# Logic and Mechanized Reasoning

## Normal Forms

**Marijn J.H. Heule**

Carnegie Mellon University

Complete Sets of Connectives

Negation Normal Form

Disjunctive Normal Form

Conjunctive Normal Form

Complete Sets of Connectives

Negation Normal Form

Disjunctive Normal Form

Conjunctive Normal Form

# Complete Sets: OR and NOT

The chosen set of connectives has redundancies. The connectives can be replaced by other connectives:

$$A \leftrightarrow B \quad \equiv \quad (A \to B) \land (B \to A)$$

# Complete Sets: OR and NOT

The chosen set of connectives has redundancies. The connectives can be replaced by other connectives:

$$
\begin{aligned}
A \leftrightarrow B &\equiv (A \rightarrow B) \wedge (B \rightarrow A) \\
A \rightarrow B &\equiv \neg A \vee B
\end{aligned}
$$

# Complete Sets: OR and NOT

The chosen set of connectives has redundancies. The connectives can be replaced by other connectives:

$$
\begin{aligned}
A \leftrightarrow B &\equiv (A \rightarrow B) \wedge (B \rightarrow A) \\
A \rightarrow B &\equiv \neg A \vee B \\
A \wedge B &\equiv \neg(\neg A \vee \neg B)
\end{aligned}
$$

# Complete Sets: OR and NOT

The chosen set of connectives has redundancies. The connectives can be replaced by other connectives:

$$\begin{aligned}
A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\
A \to B &\equiv \neg A \vee B \\
A \wedge B &\equiv \neg(\neg A \vee \neg B) \\
\bot &\equiv \neg \top
\end{aligned}$$

# Complete Sets: OR and NOT

The chosen set of connectives has redundancies. The connectives can be replaced by other connectives:

$$
\begin{aligned}
A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\
A \to B &\equiv \neg A \vee B \\
A \wedge B &\equiv \neg(\neg A \vee \neg B) \\
\bot &\equiv \neg \top \\
\top &\equiv p \vee \neg p
\end{aligned}
$$

# Complete Sets: OR and NOT

The chosen set of connectives has redundancies. The connectives can be replaced by other connectives:

$$\begin{aligned} A \leftrightarrow B &\equiv (A \rightarrow B) \wedge (B \rightarrow A) \\ A \rightarrow B &\equiv \neg A \vee B \\ A \wedge B &\equiv \neg(\neg A \vee \neg B) \\ \bot &\equiv \neg \top \\ \top &\equiv p \vee \neg p \end{aligned}$$

A set of connectives is complete if it can express all Boolean functions

# Complete Sets: AND and NOT

Now let's do the same for AND and NOT:

$$A \leftrightarrow B \quad \equiv$$

# Complete Sets: AND and NOT

Now let's do the same for AND and NOT:

$$\begin{aligned} A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\ A \to B &\equiv \end{aligned}$$

# Complete Sets: AND and NOT

Now let's do the same for AND and NOT:

$$\begin{aligned} A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\ A \to B &\equiv \neg A \vee B \\ A \vee B &\equiv \end{aligned}$$

# Complete Sets: AND and NOT

Now let's do the same for AND and NOT:

$$\begin{aligned} A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\ A \to B &\equiv \neg A \vee B \\ A \vee B &\equiv \neg(\neg A \wedge \neg B) \\ \top &\equiv \end{aligned}$$

# Complete Sets: AND and NOT

Now let's do the same for AND and NOT:

$$
\begin{aligned}
A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\
A \to B &\equiv \neg A \vee B \\
A \vee B &\equiv \neg(\neg A \wedge \neg B) \\
\top &\equiv \neg \bot \\
\bot &\equiv
\end{aligned}
$$

# Complete Sets: AND and NOT

Now let's do the same for AND and NOT:

$$\begin{aligned} A \leftrightarrow B &\equiv (A \to B) \wedge (B \to A) \\ A \to B &\equiv \neg A \vee B \\ A \vee B &\equiv \neg(\neg A \wedge \neg B) \\ \top &\equiv \neg \bot \\ \bot &\equiv p \wedge \neg p \end{aligned}$$

Complete Sets of Connectives

## Negation Normal Form

Disjunctive Normal Form

Conjunctive Normal Form

# Negation Normal Form: Introduction

The set of propositional formulas in negation normal form (NNF) is generated inductively as follows:

- Each variable $p_i$ is in negation normal form.
- The negation $\neg p_i$ of a propositional variable is in negation normal form.
- $\top$ and $\bot$ are in negation normal form.
- If $A$ and $B$ are in negation normal form, then so are $A \wedge B$ and $A \vee B$.

# Negation Normal Form: Introduction

The set of propositional formulas in negation normal form (NNF) is generated inductively as follows:

- Each variable $p_i$ is in negation normal form.
- The negation $\neg p_i$ of a propositional variable is in negation normal form.
- $\top$ and $\bot$ are in negation normal form.
- If $A$ and $B$ are in negation normal form, then so are $A \wedge B$ and $A \vee B$.

Example (Which formulas are in NNF?)

- $p \vee (q \wedge \neg p)$
- $p \to q$
- $\neg A \wedge (B \vee A)$

# Negation Normal Form: Recall Harder Example

Recall: For any propositional variables $p$, $q$, and $r$, we have
$\neg((p \vee q) \wedge (q \to r)) \equiv (\neg p \vee q) \wedge (\neg p \vee \neg r) \wedge (\neg q \vee \neg r)$.

Proof.

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

**Which formulas are in NNF?** $\square$

# Negation Normal Form: Lemma

**Lemma**

*Every propositional formula is equivalent to one in negation normal form.*

**Proof.**

First use the identities $A \leftrightarrow B \equiv (A \to B) \wedge (B \to A)$ and $A \to B \equiv \neg A \vee B$ to get rid of $\leftrightarrow$ and $\to$. Then use De Morgan's laws together with $\neg\neg A \equiv A$, $\neg\top \equiv \bot$, and $\neg\top \equiv \bot$ to push negations down to the atomic formulas. $\square$

Complete Sets of Connectives

Negation Normal Form

## Disjunctive Normal Form

Conjunctive Normal Form

# Disjunctive Normal Form: Introduction

A literal is a propositional variable $p$ or its negation $\neg p$.

A propositional formula is in Disjunctive Normal Form (DNF) if it is written as a disjunction of conjunctions of literals.

$$\bigvee_{i<n} \left( \bigwedge_{j<m_i} (\neg) p_{i,j} \right)$$

# Disjunctive Normal Form: Introduction

A literal is a propositional variable $p$ or its negation $\neg p$.

A propositional formula is in Disjunctive Normal Form (DNF) if it is written as a disjunction of conjunctions of literals.

$$\bigvee_{i<n} \left( \bigwedge_{j<m_i} (\neg) p_{i,j} \right)$$

A conjunction of literals is called a cube. $\top$ is the empty cube.

# Disjunctive Normal Form: Introduction

A literal is a propositional variable $p$ or its negation $\neg p$.

A propositional formula is in Disjunctive Normal Form (DNF) if it is written as a disjunction of conjunctions of literals.

$$\bigvee_{i<n} \left( \bigwedge_{j<m_i} (\neg) p_{i,j} \right)$$

A conjunction of literals is called a cube. $\top$ is the empty cube.

Example (Which formulas are in DNF?)

- $p \vee q$
- $p \wedge q$
- $(p \wedge q) \vee \neg(p \wedge q)$

# Disjunctive Normal Form: Lemma

**Lemma**

*The conjunction of two DNF formulas is equivalent to a DNF formula.*

# Disjunctive Normal Form: Lemma

Lemma

*The conjunction of two DNF formulas is equivalent to a DNF formula.*

Proof.

True. Recall that $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$.

# Disjunctive Normal Form: Lemma

Lemma

*The conjunction of two DNF formulas is equivalent to a DNF formula.*

Proof.

True. Recall that $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$.

By induction on $n$, we have that for every sequence of formulas $B_0, \ldots, B_{n-1}$ we have $A \wedge \bigvee_{i<n} B_i \equiv \bigvee_{i<n}(A \wedge B_i)$.

# Disjunctive Normal Form: Lemma

Lemma

*The conjunction of two DNF formulas is equivalent to a DNF formula.*

Proof.

True. Recall that $A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$.

By induction on $n$, we have that for every sequence of formulas $B_0, \ldots, B_{n-1}$ we have $A \wedge \bigvee_{i<n} B_i \equiv \bigvee_{i<n}(A \wedge B_i)$.

Then by induction on $n'$ we have
$\bigvee_{i'<n'} A_{i'} \wedge \bigvee_{i<n} B_i \equiv \bigvee_{i'<n'} \bigvee_{i<n}(A_{i'} \wedge B_i)$.

Since each $A_{i'}$ and each $B_i$ is a conjunction of literals, this yields the result. □

# Disjunctive Normal Form: Proposition 1

Proposition

*Every propositional formula is equivalent to one in disjunctive normal form.*

True or false?

# Disjunctive Normal Form: Proposition 1

**Proposition**

*Every propositional formula is equivalent to one in disjunctive normal form.*

True or false?

**Proof.**

True. Since we already know that every formula is equivalent to one in negation normal form, we can use induction on that set of formulas. The claim is clearly true of $\top$, $\bot$, $p_i$, and $\neg p_i$. By the previous lemma, whenever it is true of $A$ and $B$, it is also true of $A \vee B$. $\square$

# Disjunctive Normal Form: Proposition 2

**Proposition**

*For every DNF formula $A$ one can determine satisfiability and unsatisfiability in linear time.*

True or false?

# Disjunctive Normal Form: Proposition 2

**Proposition**

*For every DNF formula $A$ one can determine satisfiability and unsatisfiability in linear time.*

True or false?

**Proof.**

True. A cube with a pair of complementary literals $p_i$ and $\neg p_i$ is equal to $\perp$. Computing whether a cube is equal to $\perp$ can be done in linear time. A formula is satisfiable if $A$ contains at least one cube that is not equal to $\perp$ and unsatisfiable otherwise.

□

## Disjunctive Normal Form: Diplomacy Problem

"You are chief of protocol for the embassy ball. The crown prince instructs you either to invite *Peru* or to exclude *Qatar*. The queen asks you to invite either *Qatar* or *Romania* or both. The king, in a spiteful mood, wants to snub either *Romania* or *Peru* or both. Is there a guest list that will satisfy the whims of the entire royal family?"

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

**How to convert this into DNF?**

# Disjunctive Normal Form: Truth Table to DNF

$$\Gamma = (p \vee \neg q) \wedge (q \vee r) \wedge (\neg r \vee \neg p)$$

| $\tau(p)$ | $\tau(q)$ | $\tau(r)$ | falsifies | $[\![\Gamma]\!]_\tau$ |
|---|---|---|---|---|
| $\bot$ | $\bot$ | $\bot$ | $(q \vee r)$ | $\bot$ |
| $\bot$ | $\bot$ | $\top$ | — | $\top$ |
| $\bot$ | $\top$ | $\bot$ | $(p \vee \neg q)$ | $\bot$ |
| $\bot$ | $\top$ | $\top$ | $(p \vee \neg q)$ | $\bot$ |
| $\top$ | $\bot$ | $\bot$ | $(q \vee r)$ | $\bot$ |
| $\top$ | $\bot$ | $\top$ | $(\neg r \vee \neg p)$ | $\bot$ |
| $\top$ | $\top$ | $\bot$ | — | $\top$ |
| $\top$ | $\top$ | $\top$ | $(\neg r \vee \neg p)$ | $\bot$ |

# Disjunctive Normal Form: Truth Table to DNF

$$\Gamma = (p \vee \neg q) \wedge (q \vee r) \wedge (\neg r \vee \neg p)$$

| $\tau(p)$ | $\tau(q)$ | $\tau(r)$ | falsifies | $[\![\Gamma]\!]_\tau$ |
|---|---|---|---|---|
| $\bot$ | $\bot$ | $\bot$ | $(q \vee r)$ | $\bot$ |
| $\bot$ | $\bot$ | $\top$ | — | $\top$ |
| $\bot$ | $\top$ | $\bot$ | $(p \vee \neg q)$ | $\bot$ |
| $\bot$ | $\top$ | $\top$ | $(p \vee \neg q)$ | $\bot$ |
| $\top$ | $\bot$ | $\bot$ | $(q \vee r)$ | $\bot$ |
| $\top$ | $\bot$ | $\top$ | $(\neg r \vee \neg p)$ | $\bot$ |
| $\top$ | $\top$ | $\bot$ | — | $\top$ |
| $\top$ | $\top$ | $\top$ | $(\neg r \vee \neg p)$ | $\bot$ |

The DNF consists of all assignments that satisfy the formula:

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

$$(p \vee \neg q) \wedge (q \vee r) \wedge (\neg r \vee \neg p) \quad \equiv$$

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

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

# Disjunctive Normal Form: Applying Distributive Laws

An alternative approach is applying the distributive laws

$$
\begin{aligned}
(p \vee \neg q) \wedge (q \vee r) \wedge (\neg r \vee \neg p) &\equiv \\
(p \wedge (q \vee r)) \vee (\neg q \wedge (q \vee r)) \wedge (\neg r \vee \neg p) &\equiv \\
(p \wedge q) \vee (p \wedge r) \vee (\neg q \wedge q) \vee (\neg q \wedge r) \wedge (\neg r \vee \neg p) &\equiv \\
(p \wedge q) \vee (p \wedge r) \vee \bot \vee (\neg q \wedge r) \wedge (\neg r \vee \neg p) &\equiv \\
(p \wedge q) \vee (p \wedge r) \vee (\neg q \wedge r) \wedge (\neg r \vee \neg p) &\equiv \\
(\neg r \wedge p \wedge q) \vee (\neg r \wedge p \wedge r) \vee (\neg r \wedge \neg q \wedge r) \vee & \\
(\neg p \wedge p \wedge q) \vee (\neg p \wedge p \wedge r) \vee (\neg p \wedge \neg q \wedge r) &\equiv \\
(\neg r \wedge p \wedge q) \vee \bot \vee \bot \vee \bot \vee \bot \vee (\neg p \wedge \neg q \wedge r) &\equiv \\
(\neg r \wedge p \wedge q) \vee (\neg p \wedge \neg q \wedge r). &
\end{aligned}
$$

# Disjunctive Normal Form: Complexity

What is the worst case cost of applying the distributive laws?

# Disjunctive Normal Form: Complexity

What is the worst case cost of applying the distributive laws?

In some cases, converting a formula to DNF can have an exponential explosion on the size of the formula.

If we convert $(p_1 \vee q_1) \wedge (p_2 \vee q_2) \wedge \ldots \wedge (p_n \vee q_n)$ using the distributive laws to DNF:

$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \vee (q_1 \wedge p_2 \wedge \ldots \wedge p_n) \vee \ldots \vee$
$(q_1 \wedge q_2 \wedge \ldots \wedge q_n)$

Complete Sets of Connectives

Negation Normal Form

Disjunctive Normal Form

Conjunctive Normal Form

# Conjunctive Normal Form: Introduction

A literal is a propositional variable $p$ or its negation $\neg p$.

A propositional formula is in Conjunctive Normal Form (CNF) if it is written as a conjunction of disjunctions of literals.

$$\bigwedge_{i<n} \left( \bigvee_{j<m_i} (\neg) p_{i,j} \right)$$

# Conjunctive Normal Form: Introduction

A literal is a propositional variable $p$ or its negation $\neg p$.

A propositional formula is in Conjunctive Normal Form (CNF) if it is written as a conjunction of disjunctions of literals.

$$\bigwedge_{i<n} \left( \bigvee_{j<m_i} (\neg) p_{i,j} \right)$$

A clause is a disjunction of literals. $\perp$ denotes the empty clause.

# Conjunctive Normal Form: Introduction

A literal is a propositional variable $p$ or its negation $\neg p$.

A propositional formula is in Conjunctive Normal Form (CNF) if it is written as a conjunction of disjunctions of literals.

$$\bigwedge_{i<n} \left( \bigvee_{j<m_i} (\neg) p_{i,j} \right)$$

A clause is a disjunction of literals. $\perp$ denotes the empty clause.

Example (Which formulas are in CNF?)

- $p \vee q$
- $p \wedge q$
- $(p \vee q) \wedge \neg(p \vee q)$

# Conjunctive Normal Form: Proposition

Proposition

*For every CNF formula $A$ one can determine whether it is valid in linear time.*

True or false?

# Conjunctive Normal Form: Proposition

Proposition

*For every CNF formula $A$ one can determine whether it is valid in linear time.*

True or false?

Proof.

True. A clause with a pair of complementary literals $p_i$ and $\neg p_i$ is equal to $\top$. Computing whether a clause is equal to $\top$ can be done in linear time. A formula is valid if and only if all clauses are equal to $\top$.

□

# Conjunctive Normal Form: Input Form of Reasoning Tools

Most reasoning tools for propositional logic require CNF input

- Transforming a formula to CNF can also be exponential...
- But, it can be avoided by focusing on equisatisfiability.
- The performance of solvers depend on the transformation.
- Typically the smaller the CNF, the easier to solve it.

# Conjunctive Normal Form: Input Form of Reasoning Tools

Most reasoning tools for propositional logic require CNF input

- Transforming a formula to CNF can also be exponential...
- But, it can be avoided by focusing on equisatisfiability.
- The performance of solvers depend on the transformation.
- Typically the smaller the CNF, the easier to solve it.

Let's look at transforming common constraints into CNF

# Conjunctive Normal Form: AtLeastOne

Given a set of propositions $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \geq 1$$

in CNF?

Hint: This is easy...

# Conjunctive Normal Form: AtLeastOne

Given a set of propositions $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \geq 1$$

in CNF?

Hint: This is easy...

$$(p_1 \vee p_2 \vee \cdots \vee p_n)$$

# Conjunctive Normal Form: Parity Constraints

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

# Conjunctive Normal Form: Parity Constraints

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

$p_1 \oplus \cdots \oplus p_n = 1$ is *true* if and only if an odd number of $p_i$ is assigned to *true*. Consider the case with two literals:

| $\tau(p_1)$ | $\tau(p_2)$ | $[\![p_1 \oplus p_2 = 1]\!]_\tau$ |
|---|---|---|
| $\bot$ | $\bot$ | $\bot$ |
| $\bot$ | $\top$ | $\top$ |
| $\top$ | $\bot$ | $\top$ |
| $\top$ | $\top$ | $\bot$ |

# Conjunctive Normal Form: Parity Constraints

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

$p_1 \oplus \cdots \oplus p_n = 1$ is *true* if and only if an odd number of $p_i$ is assigned to *true*. Consider the case with two literals:

| $\tau(p_1)$ | $\tau(p_2)$ | $[\![p_1 \oplus p_2 = 1]\!]_\tau$ |
|---|---|---|
| $\perp$ | $\perp$ | $\perp$ |
| $\perp$ | $\top$ | $\top$ |
| $\top$ | $\perp$ | $\top$ |
| $\top$ | $\top$ | $\perp$ |

$$(p_1 \vee p_2) \wedge (\neg p_1 \vee \neg p_2)$$

# Conjunctive Normal Form: Exponential Transformation

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

The direct encoding requires $2^{n-1}$ clauses of length $n$:

$$\bigwedge_{\text{even } \#\neg} ((\neg)p_1 \vee (\neg)p_2 \vee \cdots \vee (\neg)p_n)$$

# Conjunctive Normal Form: Exponential Transformation

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

The direct encoding requires $2^{n-1}$ clauses of length $n$:

$$\bigwedge_{\text{even } \#\neg} ((\neg)p_1 \vee (\neg)p_2 \vee \cdots \vee (\neg)p_n)$$

$$\begin{aligned} p_1 \oplus p_2 \oplus p_3 = 1 \quad \leftrightarrow \quad & (p_1 \vee p_2 \vee p_3) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge \\ & (\neg p_1 \vee p_2 \vee \neg p_3) \wedge (p_1 \vee \neg p_2 \vee \neg p_3) \end{aligned}$$

# Conjunctive Normal Form: Exponential Transformation

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

The direct encoding requires $2^{n-1}$ clauses of length $n$:

$$\bigwedge_{\text{even } \#\neg} ((\neg)p_1 \vee (\neg)p_2 \vee \cdots \vee (\neg)p_n)$$

$$\begin{aligned} p_1 \oplus p_2 \oplus p_3 = 1 \quad \leftrightarrow \quad & (p_1 \vee p_2 \vee p_3) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge \\ & (\neg p_1 \vee p_2 \vee \neg p_3) \wedge (p_1 \vee \neg p_2 \vee \neg p_3) \end{aligned}$$

Question: How many assignments satisfy this formula?

# Conjunctive Normal Form: Exponential Transformation

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

The direct encoding requires $2^{n-1}$ clauses of length $n$:

$$\bigwedge_{\text{even } \#\neg} ((\neg)p_1 \vee (\neg)p_2 \vee \cdots \vee (\neg)p_n)$$

$$p_1 \oplus p_2 \oplus p_3 = 1 \quad \leftrightarrow \quad (p_1 \vee p_2 \vee p_3) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (\neg p_1 \vee p_2 \vee \neg p_3) \wedge (p_1 \vee \neg p_2 \vee \neg p_3)$$

Question: How many assignments satisfy this formula? 4

# Conjunctive Normal Form: Exponential Transformation

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

The direct encoding requires $2^{n-1}$ clauses of length $n$:

$$\bigwedge_{\text{even } \#\neg} ((\neg)p_1 \vee (\neg)p_2 \vee \cdots \vee (\neg)p_n)$$

Can we encode large parity constraints with less clauses?

# Conjunctive Normal Form: Exponential Transformation

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 \oplus \cdots \oplus p_n = 1$$

in CNF?

The direct encoding requires $2^{n-1}$ clauses of length $n$:

$$\bigwedge_{\text{even } \#\neg} ((\neg)p_1 \vee (\neg)p_2 \vee \cdots \vee (\neg)p_n)$$

Can we encode large parity constraints with less clauses?

Compact: $(p_1 \oplus p_2 \oplus p_3 \oplus q = 1) \wedge (\neg q \oplus p_4 \oplus \cdots \oplus p_n = 1)$

Tradeoff: increase the number of variables but decreases the number of clauses!

# Conjunctive Normal Form: AtMostOne Pairwise Encoding

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \leq 1$$

in CNF?

# Conjunctive Normal Form: AtMostOne Pairwise Encoding

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \leq 1$$

in CNF?

The direct encoding requires $n(n-1)/2$ binary clauses:

$$\bigwedge_{1 \leq i < j \leq n} (\neg p_i \vee \neg p_j)$$

# Conjunctive Normal Form: AtMostOne Pairwise Encoding

Given a set of Boolean variables $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \leq 1$$

in CNF?

The direct encoding requires $n(n-1)/2$ binary clauses:

$$\bigwedge_{1 \leq i < j \leq n} (\neg p_i \vee \neg p_j)$$

Is it possible to use fewer clauses?

# Conjunctive Normal Form: AtMostOne Linear Encoding

Given a set of propositions $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \leq 1$$

in CNF using a linear number of binary clauses?

# Conjunctive Normal Form: AtMostOne Linear Encoding

Given a set of propositions $p_1, \ldots, p_n$, how to express

$$p_1 + \cdots + p_n \leq 1$$

in CNF using a linear number of binary clauses?

Split the constraint using additional variables: Apply the direct encoding if $n \leq 4$ otherwise replace $p_1 + \cdots + p_n \leq 1$ by

$$(p_1 + p_2 + p_3 + q \leq 1) \land (\neg q + p_4 + \cdots + p_n \leq 1)$$

resulting in $3n - 6$ clauses and $(n-3)/2$ new variables.

# Conjunctive Normal Form: Order Matters

Split the constraint using additional variables: Apply the direct encoding if $n \leq k$ otherwise replace $p_1 + \cdots + p_n \leq 1$ by

$$A : (p_1 + \cdots + p_k + q \leq 1) \wedge (\neg q + p_{k+1} + \cdots + p_n \leq 1)$$

$$B : (p_1 + \cdots + p_k + q \leq 1) \wedge (p_{k+1} + \cdots + p_n + \neg q \leq 1)$$

Is there a difference?

# Conjunctive Normal Form: Order Matters

Split the constraint using additional variables: Apply the direct encoding if $n \leq k$ otherwise replace $p_1 + \cdots + p_n \leq 1$ by

$$A : (p_1 + \cdots + p_k + q \leq 1) \wedge (\neg q + p_{k+1} + \cdots + p_n \leq 1)$$

$$B : (p_1 + \cdots + p_k + q \leq 1) \wedge (p_{k+1} + \cdots + p_n + \neg q \leq 1)$$

Is there a difference?

$$\begin{aligned} A : \quad & (p_1 + p_2 + q_1 \leq 1) \wedge (\neg q_1 + p_3 + q_2 \leq 1) \wedge \\ & (\neg q_2 + p_4 + q_3 \leq 1) \wedge (\neg q_3 + p_5 + p_6 \leq 1) \end{aligned}$$

$$\begin{aligned} B : \quad & (p_1 + p_2 + q_1 \leq 1) \wedge (p_3 + p_4 + q_2 \leq 1) \wedge \\ & (p_5 + p_6 + q_3 \leq 1) \wedge (\neg q_1 + \neg q_2 + \neg q_3 \leq 1) \end{aligned}$$

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \lor \neg p_2$ | $\neg p_1 \lor q$ |
| | $\neg q \lor \neg p_2$ |

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \vee \neg p_2$ | $\neg p_1 \vee q$ |
| | $\neg q \vee \neg p_2$ |

Question: Is $A$ equivalent to $B$?

Note: $A \leftrightarrow B$ if $\neg A \wedge B$ and $A \wedge \neg B$ are unsatisfiable.

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \vee \neg p_2$ | $\neg p_1 \vee q$ |
| | $\neg q \vee \neg p_2$ |

Is $\neg A \wedge B$ unsatisfiable?

Note: $\neg A \equiv p_1 \wedge p_2$

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \vee \neg p_2$ | $\neg p_1 \vee q$ |
| | $\neg q \vee \neg p_2$ |

Is $\neg A \wedge B$ unsatisfiable? yes!

Note: $\neg A \equiv p_1 \wedge p_2$

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \vee \neg p_2$ | $\neg p_1 \vee q$ |
| | $\neg q \vee \neg p_2$ |

Is $A \wedge \neg B$ unsatisfiable?

Note: $\neg B \equiv (p_1 \vee q) \wedge (p_1 \vee p_2) \wedge (\overline{q} \vee p_2)$

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \vee \neg p_2$ | $\neg p_1 \vee q$ |
| | $\neg q \vee \neg p_2$ |

Is $A \wedge \neg B$ unsatisfiable? no!

Note: $\neg B \equiv (p_1 \vee q) \wedge (p_1 \vee p_2) \wedge (\overline{q} \vee p_2)$

# Conjunctive Normal Form: AtMostOne Equivalence

Are these two formulas of $p_1 + p_2 \leq 1$ equivalent?

| $A$ (direct encoding) | $B$ (split encoding) |
|---|---|
| $\neg p_1 \vee \neg p_2$ | $\neg p_1 \vee q$ |
| | $\neg q \vee \neg p_2$ |

$A$ and $B$ are equisatisfiable:

- $A$ is satisfiable iff $B$ is satisfiable.

Note: Equisatisfiability is weaker than equivalence but useful if all we want to do is determine satisfiability.