightarrow B) \land ((A \land \neg C) \leftrightarrow B)) \]

<p>| | | | | | | | | | | | | | | | | | | | | | |
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |</p>
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>( \neg A \rightarrow B )</th>
<th>( A \land \neg C )</th>
<th>((A \land \neg C) \leftrightarrow B)</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
The equivalent DNF formula is

\[(A \land \neg B \land C) \lor (A \land B \land \neg C)\]

The equivalent CNF formula is

\[(A \lor B \lor C) \land (A \lor B \land \neg C) \land (A \lor \neg B \lor C) \land (\neg A \land B \land C) \land (\neg A \land B \land \neg C)\]

which is very opaque.

\[\text{This is also an equivalent CNF formula.}\]

\[A \land (B \lor \neg C) = A \land (B \lor C) \land (\neg B \land C)\]
Let's try the other method (for CNF):

\[ F = ((\neg A \rightarrow B) \land (A \land \neg C) \rightarrow B) \]

\[ \equiv ((\neg A \lor B) \land ((A \land \neg C) \rightarrow B) \land (B \rightarrow (A \land \neg C))) \equiv \]

\[ \equiv ((A \lor B) \land ((\neg A \land \neg C) \lor B) \land (\neg B \lor (A \land \neg C))) \equiv \]

\[ \equiv (A \lor B) \land (((\neg A \land \neg C) \lor B) \land (\neg B \lor (A \land \neg C))) \equiv \]

\[ \equiv (A \lor B) \land (((\neg A \lor \neg C) \lor B) \land (\neg B \lor (A \land \neg C))) \equiv \]

\[ \equiv (A \lor B) \land (\neg A \lor (\neg C \lor B) \land (\neg B \lor (A \land \neg C))) \equiv \]

\[ \equiv (A \lor B) \land (\neg A \lor (B \lor \neg C) \land (\neg B \lor (A \land \neg C)) \equiv \]
\[ \equiv A \land (\neg A \lor B \land C) \land (\neg B \lor \neg C) \equiv A \land (B \land C) \land (\neg B \lor \neg C) \]

**Resolution**

The (propositional) resolution rule:

Assume a formula in **clause** form:

Start with F a CNF, so

\[ F = (L_{1,1} \lor \ldots \lor L_{1,n_1}) \land \ldots \land (L_{k,1} \lor \ldots \lor L_{k,n_k}) \]
(F is a conjunction of k clauses, where
the ith clause contains m_i literals)

Write each clause as a set, so F is
expressed as a set of clauses:

\[ F = \{ \{ L_{i1}, \ldots, L_{im_i} \}, \ldots, \{ L_{k1}, \ldots, L_{km_k} \} \} \]

**Definition: Resolvent.**

C_i and C_j are clauses. Then \( R \) is a resolvent of
C_i and C_j if there is a literal \( L \in C_i \) such
that \( L \subseteq C \) and \( R = (C_1 - \{L\}) \cup (C_2 - \{L\}) \).

\[ L = \begin{cases} \neg A_i & \text{if } L = A_i \\ A_i & \text{if } L = \neg A_i \end{cases} \]

Graphical notation: \( \{A, \neg c\} \rightarrow \{A, B, C\} \)

The resolvent \( \{A, B\} \) of \( \{A, \neg c\} \) and \( \{A, B, C\} \)

Theorem. Let \( \Phi \) be a CNF formula, represented
as a set of clauses. Let \( R \) be a resolvent of two clauses \( C_1 \) and \( C_2 \) in \( F \). Then, \( F \) and \( F \cup \{ R \} \) are equivalent.

**Definition.** Let \( F \) be a set of clauses. Then \( \text{Res}(F) \) is defined as:

\[
\text{Res}(F) = F \cup \{ R \mid R \text{ is a resolvent of two clauses in } F \}
\]

Furthermore, define
$\text{Res}^0(F) = F$

$\text{Res}_{p+1}^n(F) = \text{Res}_p(\text{Res}_p^n(F))$ for $n \geq 0$, and

finally let

$\text{Res}_{p+1}^*(F) = \bigcup_{n \geq 0} \text{Res}_{p+1}^n(F)$.

It can be proved that for every finite $F$, there is a $k$ s.t. $\text{Res}_{p+1}^k(F) = \text{Res}_{p+1}^*(F)$.

Define. The empty resolvent is the resolvent of $c_1 = \{L\}$ and $c_2 = \{Z\}$. 
This is also called the empty clause, and is denoted by ∅.

\[\{\neg A_3\} \vdash \{A_3\}\]

The Resolution Theorem (of propositional logic)

[J.A. Robinson proved this for FOL around 1960.]

A clause set \(\mathcal{F}\) is unsatisfiable iff \(\emptyset \in \text{Res}^{m}(\mathcal{F})\).

Example (Ex 3.5 Skriini)

Verify resolution.

Show that \((A \land B \land C)\rightarrow \emptyset\) is a consequence of
The class set \( F = \{ A, B, \neg A, \neg B, C, \neg C, A \land B, A \land C, B \land C, \neg A \land \neg B \land \neg C \} \) is unsatisfiable.

5. Negate \( F \) and show that \( F \land \neg F \) is unsatisfiable.

\[ \neg F = \{ \neg A, \neg B, \neg C \}\] Show that \( F \land \neg F \) is unsatisfiable, i.e., by truth values, show that
\[ \text{Res}^*(\text{Funct}) = \emptyset. \]

\{ A, \neg C \} \quad \{ A, B, C \} \quad \{ \neg A \}, \neg B, \neg C \quad \{ A, B \}

\{ A, B \} \quad \{ \neg A, B, \neg C \}