Outline

Literal, Clause

- Literal: either an atom p (positive literal) or its negation $\neg p$ (negative literal).

Literal, Clause

- Literal: either an atom p (positive literal) or its negation $\neg p$ (negative literal).
- The complementary literal to L:

$$\overline{L} \stackrel{\text{def}}{=} \begin{cases} \neg L, & \text{if } L \text{ is positive;} \\ p, & \text{if } L \text{ has the form } \neg p. \end{cases}$$

In other words, p and $\neg p$ are complementary.

Literal, Clause

- Literal: either an atom p (positive literal) or its negation $\neg p$ (negative literal).
- The complementary literal to L:

$$\overline{L} \stackrel{\text{def}}{=} \begin{cases} \neg L, & \text{if } L \text{ is positive;} \\ p, & \text{if } L \text{ has the form } \neg p. \end{cases}$$

  In other words, p and $\neg p$ are complementary.
- Clause: a disjunction $L_1 \lor \ldots \lor L_n$, $n \geq 0$ of literals.

Literal, Clause

- Literal: either an atom p (positive literal) or its negation $\neg p$ (negative literal).
- The complementary literal to L:

$$\overline{L} \stackrel{\text{def}}{=} \begin{cases} \neg L, & \text{if } L \text{ is positive;} \\ p, & \text{if } L \text{ has the form } \neg p. \end{cases}$$

  In other words, p and $\neg p$ are complementary.
- Clause: a disjunction $L_1 \vee \ldots \vee L_n$, $n \geq 0$ of literals.
  - Empty clause, denoted by $\square$: $n = 0$ (the empty clause is false in every interpretation).
  - Unit clause: $n = 1$.
  - Horn clause: a clause with at most one positive literal.

CNF

- A formula A is in conjunctive normal form, or simply CNF, if it is either $\top$, or $\bot$, or a conjunction of disjunctions of literals:

$$A = \bigwedge_i \bigvee_j L_{i,j}.$$

(In other words, A is a conjunction of clauses.)

- A formula B is called a conjunctive normal form of a formula A if B is equivalent to A and B is in conjunctive normal form.

Satisfiability for Formulas in CNF and Sets of Clauses

- An interpretation I satisfies a formula in CNF

$$A = \bigwedge_{i=1,\ldots,n} C_i.$$

  if and only if it satisfies the set of clauses

$$\{C_i, \ldots, C_n\}.$$

- An interpretation I satisfies a clause

$$L_1 \vee \ldots \vee L_k$$

  if and only if it satisfies at least one literal L_m in this clause.

Satisfiability for Formulas in CNF and Sets of Clauses

- An interpretation I satisfies a formula in CNF

$$A = \bigwedge_{i=1,\ldots,n} C_i.$$

  if and only if it satisfies the set of clauses

$$\{C_i, \ldots, C_n\}.$$

- An interpretation I satisfies a clause

$$L_1 \vee \ldots \vee L_k$$

  if and only if it satisfies at least one literal L_m in this clause.

An interpretation satisfies a set of clauses S if each clause C in S contains at least one literal true in this interpretation.

CNF Transformation

We will transform formulas to their CNFs using the following rewrite rule system:

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \rightarrow B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) \quad \wedge \\
 & & \ldots \quad \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n).
\end{array}
$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & \begin{array}{cc} (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\ \ldots & \land \\ (A_m \lor B_1 \lor \ldots \lor B_n). & \end{array}
\end{array}$$

CNF, Example

$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) \quad \land \\
 & & \ldots \quad \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n).
\end{array}
$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & \quad (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
& & \qquad \cdots & \land \\
& & \quad (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r)) \Rightarrow$

$\neg(\neg((p \to q) \wedge (p \wedge q \to r)) \vee (p \to r)) \Rightarrow$

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \to B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & \begin{array}{cc} (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\ \ldots & \wedge \\ (A_m \vee B_1 \vee \ldots \vee B_n). & \end{array}
\end{array}
$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \wedge (p \wedge q \to r)) \vee (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \wedge (p \wedge q \to r)) \wedge \neg(p \to r) \Rightarrow$$

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \to B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) \quad \wedge \\
 & & \ldots \quad \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n).
\end{array}$$

CNF, Example

$$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \wedge (p \wedge q \to r)) \vee (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \wedge (p \wedge q \to r)) \wedge \neg(p \to r) \Rightarrow$$
$$(p \to q) \wedge (p \wedge q \to r) \wedge \neg(p \to r) \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), & \\
A \to B & \Rightarrow & \neg A \vee B, & \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, & \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\
 & & \ldots & \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) \quad \land \\
 & & \ldots \quad \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n).
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) \quad \land \\
 & & \ldots \quad \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n).
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land p \land \neg r \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land p \land \neg r \Rightarrow$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land p \land \neg r \Rightarrow$$
$$(p \to q) \land (\neg(p \land q) \lor r) \land p \land \neg r \Rightarrow$$

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) \quad \land \\
 & & \ldots \quad \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n).
\end{array}$$

CNF, Example

$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$
$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$
$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$
$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$
$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$
$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$
$(p \to q) \land (p \land q \to r) \land p \land \neg r \Rightarrow$
$(p \to q) \land (\neg(p \land q) \lor r) \land p \land \neg r \Rightarrow$

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) \quad \land \\
 & & \ldots \quad \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n).
\end{array}
$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land p \land \neg r \Rightarrow$$
$$(p \to q) \land (\neg(p \land q) \lor r) \land p \land \neg r \Rightarrow$$
$$(p \to q) \land (\neg p \lor \neg q \lor r) \land p \land \neg r$$

$$\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), & \\
A \to B & \Rightarrow & \neg A \lor B, & \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, & \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) & \land \\
 & & \ldots & \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n). &
\end{array}$$

CNF, Example

$$
\begin{aligned}
&\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r)) \Rightarrow \\
&\neg(\neg((p \to q) \wedge (p \wedge q \to r)) \vee (p \to r)) \Rightarrow \\
&\neg\neg((p \to q) \wedge (p \wedge q \to r)) \wedge \neg(p \to r) \Rightarrow \\
&(p \to q) \wedge (p \wedge q \to r) \wedge \neg(p \to r) \Rightarrow \\
&(p \to q) \wedge (p \wedge q \to r) \wedge \neg(\neg p \vee r) \Rightarrow \\
&(p \to q) \wedge (p \wedge q \to r) \wedge \neg\neg p \wedge r \Rightarrow \\
&(p \to q) \wedge (p \wedge q \to r) \wedge p \wedge \neg r \Rightarrow \\
&(p \to q) \wedge (\neg(p \wedge q) \vee r) \wedge p \wedge \neg r \Rightarrow \\
&(p \to q) \wedge (\neg p \vee \neg q \vee r) \wedge p \wedge \neg r
\end{aligned}
$$

$$
\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), & \\
A \to B & \Rightarrow & \neg A \vee B, & \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, & \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\
 & & \ldots & \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n). &
\end{array}
$$

CNF, Example

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\neg(\neg((p \to q) \land (p \land q \to r)) \lor (p \to r)) \Rightarrow$$
$$\neg\neg((p \to q) \land (p \land q \to r)) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(p \to r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg(\neg p \lor r) \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land \neg\neg p \land r \Rightarrow$$
$$(p \to q) \land (p \land q \to r) \land p \land \neg r \Rightarrow$$
$$(p \to q) \land (\neg(p \land q) \lor r) \land p \land \neg r \Rightarrow$$
$$(p \to q) \land (\neg p \lor \neg q \lor r) \land p \land \neg r$$
$$(\neg p \lor q) \land (\neg p \lor \neg q \lor r) \land p \land \neg r$$

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \lor B) \land (\neg B \lor A), \\
A \to B & \Rightarrow & \neg A \lor B, \\
\neg(A \land B) & \Rightarrow & \neg A \lor \neg B, \\
\neg(A \lor B) & \Rightarrow & \neg A \land \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \land \ldots \land A_m) \lor B_1 \lor \ldots \lor B_n & \Rightarrow & (A_1 \lor B_1 \lor \ldots \lor B_n) \quad \land \\
 & & \ldots \quad \land \\
 & & (A_m \lor B_1 \lor \ldots \lor B_n).
\end{array}$$

The resulting formula is in CNF.

Why the CNF Transformation Algorithm is Correct

$$
\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), & \\
A \rightarrow B & \Rightarrow & \neg A \vee B, & \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, & \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & \quad (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\
 & & \qquad \ldots & \wedge \\
 & & \quad (A_m \vee B_1 \vee \ldots \vee B_n). &
\end{array}
$$

Why the CNF Transformation Algorithm is Correct

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \rightarrow B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) \quad \wedge \\
 & & \ldots \quad \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n).
\end{array}$$

A formula to which no rewrite rule is applicable

- contains no $\leftrightarrow$;

Why the CNF Transformation Algorithm is Correct

$$
\begin{array}{rcll}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), & \\
A \rightarrow B & \Rightarrow & \neg A \vee B, & \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, & \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, & \\
\neg\neg A & \Rightarrow & A, & \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & \quad (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\
 & & \quad \ldots & \wedge \\
 & & \quad (A_m \vee B_1 \vee \ldots \vee B_n). &
\end{array}
$$

A formula to which no rewrite rule is applicable

- contains no $\leftrightarrow$;
- contains no $\rightarrow$;

Why the CNF Transformation Algorithm is Correct

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \rightarrow B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) \quad \wedge \\
 & & \ldots \quad \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n).
\end{array}
$$

A formula to which no rewrite rule is applicable

- contains no $\leftrightarrow$;
- contains no $\rightarrow$;
- may only contain $\neg$ applied to atoms;

Why the CNF Transformation Algorithm is Correct

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \rightarrow B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & (A_1 \vee B_1 \vee \ldots \vee B_n) \quad \wedge \\
 & & \ldots \quad \wedge \\
 & & (A_m \vee B_1 \vee \ldots \vee B_n).
\end{array}
$$

A formula to which no rewrite rule is applicable

- contains no $\leftrightarrow$;
- contains no $\rightarrow$;
- may only contain $\neg$ applied to atoms;
- cannot contain $\wedge$ in the scope of $\vee$;

Why the CNF Transformation Algorithm is Correct

$$
\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \rightarrow B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & \begin{array}{cc} (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\ \ldots & \wedge \\ (A_m \vee B_1 \vee \ldots \vee B_n). & \end{array}
\end{array}
$$

A formula to which no rewrite rule is applicable

- contains no $\leftrightarrow$;
- contains no $\rightarrow$;
- may only contain $\neg$ applied to atoms;
- cannot contain $\wedge$ in the scope of $\vee$;
- (hence) is in CNF.

Why the CNF Transformation Algorithm is Correct

$$\begin{array}{rcl}
A \leftrightarrow B & \Rightarrow & (\neg A \vee B) \wedge (\neg B \vee A), \\
A \rightarrow B & \Rightarrow & \neg A \vee B, \\
\neg(A \wedge B) & \Rightarrow & \neg A \vee \neg B, \\
\neg(A \vee B) & \Rightarrow & \neg A \wedge \neg B, \\
\neg\neg A & \Rightarrow & A, \\
(A_1 \wedge \ldots \wedge A_m) \vee B_1 \vee \ldots \vee B_n & \Rightarrow & \begin{array}{cc} (A_1 \vee B_1 \vee \ldots \vee B_n) & \wedge \\ \ldots & \wedge \\ (A_m \vee B_1 \vee \ldots \vee B_n). & \end{array}
\end{array}$$

A formula to which no rewrite rule is applicable

- contains no $\leftrightarrow$;
- contains no $\rightarrow$;
- may only contain $\neg$ applied to atoms;
- cannot contain $\wedge$ in the scope of $\vee$;
- (hence) is in CNF.

Termination is a separate issue ...

CNF and Satisfiability

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\cdots$$
$$(\neg p \lor q) \land (\neg p \lor \neg q \lor r) \land p \land \neg r$$

CNF and Satisfiability

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\cdots$$
$$(\neg p \lor q) \land (\neg p \lor \neg q \lor r) \land p \land \neg r$$

Therefore, the formula

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$$

has the same models as the set consisting of four clauses

$$\neg p \lor q$$
$$\neg p \lor \neg q \lor r$$
$$p$$
$$\neg r$$

CNF and Satisfiability

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r)) \Rightarrow$$
$$\dots$$
$$(\neg p \lor q) \land (\neg p \lor \neg q \lor r) \land p \land \neg r$$

Therefore, the formula

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$$

has the same models as the set consisting of four clauses

$$\begin{array}{l} \neg p \lor q \\ \neg p \lor \neg q \lor r \\ p \\ \neg r \end{array}$$

The CNF transformation reduces the satisfiability problem for formulas to the satisfiability problem for sets of clauses.

Problem

Compute the CNF of

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))).$$

Problem

Compute the CNF of

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))).$$

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$

Problem

Compute the CNF of

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))).$$

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))) \Rightarrow$$

$$(\neg p_1 \lor (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) \;\land$$
$$(p_1 \lor \neg(p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))))$$

Problem

Compute the CNF of

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))).$$

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))) \Rightarrow$$

$$\begin{aligned} &(\neg p_1 \lor (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) \land \\ &(p_1 \lor \neg(p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) \Rightarrow \end{aligned}$$

$$\begin{aligned} (\neg p_1 \lor \quad &((\neg p_2 \lor (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))) \land \\ &(p_2 \lor \neg(p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))))) \land \\ (p_1 \lor \neg(p_2 &\leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) \end{aligned}$$

Problem

Compute the CNF of

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))).$$

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))) \Rightarrow$$

$$\begin{array}{l} (\neg p_1 \lor (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) \ \land \\ (p_1 \lor \neg(p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) \Rightarrow \end{array}$$

$$\begin{array}{ll} (\neg p_1 \lor & ((\neg p_2 \lor (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))) \land \\ & (p_2 \lor \neg(p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))))) \land \\ (p_1 \lor \neg(p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))) & \end{array}$$

If we continue, the formula will grow exponentially.

CNF is Exponential

There are formulas for which the shortest CNF has an exponential size.

CNF is Exponential

There are formulas for which the shortest CNF has an exponential size.

Is there any way to avoid exponential blowup?

Idea

Using so-called naming or definition introduction.

Idea

Using so-called naming or definition introduction.

- Take a non-trivial subformula A.

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$

Idea

Using so-called naming or definition introduction.

- Take a non-trivial subformula A.
- Introduce a new name n for it. A name is a new propositional variable.

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$

Idea

Using so-called naming or definition introduction.

- Take a non-trivial subformula A.
- Introduce a new name n for it. A name is a new propositional variable.
- Add a formula stating that n is equivalent to A (definition for n).

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$
$$n \leftrightarrow (p_5 \leftrightarrow p_6)$$

Idea

Using so-called naming or definition introduction.

- Take a non-trivial subformula A.
- Introduce a new name n for it. A name is a new propositional variable.
- Add a formula stating that n is equivalent to A (definition for n).

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$
$$n \leftrightarrow (p_5 \leftrightarrow p_6)$$

- Replace the subformula by its name:

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow n)))$$
$$n \leftrightarrow (p_5 \leftrightarrow p_6)$$

Idea

Using so-called naming or definition introduction.

- Take a non-trivial subformula A.
- Introduce a new name n for it. A name is a new propositional variable.
- Add a formula stating that n is equivalent to A (definition for n).

$$
\begin{array}{l}
p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6)))) \\
n \leftrightarrow (p_5 \leftrightarrow p_6)
\end{array}
$$

- Replace the subformula by its name:

$$
\begin{array}{l}
p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow n))) \\
n \leftrightarrow (p_5 \leftrightarrow p_6)
\end{array}
$$

The new set of two formulas has the same models as the original one if we restrict ourselves to the original set of variables $\{p_1, \ldots, p_6\}$.

Idea

Using so-called naming or definition introduction.

- Take a non-trivial subformula A.
- Introduce a new name n for it. A name is a new propositional variable.
- Add a formula stating that n is equivalent to A (definition for n).

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$
$$n \leftrightarrow (p_5 \leftrightarrow p_6)$$

- Replace the subformula by its name:

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow n)))$$
$$n \leftrightarrow (p_5 \leftrightarrow p_6)$$

The new set of two formulas has the same models as the original one if we restrict ourselves to the original set of variables $\{p_1, \ldots, p_6\}$.
But this set is not equivalent to the original formula.

After Several Steps

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$

$$\begin{array}{l} p_1 \leftrightarrow (p_2 \leftrightarrow n_3); \\ n_3 \leftrightarrow (p_3 \leftrightarrow n_4); \\ n_4 \leftrightarrow (p_4 \leftrightarrow n_5); \\ n_5 \leftrightarrow (p_5 \leftrightarrow p_6). \end{array}$$

After Several Steps

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$

$$\begin{array}{l} p_1 \leftrightarrow (p_2 \leftrightarrow n_3); \\ n_3 \leftrightarrow (p_3 \leftrightarrow n_4); \\ n_4 \leftrightarrow (p_4 \leftrightarrow n_5); \\ n_5 \leftrightarrow (p_5 \leftrightarrow p_6). \end{array}$$

The conversion of the original formula to CNF introduces 32 copies of p_6.

After Several Steps

$$p_1 \leftrightarrow (p_2 \leftrightarrow (p_3 \leftrightarrow (p_4 \leftrightarrow (p_5 \leftrightarrow p_6))))$$

$$\begin{array}{l} p_1 \leftrightarrow (p_2 \leftrightarrow n_3); \\ n_3 \leftrightarrow (p_3 \leftrightarrow n_4); \\ n_4 \leftrightarrow (p_4 \leftrightarrow n_5); \\ n_5 \leftrightarrow (p_5 \leftrightarrow p_6). \end{array}$$

The conversion of the original formula to CNF introduces 32 copies of p_6.

The conversion of the new set of formulas to CNF introduces 4 copies of p_6.

Clausal Form

- Clausal form of a formula A: a set of clauses which is satisfiable if and only if A is satisfiable.

Clausal Form

- Clausal form of a formula A: a set of clauses which is satisfiable if and only if A is satisfiable.
- Clausal form of a set S of formulas: a set of clauses which is satisfiable if and only if so is S.

Clausal Form

- Clausal form of a formula A: a set of clauses which is satisfiable if and only if A is satisfiable.
- Clausal form of a set S of formulas: a set of clauses which is satisfiable if and only if so is S.

We can require even more: that A and S have the same models in the language of A.

Clausal Form

- Clausal form of a formula A: a set of clauses which is satisfiable if and only if A is satisfiable.
- Clausal form of a set S of formulas: a set of clauses which is satisfiable if and only if so is S.

We can require even more: that A and S have the same models in the language of A.

Using clausal normal forms instead of conjunctive normal forms we can convert any formula to a set of clauses in almost linear time.

Definitional Clause Form Transformation

This algorithm converts a formula A into a set of clauses S such that S is a clausal normal form of A.

If A has the form $C_1 \wedge \ldots \wedge C_n$, where $n \geq 1$ and each C_i is a clause, then $S \stackrel{\text{def}}{=} \{C_1, \ldots, C_n\}$.

Otherwise, introduce a name for each subformula B of A such that B is not a literal and use this name instead of the formula.

Example

	subformula	definition	clauses
	$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r))$		

Converting a formula to clausal form.

Example

	subformula	definition	clauses
	$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$		
	$(p \to q) \land (p \land q \to r) \to (p \to r)$		
	$(p \to q) \land (p \land q \to r)$		
	$p \to q$		
	$p \land q \to r$		
	$p \land q$		
	$p \to r$		

Take all subformulas that are not literals.

Example

	subformula	definition	clauses
n_1	$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$		
n_2	$(p \to q) \land (p \land q \to r) \to (p \to r)$		
n_3	$(p \to q) \land (p \land q \to r)$		
n_4	$p \to q$		
n_5	$p \land q \to r$		
n_6	$p \land q$		
n_7	$p \to r$		

Introduce names for these formulas.

Example

	subformula	definition	clauses
n_1	$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$	$n_1 \leftrightarrow \neg n_2$	
n_2	$(p \to q) \land (p \land q \to r) \to (p \to r)$	$n_2 \leftrightarrow (n_3 \to n_7)$	
n_3	$(p \to q) \land (p \land q \to r)$	$n_3 \leftrightarrow (n_4 \land n_5)$	
n_4	$p \to q$	$n_4 \leftrightarrow (p \to q)$	
n_5	$p \land q \to r$	$n_5 \leftrightarrow (n_6 \to r)$	
n_6	$p \land q$	$n_6 \leftrightarrow (p \land q)$	
n_7	$p \to r$	$n_7 \leftrightarrow (p \to r)$	

Introduce definitions.

Example

	subformula	definition	clauses
			n_1
n_1	$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r))$	$n_1 \leftrightarrow \neg n_2$	$\neg n_1 \vee \neg n_2$ $n_1 \vee n_2$
n_2	$(p \to q) \wedge (p \wedge q \to r) \to (p \to r)$	$n_2 \leftrightarrow (n_3 \to n_7)$	$\neg n_2 \vee \neg n_3 \vee n_7$ $n_3 \vee n_2$ $\neg n_7 \vee n_2$
n_3	$(p \to q) \wedge (p \wedge q \to r)$	$n_3 \leftrightarrow (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$ $\neg n_3 \vee n_5$ $\neg n_4 \vee \neg n_5 \vee n_3$
n_4	$p \to q$	$n_4 \leftrightarrow (p \to q)$	$\neg n_4 \vee \neg p \vee q$ $p \vee n_4$ $\neg q \vee n_4$
n_5	$p \wedge q \to r$	$n_5 \leftrightarrow (n_6 \to r)$	$\neg n_5 \vee \neg n_6 \vee r$ $n_6 \vee n_5$ $\neg r \vee n_5$
n_6	$p \wedge q$	$n_6 \leftrightarrow (p \wedge q)$	$\neg n_6 \vee p$ $\neg n_6 \vee q$ $\neg p \vee \neg q \vee n_6$
n_7	$p \to r$	$n_7 \leftrightarrow (p \to r)$	$\neg n_7 \vee \neg p \vee r$ $p \vee n_7$ $\neg r \vee n_7$

Convert the resulting formula to CNF using the standard algorithm.

Optimised Definitional Clause Form Transformation

If we introduce a name for a subformula and the occurence of the subformula is positive or negative, then an implication is used instead of equivalence.

Optimised Definitional Clause Form Transformation

If we introduce a name for a subformula and the occurence of the subformula is positive or negative, then an implication is used instead of equivalence.

See Chapter 5 for a precise description!

Example

	subformula	definition	clauses
			n_1
n_1	$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r))$	$n_1 \to \neg n_2$	$\neg n_1 \vee \neg n_2$ $n_1 \vee n_2$
n_2	$(p \to q) \wedge (p \wedge q \to r) \to (p \to r)$	$(n_3 \to n_7) \to n_2$	$\neg n_2 \vee \neg n_3 \vee n_7$ $n_3 \vee n_2$ $\neg n_7 \vee n_2$
n_3	$(p \to q) \wedge (p \wedge q \to r)$	$n_3 \to (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$ $\neg n_3 \vee n_5$ $\neg n_4 \vee \neg n_5 \vee n_3$
n_4	$p \to q$	$n_4 \to (p \to q)$	$\neg n_4 \vee \neg p \vee q$ $p \vee n_4$ $\neg q \vee n_4$
n_5	$p \wedge q \to r$	$n_5 \to (n_6 \to r)$	$\neg n_5 \vee \neg n_6 \vee r$ $n_6 \vee n_5$ $\neg r \vee n_5$
n_6	$p \wedge q$	$n_6 \to (p \wedge q)$	$\neg n_6 \vee p$ $\neg n_6 \vee q$ $\neg p \vee \neg q \vee n_6$
n_7	$p \to r$	$(p \to r) \to n_7$	$\neg n_7 \vee \neg p \vee r$ $p \vee n_7$ $\neg r \vee n_7$

Example

	subformula	definition	clauses
			n_1
n_1	$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$	$n_1 \to \neg n_2$	$\neg n_1 \lor \neg n_2$ $n_1 \lor n_2$
n_2	$(p \to q) \land (p \land q \to r) \to (p \to r)$	$(n_3 \to n_7) \to n_2$	$\neg n_2 \lor \neg n_3 \lor n_7$ $n_3 \lor n_2$ $\neg n_7 \lor n_2$
n_3	$(p \to q) \land (p \land q \to r)$	$n_3 \to (n_4 \land n_5)$	$\neg n_3 \lor n_4$ $\neg n_3 \lor n_5$ $\neg n_4 \lor \neg n_5 \lor n_3$
n_4	$p \to q$	$n_4 \to (p \to q)$	$\neg n_4 \lor \neg p \lor q$ $p \lor n_4$ $\neg q \lor n_4$
n_5	$p \land q \to r$	$n_5 \to (n_6 \to r)$	$\neg n_5 \lor \neg n_6 \lor r$ $n_6 \lor n_5$ $\neg r \lor n_5$
n_6	$p \land q$	$n_6 \to (p \land q)$	$\neg n_6 \lor p$ $\neg n_6 \lor q$ $\neg p \lor \neg q \lor n_6$
n_7	$p \to r$	$(p \to r) \to n_7$	$\neg n_7 \lor \neg p \lor r$ $p \lor n_7$ $\neg r \lor n_7$

All clauses shown in the red color are not generated by the optimised transformation.

Example

	subformula	definition	clauses
			n_1
n_1	$\neg((p \to q) \wedge (p \wedge q \to r) \to (p \to r))$	$n_1 \to \neg n_2$	$\neg n_1 \vee \neg n_2$
n_2	$(p \to q) \wedge (p \wedge q \to r) \to (p \to r)$	$(n_3 \to n_7) \to n_2$	$n_3 \vee n_2$ $\neg n_7 \vee n_2$
n_3	$(p \to q) \wedge (p \wedge q \to r)$	$n_3 \to (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$ $\neg n_3 \vee n_5$
n_4	$p \to q$	$n_4 \to (p \to q)$	$\neg n_4 \vee \neg p \vee q$
n_5	$p \wedge q \to r$	$n_5 \to (n_6 \to r)$	$\neg n_5 \vee \neg n_6 \vee r$
n_6	$p \wedge q$	$n_6 \to (p \wedge q)$	$\neg p \vee \neg q \vee n_6$
n_7	$p \to r$	$(p \to r) \to n_7$	$p \vee n_7$ $\neg r \vee n_7$

The optimised transformation gives fewer clauses.

Satisfiability-Checking for Sets of Clauses

The CNF transformation of

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$$

gives the set of four clauses:

$$\begin{array}{l} \neg p \lor q \\ \neg p \lor \neg q \lor r \\ p \\ \neg r \end{array}$$

Satisfiability-Checking for Sets of Clauses

The CNF transformation of

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$$

gives the set of four clauses:

$$\begin{array}{l} \neg p \lor q \\ \neg p \lor \neg q \lor r \\ p \\ \neg r \end{array}$$

Every interpretation that satisfies this set of clauses must assign 1 to p and 0 to r, so we do not have to guess values of these variables.

Satisfiability-Checking for Sets of Clauses

The CNF transformation of

$$\neg((p \to q) \land (p \land q \to r) \to (p \to r))$$

gives the set of four clauses:

$$\begin{array}{l} \neg p \lor q \\ \neg p \lor \neg q \lor r \\ p \\ \neg r \end{array}$$

Every interpretation that satisfies this set of clauses must assign 1 to p and 0 to r, so we do not have to guess values of these variables.

In fact, we can do even better and establish unsatisfiability without any guessing.

Searching for Satisfiability

$$\{ \quad \}$$

$\neg p \vee q$
$\neg p \vee \neg q \vee r$
p
$\neg r$

Searching for Satisfiability

$\{ \qquad \}$

$\neg p \vee q$
$\neg p \vee \neg q \vee r$
p
$\neg r$

Searching for Satisfiability

$$\{p \mapsto 1 \qquad\qquad\qquad\qquad \}$$

$$\neg p \vee q$$
$$\neg p \vee \neg q \vee r$$
$$p$$
$$\neg r$$

Searching for Satisfiability

$$\{p \mapsto 1 \qquad\qquad\qquad\qquad \}$$

$$\neg p \lor q$$
$$\neg p \lor \neg q \lor r$$
$$p$$
$$\neg r$$

Searching for Satisfiability

$\{p \mapsto 1 \qquad\qquad\qquad \}$

q
$\neg q \vee r$

$\neg r$

Searching for Satisfiability

$\{p \mapsto 1 \qquad\qquad\qquad\qquad \}$

q

$\neg q \vee r$

$\neg r$

Searching for Satisfiability

$\{p \mapsto 1, r \mapsto 0 \quad \}$

q

$\neg q \vee r$

$\neg r$

Searching for Satisfiability

$\{p \mapsto 1, r \mapsto 0 \qquad\qquad \}$

q

$\neg q \vee r$

$\neg r$

Searching for Satisfiability

$$\{p \mapsto 1, r \mapsto 0 \qquad\qquad \}$$

q

$\neg q$

Searching for Satisfiability

$\{p \mapsto 1, r \mapsto 0 \qquad\qquad \}$

q

$\neg q$

Searching for Satisfiability

$$\{p \mapsto 1, r \mapsto 0, q \mapsto 1\}$$

q

$\neg q$

Searching for Satisfiability

$$\{p \mapsto 1, r \mapsto 0, q \mapsto 1\}$$

q
$\neg q$

Searching for Satisfiability

$$\{p \mapsto 1, r \mapsto 0, q \mapsto 1\}$$

$\square$

This set of clauses is unsatisfiable.

Unit Propagation

Let S be a set of clauses. Unit propagation: repeatedly performing the following transformation: if S contains a unit clause, i.e. a clause consisting of one literal L, then

1. remove from S every clause of the form $L \vee C'$;
2. replace in S every clause of the form $\overline{L} \vee C'$ by the clause C'.

Unit Propagation, Example

n_1
$\neg n_1 \vee \neg n_2$
$n_1 \vee n_2$
$\neg n_2 \vee \neg n_3 \vee n_7$
$n_3 \vee n_2$
$\neg n_7 \vee n_2$
$\neg n_3 \vee n_4$
$\neg n_3 \vee n_5$
$\neg n_4 \vee \neg n_5 \vee n_3$
$\neg n_4 \vee \neg p \vee q$
$p \vee n_4$

$\neg q \vee n_4$
$\neg n_5 \vee \neg n_6 \vee r$
$n_6 \vee n_5$
$\neg r \vee n_5$
$\neg n_6 \vee p$
$\neg n_6 \vee q$
$\neg p \vee \neg q \vee n_6$
$\neg n_7 \vee \neg p \vee r$
$p \vee n_7$
$\neg r \vee n_7$

Unit Propagation, Example

n_1
$\neg n_1 \vee \neg n_2$
$n_1 \vee n_2$
$\neg n_2 \vee \neg n_3 \vee n_7$
$n_3 \vee n_2$
$\neg n_7 \vee n_2$
$\neg n_3 \vee n_4$
$\neg n_3 \vee n_5$
$\neg n_4 \vee \neg n_5 \vee n_3$
$\neg n_4 \vee \neg p \vee q$
$p \vee n_4$

$\neg q \vee n_4$
$\neg n_5 \vee \neg n_6 \vee r$
$n_6 \vee n_5$
$\neg r \vee n_5$
$\neg n_6 \vee p$
$\neg n_6 \vee q$
$\neg p \vee \neg q \vee n_6$
$\neg n_7 \vee \neg p \vee r$
$p \vee n_7$
$\neg r \vee n_7$

Unit Propagation, Example

$\neg n_2$

$\neg n_2 \lor \neg n_3 \lor n_7$
$n_3 \lor n_2$
$\neg n_7 \lor n_2$
$\neg n_3 \lor n_4$
$\neg n_3 \lor n_5$
$\neg n_4 \lor \neg n_5 \lor n_3$
$\neg n_4 \lor \neg p \lor q$
$p \lor n_4$

$\neg q \lor n_4$
$\neg n_5 \lor \neg n_6 \lor r$
$n_6 \lor n_5$
$\neg r \lor n_5$
$\neg n_6 \lor p$
$\neg n_6 \lor q$
$\neg p \lor \neg q \lor n_6$
$\neg n_7 \lor \neg p \lor r$
$p \lor n_7$
$\neg r \lor n_7$

Unit Propagation, Example

$\neg n_2$

$\neg n_2 \vee \neg n_3 \vee n_7$
$n_3 \vee n_2$
$\neg n_7 \vee n_2$
$\neg n_3 \vee n_4$
$\neg n_3 \vee n_5$
$\neg n_4 \vee \neg n_5 \vee n_3$
$\neg n_4 \vee \neg p \vee q$
$p \vee n_4$

$\neg q \vee n_4$
$\neg n_5 \vee \neg n_6 \vee r$
$n_6 \vee n_5$
$\neg r \vee n_5$
$\neg n_6 \vee p$
$\neg n_6 \vee q$
$\neg p \vee \neg q \vee n_6$
$\neg n_7 \vee \neg p \vee r$
$p \vee n_7$
$\neg r \vee n_7$

Unit Propagation, Example

n_3
$\neg n_7$
$\neg n_3 \vee n_4$
$\neg n_3 \vee n_5$
$\neg n_4 \vee \neg n_5 \vee n_3$
$\neg n_4 \vee \neg p \vee q$
$p \vee n_4$

$\neg q \vee n_4$
$\neg n_5 \vee \neg n_6 \vee r$
$n_6 \vee n_5$
$\neg r \vee n_5$
$\neg n_6 \vee p$
$\neg n_6 \vee q$
$\neg p \vee \neg q \vee n_6$
$\neg n_7 \vee \neg p \vee r$
$p \vee n_7$
$\neg r \vee n_7$

Unit Propagation, Example

n_3
$\neg n_7$
$\neg n_3 \vee n_4$
$\neg n_3 \vee n_5$
$\neg n_4 \vee \neg n_5 \vee n_3$
$\neg n_4 \vee \neg p \vee q$
$p \vee n_4$

$\neg q \vee n_4$
$\neg n_5 \vee \neg n_6 \vee r$
$n_6 \vee n_5$
$\neg r \vee n_5$
$\neg n_6 \vee p$
$\neg n_6 \vee q$
$\neg p \vee \neg q \vee n_6$
$\neg n_7 \vee \neg p \vee r$
$p \vee n_7$
$\neg r \vee n_7$

Unit Propagation, Example

$$\neg q \vee n_4$$
$$\neg n_5 \vee \neg n_6 \vee r$$
$$n_6 \vee n_5$$
$$\neg r \vee n_5$$
$$\neg n_6 \vee p$$
$$\neg n_6 \vee q$$

n_4	$\neg p \vee \neg q \vee n_6$
n_5	
	p
$\neg n_4 \vee \neg p \vee q$	$\neg r$
$p \vee n_4$	

Unit Propagation, Example

$\neg q \vee n_4$
$\neg n_5 \vee \neg n_6 \vee r$
$n_6 \vee n_5$
$\neg r \vee n_5$
$\neg n_6 \vee p$
$\neg n_6 \vee q$
$\neg p \vee \neg q \vee n_6$

n_4
n_5

p
$\neg r$

$\neg n_4 \vee \neg p \vee q$
$p \vee n_4$

Unit Propagation, Example

$\neg n_6$

$\neg n_6 \vee q$

$\neg q \vee n_6$

q

Unit Propagation, Example

$\neg n_6$

$\neg n_6 \vee q$

$\neg q \vee n_6$

q

Unit Propagation, Example

$\square$

We established unsatisfiability of this set of clauses in a completely deterministic way, by unit propagation.

DPLL = Splitting + Unit Propagation

```
procedure DPLL(S)
input: set of clauses S
output: satisfiable or unsatisfiable
parameters: function select_literal
begin
  S  :=  propagate(S)
  if S is empty then return satisfiable
  if S contains □ then return unsatisfiable
  L  :=  select_literal(S)
  if DPLL(S ∪ {L}) = satisfiable
    then return satisfiable
    else return DPLL(S ∪ {L̄})
end
```

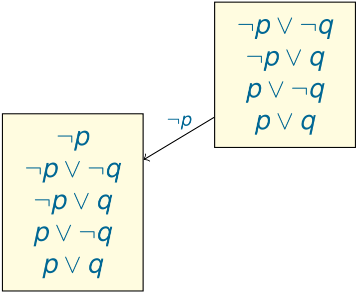
DPLL. Example 1

Can be illustrated using DPLL trees (similar to splitting trees).

$$\neg p \lor \neg q$$
$$\neg p \lor q$$
$$p \lor \neg q$$
$$p \lor q$$

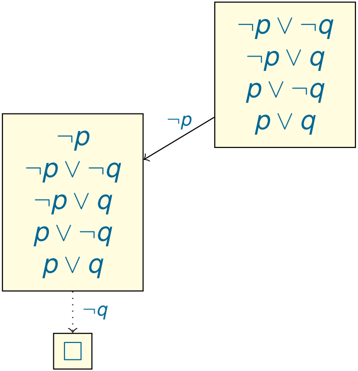
DPLL. Example 1

Can be illustrated using DPLL trees (similar to splitting trees).



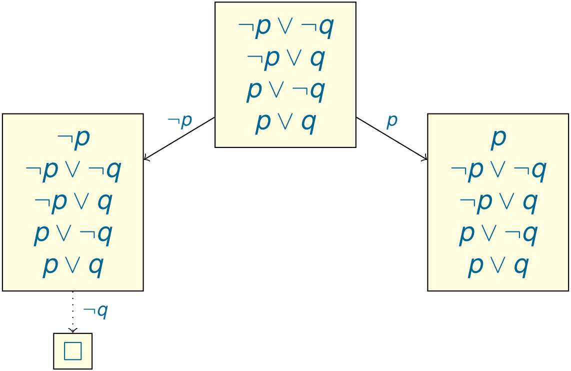
DPLL. Example 1

Can be illustrated using DPLL trees (similar to splitting trees).



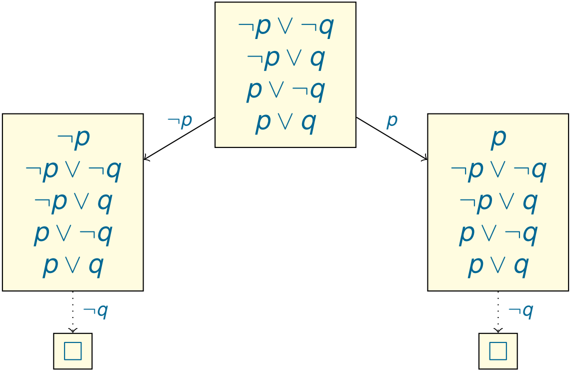
DPLL. Example 1

Can be illustrated using DPLL trees (similar to splitting trees).



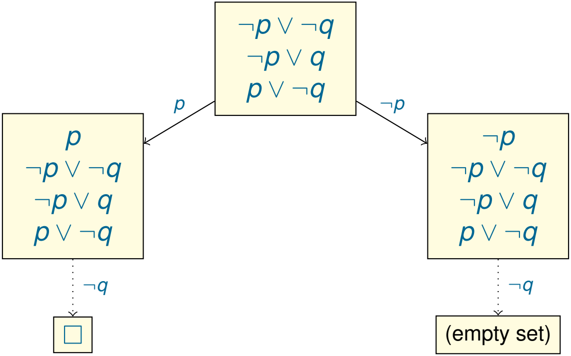
DPLL. Example 1

Can be illustrated using DPLL trees (similar to splitting trees).



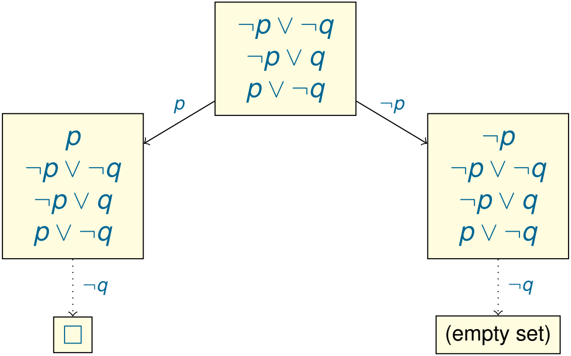
Since all branches end up in a set contaning the empty clause, the initial set of clauses is unsatisfiable.

DPLL. Example 2



The set of clauses is satisfiable.

DPLL. Example 2



The set of clauses is satisfiable.
The model can be obtained by collecting all selected literals and literals used in unit propagation on the branch resulting in the empty set.
This DPLL tree gives us the model $\{p \mapsto 0, q \mapsto 0\}$.

Two Optimisations

Any clause clause $p \vee \neg p \vee C$ is a tautology. Tautologies can be removed.

Two Optimisations

Any clause clause $p \vee \neg p \vee C$ is a tautology. Tautologies can be removed.

A literal L in S is called pure if S contains no clauses of the form $\overline{L} \vee C$

Two Optimisations

Any clause clause $p \lor \neg p \lor C$ is a tautology. Tautologies can be removed.
A literal L in S is called pure if S contains no clauses of the form $\overline{L} \lor C$

All clauses containing a pure literal can be satisfied by assigning a suitable truth value to the variable of this literal.

Hence, clauses containing pure literals can be removed, too.

Pure Literals: Example

$$\neg p_2 \vee \neg p_3$$
$$p_1 \vee \neg p_2$$
$$\neg p_1 \vee p_2 \vee \neg p_3$$
$$\neg p_1 \vee \neg p_3$$
$$p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee \neg p_3$$

Pure Literals: Example

$$\neg p_2 \lor \neg p_3$$
$$p_1 \lor \neg p_2$$
$$\neg p_1 \lor p_2 \lor \neg p_3$$
$$\neg p_1 \lor \neg p_3$$
$$p_1 \lor p_2$$
$$\neg p_1 \lor \neg p_2 \lor \neg p_3$$

The literal $\neg p_3$ is pure in this set. We can remove all clauses containing this literal.

Pure Literals: Example

$$p_1 \vee \neg p_2$$

$$p_1 \vee p_2$$

Pure Literals: Example

$$p_1 \vee \neg p_2$$

$$p_1 \vee p_2$$

Now the literal p_1 is pure in this set. We can remove all clauses containing this literal.

Pure Literals: Example

We obtain the empty set of clauses.

Pure Literals: Example

$$\neg p_2 \lor \neg p_3$$
$$p_1 \lor \neg p_2$$
$$\neg p_1 \lor p_2 \lor \neg p_3$$
$$\neg p_1 \lor \neg p_3$$
$$p_1 \lor p_2$$
$$\neg p_1 \lor \neg p_2 \lor \neg p_3$$

We obtain the empty set of clauses.

This gives us two models:

$$\{p_1 \mapsto 1, p_2 \mapsto 0, p_3 \mapsto 0\}$$
$$\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 0\}$$

Horn Clauses

A clause is called Horn if it contains at most one positive literal.
Examples:

$$p_1$$
$$\neg p_1 \lor p_2$$
$$\neg p_1 \lor \neg p_2 \lor p_3$$
$$\neg p_3 \lor \neg p_4$$

Horn Clauses

A clause is called Horn if it contains at most one positive literal.
Examples:

$$p_1$$
$$\neg p_1 \lor p_2$$
$$\neg p_1 \lor \neg p_2 \lor p_3$$
$$\neg p_3 \lor \neg p_4$$

The following clauses are non-Horn:

$$p_1 \lor p_2$$
$$p_1 \lor \neg p_2 \lor p_3$$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_2$$
$$\neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_3$$

$$\neg p_3 \vee \neg p_4$$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$\neg p_4$$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.
Therefore, unit propagation applied to a set C of Horn clauses gives a set C' of Horn clauses.

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.
Therefore, unit propagation applied to a set C of Horn clauses gives a set C' of Horn clauses.

Two cases:

1. C' contains $\square$. Then, C' (and hence C) is unsatisfiable.

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.
Therefore, unit propagation applied to a set C of Horn clauses gives a set C' of Horn clauses.

Two cases:

1. C' contains $\square$. Then, C' (and hence C) is unsatisfiable.
2. C' does not contain $\square$.

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$\begin{gathered} p_1 \\ \neg p_1 \vee p_2 \\ \neg p_1 \vee \neg p_2 \vee p_3 \\ \neg p_3 \vee \neg p_4 \end{gathered}$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.
Therefore, unit propagation applied to a set C of Horn clauses gives a set C' of Horn clauses.

Two cases:

1. C' contains $\square$. Then, C' (and hence C) is unsatisfiable.
2. C' does not contain $\square$. Then:
   - Each clause in C' has at least two literals.

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$\begin{array}{c} p_1 \\ \neg p_1 \lor p_2 \\ \neg p_1 \lor \neg p_2 \lor p_3 \\ \neg p_3 \lor \neg p_4 \end{array}$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.
Therefore, unit propagation applied to a set C of Horn clauses gives a set C' of Horn clauses.

Two cases:

1. C' contains $\square$. Then, C' (and hence C) is unsatisfiable.
2. C' does not contain $\square$. Then:
   - Each clause in C' has at least two literals.
   - Hence each clause in C' contains at least one negative literal;

Satisfiability of Horn Clauses Can be Decided by Unit Propagation

Example:

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_3 \vee \neg p_4$$

Model: $\{p_1 \mapsto 1, p_2 \mapsto 1, p_3 \mapsto 1, p_4 \mapsto 0\}$
Note that deleting a literal from a Horn clause gives a Horn clause.
Therefore, unit propagation applied to a set C of Horn clauses gives a set C' of Horn clauses.

Two cases:

1. C' contains $\square$. Then, C' (and hence C) is unsatisfiable.
2. C' does not contain $\square$. Then:
   - Each clause in C' has at least two literals.
   - Hence each clause in C' contains at least one negative literal;
   - Hence setting all variables in C' to 0 satisfies C'.

Expressing Properties "k out of n variables are true"

Suppose we have variables $v_1, \ldots, v_n$ and want to express that exactly k of them are true. These formulas are very useful for encoding various problems in SAT.
We will write this property as a formula $T_k(v_1, \ldots, v_n)$.

Expressing Properties "k out of n variables are true"

Suppose we have variables $v_1, \ldots, v_n$ and want to express that exactly k of them are true. These formulas are very useful for encoding various problems in SAT.
We will write this property as a formula $T_k(v_1, \ldots, v_n)$.

First, let us express some simple special cases:

$$T_0(v_1, \ldots, v_n) \stackrel{\text{def}}{=} \neg v_1 \wedge \ldots \wedge \neg v_n$$
$$T_1(v_1, \ldots, v_n) \stackrel{\text{def}}{=} (v_1 \vee \ldots \vee v_n) \wedge \bigwedge_{i<j}(\neg v_i \vee \neg v_j)$$

Expressing Properties "k out of n variables are true"

Suppose we have variables $v_1, \ldots, v_n$ and want to express that exactly k of them are true. These formulas are very useful for encoding various problems in SAT.
We will write this property as a formula $T_k(v_1, \ldots, v_n)$.

First, let us express some simple special cases:

$$T_0(v_1, \ldots, v_n) \stackrel{\text{def}}{=} \neg v_1 \wedge \ldots \wedge \neg v_n$$
$$T_1(v_1, \ldots, v_n) \stackrel{\text{def}}{=} (v_1 \vee \ldots \vee v_n) \wedge \bigwedge_{i<j} (\neg v_i \vee \neg v_j)$$

$$T_{n-1}(v_1, \ldots, v_n) \stackrel{\text{def}}{=} (\neg v_1 \vee \ldots \vee \neg v_n) \wedge \bigwedge_{i<j} (v_i \vee v_j)$$
$$T_n(v_1, \ldots, v_n) \stackrel{\text{def}}{=} v_1 \wedge \ldots \wedge v_n$$

Expressing Properties "k out of n variables are true"

To define T_k for $0 < k < n$, introduce two formulas:

- $T_{\leq k}(v_1, \ldots, v_n)$: at most k variables among $v_1, \ldots, v_n$ are true, where $k = 0 \ldots n-1$;
- $T_{\geq k}(v_1, \ldots, v_n)$: at least k variables among $v_1, \ldots, v_n$ are true, where $k = 1 \ldots n$;

Expressing Properties "k out of n variables are true"

To define T_k for $0 < k < n$, introduce two formulas:

- $T_{\leq k}(v_1, \ldots, v_n)$: at most k variables among $v_1, \ldots, v_n$ are true, where $k = 0 \ldots n-1$;
- $T_{\geq k}(v_1, \ldots, v_n)$: at least k variables among $v_1, \ldots, v_n$ are true, where $k = 1 \ldots n$;

$$T_{\leq k}(v_1, \ldots, v_n) \stackrel{\text{def}}{=} \bigwedge_{\substack{x_1, \ldots, x_{k+1} \in \{v_1, \ldots, v_n\} \\ x_1, \ldots, x_{k+1} \text{ are distinct}}} \neg x_1 \lor \ldots \lor \neg x_{k+1}.$$

Expressing Properties "k out of n variables are true"

To define T_k for $0 < k < n$, introduce two formulas:

- $T_{\leq k}(v_1, \ldots, v_n)$: at most k variables among $v_1, \ldots, v_n$ are true, where $k = 0 \ldots n-1$;
- $T_{\geq k}(v_1, \ldots, v_n)$: at least k variables among $v_1, \ldots, v_n$ are true, where $k = 1 \ldots n$;

$$T_{\leq k}(v_1, \ldots, v_n) \stackrel{\text{def}}{=} \bigwedge_{\substack{x_1, \ldots, x_{k+1} \in \{v_1, \ldots, v_n\} \\ x_1, \ldots, x_{k+1} \text{ are distinct}}} \neg x_1 \vee \ldots \vee \neg x_{k+1}.$$

"At most k variables among $v_1, \ldots, v_n$ are true" is equivalent to "At most $n-k$ variables among $v_1, \ldots, v_n$ are false".

Expressing Properties "k out of n variables are true"

To define T_k for $0 < k < n$, introduce two formulas:

- $T_{\le k}(v_1, \ldots, v_n)$: at most k variables among $v_1, \ldots, v_n$ are true, where $k = 0 \ldots n-1$;
- $T_{\ge k}(v_1, \ldots, v_n)$: at least k variables among $v_1, \ldots, v_n$ are true, where $k = 1 \ldots n$;

$$T_{\le k}(v_1, \ldots, v_n) \stackrel{\text{def}}{=} \bigwedge_{\substack{x_1, \ldots, x_{k+1} \in \{v_1, \ldots, v_n\} \\ x_1, \ldots, x_{k+1} \text{ are distinct}}} \neg x_1 \vee \ldots \vee \neg x_{k+1}.$$

"At most k variables among $v_1, \ldots, v_n$ are true" is equivalent to "At most $n-k$ variables among $v_1, \ldots, v_n$ are false".

$$T_{\ge k}(v_1, \ldots, v_n) \stackrel{\text{def}}{=} \bigwedge_{\substack{x_1, \ldots, x_{n-k+1} \in \{v_1, \ldots, v_n\} \\ x_1, \ldots, x_{n-k+1} \text{ are distinct}}} x_1 \vee \ldots \vee x_{n-k+1}.$$

Sudoku

4		8	7	9			3	
		9	8	2		5		
	2							
9		2			6		1	7
		6	5	8	7	9		
7	8		2			4		6
							4	
		5		4	8	2		
	9			7	2	3		5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

Sudoku

4		8	7	9			3	
		9	8	2		5		
	2							
9		2			6		1	7
		6	5	8	7	9		
7	8		2			4		6
							4	
		5		4	8	2		
	9			7	2	3		5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

Sudoku

4		8	7	9			3	
		9	8	2		5		
	2							
9		2			6		1	7
		6	5	8	7	9		
7	8		2			4		6
							4	
		5		4	8	2		
	9			7	2	3		5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

Sudoku

4		8	7	9			3	
		9	8	2		5		
	2							
9		2			6		1	7
		6	5	8	7	9		
7	8		2			4		6
							4	
		5		4	8	2		
	9			7	2	3		5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

Sudoku

4	1	8	7	9	5	6	3	2
6	3	9	8	2	1	5	7	4
5	2	7	3	6	4	1	8	9
9	5	2	4	3	6	8	1	7
1	4	6	5	8	7	9	2	3
7	8	3	2	1	9	4	5	6
2	6	1	9	5	3	7	4	8
3	7	5	6	4	8	2	9	1
8	9	4	1	7	2	3	6	5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

This instance has exactly one solution.

Sudoku

4	1	8	7	9	5	6	3	2
6	3	9	8	2	1	5	7	4
5	2	7	3	6	4	1	8	9
9	5	2	4	3	6	8	1	7
1	4	6	5	8	7	9	2	3
7	8	3	2	1	9	4	5	6
2	6	1	9	5	3	7	4	8
3	7	5	6	4	8	2	9	1
8	9	4	1	7	2	3	6	5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

This instance has exactly one solution.

Sudoku

4	1	8	7	9	5	6	3	2
6	3	9	8	2	1	5	7	4
5	2	7	3	6	4	1	8	9
9	5	2	4	3	6	8	1	7
1	4	6	5	8	7	9	2	3
7	8	3	2	1	9	4	5	6
2	6	1	9	5	3	7	4	8
3	7	5	6	4	8	2	9	1
8	9	4	1	7	2	3	6	5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

This instance has exactly one solution.

Sudoku

4	1	8	7	9	5	6	3	2
6	3	9	8	2	1	5	7	4
5	2	7	3	6	4	1	8	9
9	5	2	4	3	6	8	1	7
1	4	6	5	8	7	9	2	3
7	8	3	2	1	9	4	5	6
2	6	1	9	5	3	7	4	8
3	7	5	6	4	8	2	9	1
8	9	4	1	7	2	3	6	5

Enter digits from 1 to 9 into the blank spaces.

Every row must contain one of each digit.

So must every column,

as must every 3x3 square.

This instance has exactly one solution.

Sudoku as an instance of SAT

9	4		8	7	9			3	
8			9	8	2		5		
7		2							
6	9		2			6		1	7
5			6	5	8	7	9		
4	7	8		2			4		6
3								4	
2			5		4	8	2		
1		9			7	2	3		5
	1	2	3	4	5	6	7	8	9

Sudoku as an instance of SAT

9	4		8	7	9			3	
8			9	8	2		5		
7		2							
6	9		2			6		1	7
5			6	5	8	7	9		
4	7	8		2			4		6
3								4	
2			5		4	8	2		
1		9			7	2	3		5
	1	2	3	4	5	6	7	8	9

Introduce 729 propositional variables v_{rcd}, where $r, c, d \in \{1, \dots, d\}$.
The variable v_{rcd} denotes that the cell in the row number r and column number c contains the digit d.

Sudoku as an instance of SAT

9	4		8	7	9			3	
8			9	8	2		5		
7		2							
6	9		2			6		1	7
5			6	5	8	7	9		
4	7	8		2			4		6
3								4	
2			5		4	8	2		
1		9			7	2	3		5
	1	2	3	4	5	6	7	8	9

Introduce 729 propositional variables v_{rcd}, where $r, c, d \in \{1, \ldots, d\}$. The variable v_{rcd} denotes that the cell in the row number r and column number c contains the digit d.

For example, this configuration satisfies the formula $v_{129} \wedge v_{268} \wedge \neg v_{691}$.

Sudoku as an instance of SAT

9	4		8	7	9			3	
8			9	8	2		5		
7		2							
6	9		2			6		1	7
5			6	5	8	7	9		
4	7	8		2			4		6
3								4	
2			5		4	8	2		
1		9			7	2	3		5
	1	2	3	4	5	6	7	8	9

Introduce 729 propositional variables v_{rcd}, where $r, c, d \in \{1, \ldots, d\}$.
The variable v_{rcd} denotes that the cell in the row number r and column number c contains the digit d.

For example, this configuration satisfies the formula $v_{129} \wedge v_{268} \wedge \neg v_{691}$.

We should express all rules of sudoku using the variables v_{rcd}.

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$$v_{rc1} \vee v_{rc2} \vee \ldots \vee v_{rc8} \vee v_{rc9}$$
$$\neg v_{rc1} \vee \neg v_{rc2}$$
$$\neg v_{rc1} \vee \neg v_{rc3}$$
$$\ldots$$
$$\neg v_{rc8} \vee \neg v_{rc9}$$

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$$v_{rc1} \lor v_{rc2} \lor \ldots \lor v_{rc8} \lor v_{rc9}$$
$$\neg v_{rc1} \lor \neg v_{rc2}$$
$$\neg v_{rc1} \lor \neg v_{rc3}$$
$$\ldots$$
$$\neg v_{rc8} \lor \neg v_{rc9}$$

Every row must contain one of each digit:

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$$v_{rc1} \vee v_{rc2} \vee \ldots \vee v_{rc8} \vee v_{rc9}$$
$$\neg v_{rc1} \vee \neg v_{rc2}$$
$$\neg v_{rc1} \vee \neg v_{rc3}$$
$$\ldots$$
$$\neg v_{rc8} \vee \neg v_{rc9}$$

Every row must contain one of each digit:

$$\{\neg v_{r,c,d} \vee \neg v_{r,c',d} \mid r, c, c', d \in \{1, ..., 9\}, c < c'\}.$$

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$v_{rc1} \vee v_{rc2} \vee \ldots \vee v_{rc8} \vee v_{rc9}$
$\neg v_{rc1} \vee \neg v_{rc2}$
$\neg v_{rc1} \vee \neg v_{rc3}$
$\ldots$
$\neg v_{rc8} \vee \neg v_{rc9}$

Every row must contain one of each digit:

$$\{\neg v_{r,c,d} \vee \neg v_{r,c',d} \mid r, c, c', d \in \{1, ..., 9\}, c < c'\}.$$

Every column must contain one of each digit: similar.
Every 3x3 square must contain one of each digit: similar.

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$$v_{rc1} \vee v_{rc2} \vee \ldots \vee v_{rc8} \vee v_{rc9}$$
$$\neg v_{rc1} \vee \neg v_{rc2}$$
$$\neg v_{rc1} \vee \neg v_{rc3}$$
$$\ldots$$
$$\neg v_{rc8} \vee \neg v_{rc9}$$

2,997 clauses,
6,561 literals

Every row must contain one of each digit:

$$\{\neg v_{r,c,d} \vee \neg v_{r,c',d} \mid r, c, c', d \in \{1, ..., 9\}, c < c'\}.$$

Every column must contain one of each digit:
similar.

Every 3x3 square must contain one of each digit:
similar.

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$v_{rc1} \vee v_{rc2} \vee \ldots \vee v_{rc8} \vee v_{rc9}$

$\neg v_{rc1} \vee \neg v_{rc2}$

$\neg v_{rc1} \vee \neg v_{rc3}$

$\ldots$

$\neg v_{rc8} \vee \neg v_{rc9}$

2,997 clauses,
6,561 literals

Every row must contain one of each digit:

$$\{\neg v_{r,c,d} \vee \neg v_{r,c',d} \mid r, c, c', d \in \{1, ..., 9\}, c < c'\}.$$

2,916 clauses,
5,832 literals

Every column must contain one of each digit:
similar.

2,916 clauses,
5,832 literals

Every 3x3 square must contain one of each digit:
similar.

2,916 clauses,
5,832 literals

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$$v_{rc1} \lor v_{rc2} \lor \ldots \lor v_{rc8} \lor v_{rc9}$$
$$\neg v_{rc1} \lor \neg v_{rc2}$$
$$\neg v_{rc1} \lor \neg v_{rc3}$$
$$\ldots$$
$$\neg v_{rc8} \lor \neg v_{rc9}$$

2,997 clauses, 6,561 literals

Every row must contain one of each digit:

$$\{\neg v_{r,c,d} \lor \neg v_{r,c',d} \mid r, c, c', d \in \{1, ..., 9\}, c < c'\}.$$

2,916 clauses, 5,832 literals

Every column must contain one of each digit: similar.

2,916 clauses, 5,832 literals

Every 3x3 square must contain one of each digit: similar.

2,916 clauses, 5,832 literals

729 variables, 11,745 clauses, 24,057 literals, nearly all clauses are binary.

Encoding Sudoku in SAT

We have to write down that each cell contains exactly one digit.

$$v_{rc1} \lor v_{rc2} \lor \ldots \lor v_{rc8} \lor v_{rc9}$$
$$\neg v_{rc1} \lor \neg v_{rc2}$$
$$\neg v_{rc1} \lor \neg v_{rc3}$$
$$\ldots$$
$$\neg v_{rc8} \lor \neg v_{rc9}$$

2,997 clauses,
6,561 literals

Every row must contain one of each digit:

$$\{\neg v_{r,c,d} \lor \neg v_{r,c',d} \mid r, c, c', d \in \{1, ..., 9\}, c < c'\}.$$

2,916 clauses,
5,832 literals

Every column must contain one of each digit: similar.

2,916 clauses,
5,832 literals

Every 3x3 square must contain one of each digit: similar.

2,916 clauses,
5,832 literals

729 variables, 11,745 clauses, 24,057 literals, nearly all clauses are binary.

Finally, we add unit clauses corresponding to the initial configuration, for example v_{129}.

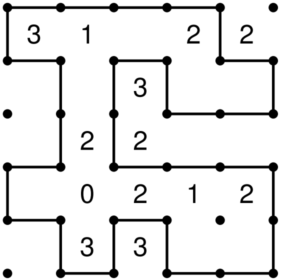
3	1		2	2
		3		
	2	2		
	0	2	1	2
	3	3		

Loop the Loop

3	1		2	2
		3		
	2	2		
	0	2	1	2
	3	3		

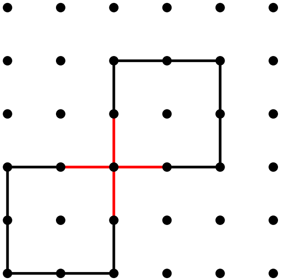
You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

Loop the Loop



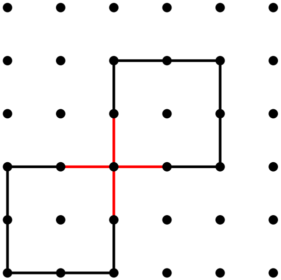
You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

Loop the Loop



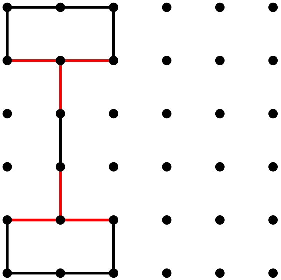
You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

Loop the Loop



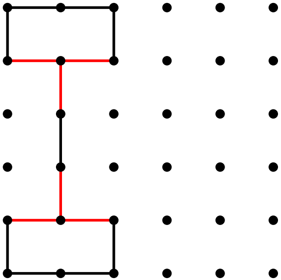
You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.
A crossing is a node with four arcs attached to it.

Loop the Loop



You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

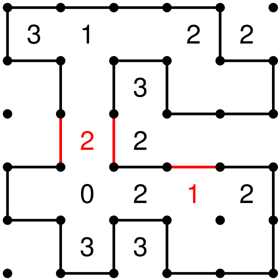
Loop the Loop



You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

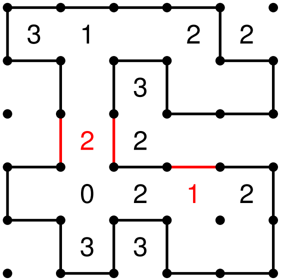
A branch is a node with three arcs attached to it.

Loop the Loop



You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

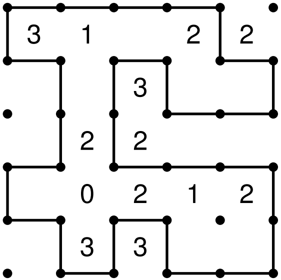
Loop the Loop



You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.

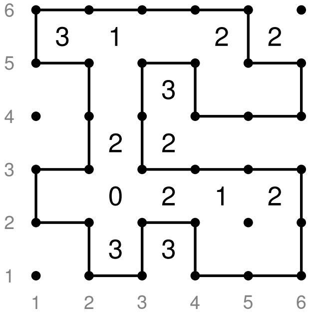
If a cell contains a number m, then there should be m arcs around this number.

Loop the Loop



You have to draw lines between the dots to form a single loop without crossings or branches. The numbers indicate how many lines surround it.
A crossing is a node with four arcs attached to it.
A branch is a node with three arcs attached to it.
If a cell contains a number m, then there should be m arcs around this number.
All these properties are formulated in terms of (a number of) arcs.

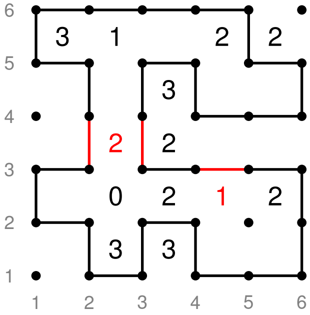
Formalisation



Introduce variables denoting arcs:

- v_{ij}: there is a vertical arc between the nodes (i, j) and $(i, j + 1)$;
- h_{ij}: there is a horizontal arc between the nodes (i, j) and $(i + 1, j)$.

Formalisation

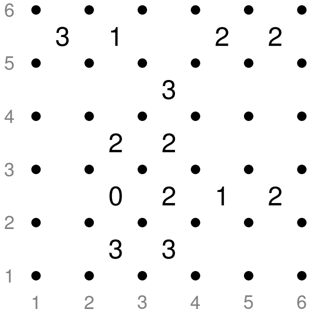


Introduce variables denoting arcs:

- v_{ij}: there is a vertical arc between the nodes (i, j) and $(i, j + 1)$;
- h_{ij}: there is a horizontal arc between the nodes (i, j) and $(i + 1, j)$.

For example, for this position we have $v_{23} \wedge v_{33} \wedge h_{43}$.

Formalisation

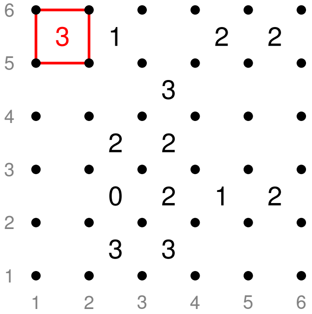


Introduce variables denoting arcs:

- v_{ij}: there is a vertical arc between the nodes (i, j) and $(i, j + 1)$;
- h_{ij}: there is a horizontal arc between the nodes (i, j) and $(i + 1, j)$.

Then almost all properties are formulated using the formulas T_k and these variables.

Formalisation



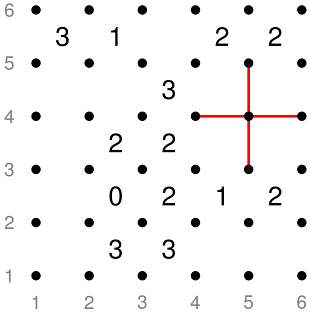
Introduce variables denoting arcs:

- v_{ij}: there is a vertical arc between the nodes (i, j) and $(i, j + 1)$;
- h_{ij}: there is a horizontal arc between the nodes (i, j) and $(i + 1, j)$.

Then almost all properties are formulated using the formulas T_k and these variables. For example,

$$T_3(v_{15}, v_{25}, h_{15}, h_{16})$$

Formalisation



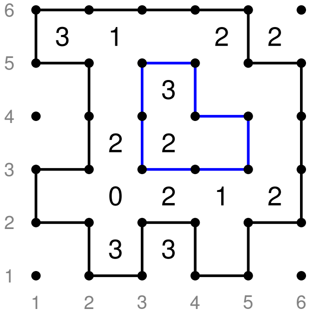
Introduce variables denoting arcs:

- v_{ij}: there is a vertical arc between the nodes (i, j) and $(i, j + 1)$;
- h_{ij}: there is a horizontal arc between the nodes (i, j) and $(i + 1, j)$.

Then almost all properties are formulated using the formulas T_k and these variables. For example,

$$T_3(v_{15}, v_{25}, h_{15}, h_{16})$$
$$T_0(v_{53}, v_{54}, h_{44}, h_{45}) \vee T_2(v_{53}, v_{54}, h_{44}, h_{45})$$

Formalisation



Introduce variables denoting arcs:

- v_{ij}: there is a vertical arc between the nodes (i, j) and $(i, j+1)$;
- h_{ij}: there is a horizontal arc between the nodes (i, j) and $(i+1, j)$.

Then almost all properties are formulated using the formulas T_k and these variables. For example,

$$T_3(v_{15}, v_{25}, h_{15}, h_{16})$$
$$T_0(v_{53}, v_{54}, h_{44}, h_{45}) \vee T_2(v_{53}, v_{54}, h_{44}, h_{45})$$

What we cannot express is the property to have a single loop. In fact, there is no simple way of expressing it in propositional logic.

Running a SAT Solver

Very simple but efficient SAT solver: MiniSat,
`http://minisat.se/`.

Running a SAT Solver

Very simple but efficient SAT solver: MiniSat,
http://minisat.se/.

$$p_1$$
$$\neg p_1 \lor p_2$$
$$\neg p_1 \lor \neg p_2 \lor p_3$$
$$\neg p_2 \lor \neg p_3$$

Running a SAT Solver

Very simple but efficient SAT solver: MiniSat, http://minisat.se/.

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_2 \vee \neg p_3$$

DIMACS input format:

```
p cnf 3 4
1 0
-1 2 0
-1 -2 3 0
-2 -3 0
```

Running a SAT Solver

Very simple but efficient SAT solver: MiniSat, `http://minisat.se/`.

$$p_1$$
$$\neg p_1 \vee p_2$$
$$\neg p_1 \vee \neg p_2 \vee p_3$$
$$\neg p_2 \vee \neg p_3$$

DIMACS input format:

```
p cnf 3 4
1 0
-1 2 0
-1 -2 3 0
-2 -3 0
```

3 variables, 4 clauses.

Running a SAT Solver

Very simple but efficient SAT solver: MiniSat,
http://minisat.se/.

$$p_1$$
$$\neg p_1 \lor p_2$$
$$\neg p_1 \lor \neg p_2 \lor p_3$$
$$\neg p_2 \lor \neg p_3$$

DIMACS input format:

```
p cnf 3 4
1 0
-1 2 0
-1 -2 3 0
-2 -3 0
```

3 variables, 4 clauses.

$\neg p_1 \lor \neg p_2 \lor p_3$

Solving Sudoku with `minisat`

How to run:

```
minisat sudoku.sat sudoku.out
```

4	1	8	7	9	5	6	3	2
6	3	9	8	2	1	5	7	4
5	2	7	3	6	4	1	8	9
9	5	2	4	3	6	8	1	7
1	4	6	5	8	7	9	2	3
7	8	3	2	1	9	4	5	6
2	6	1	9	5	3	7	4	8
3	7	5	6	4	8	2	9	1
8	9	4	1	7	2	3	6	5

End of Lecture 7

Slides for lecture 7 end here . . .