CS:4980 Topics in Computer Science II

Introduction to Automated Reasoning

Normal Forms in Propositional Logic

Cesare Tinelli

Spring 2024

The University of Iowa

Credits

These slides are based on slides originally developed by **Cesare Tinelli** at the University of Iowa, **Andrei Voronkov** at the University of Manchester, **Emina Torlak** at the University of Washington, and by **Clark Barrett**, **Caroline Trippel**, and **Andrew (Haoze) Wu** at Stanford University. Adapted by permission.

Agenda

- NNF, DNF, CNF (CC Ch. 1.6)

Normal forms

For AR purposes, the language of formulas used to model problems may be too large

AR systems usually transform input formulas to formulas in a more restricted format before reasoning about them

We call these formats *normal forms*

The normal form a formula α is usually logically equivalent to, or at least equisatisfiable with, α

Normal forms

For AR purposes, the language of formulas used to model problems may be too large

AR systems usually transform input formulas to formulas in a more restricted format before reasoning about them

We call these formats *normal forms*

The normal form a formula α is usually logically equivalent to, or at least equisatisfiable with, α

Normal forms

For AR purposes, the language of formulas used to model problems may be too large

AR systems usually transform input formulas to formulas in a more restricted format before reasoning about them

We call these formats *normal forms*

The normal form a formula α is usually logically equivalent to, or at least equisatisfiable with, α

Normal forms

For AR purposes, the language of formulas used to model problems may be too large

AR systems usually transform input formulas to formulas in a more restricted format before reasoning about them

We call these formats *normal forms*

The normal form a formula α is usually logically equivalent to, or at least equisatisfiable with, α

Normal forms for propositional logic

These three normal forms are often used:

- Negation normal form (NNF)
- Disjunctive normal form (DNF)
- Conjunctive normal form (CNF)

Every formula of PL can be converted to an equivalent formula in one of these forms

Normal forms for propositional logic

These three normal forms are often used:

- Negation normal form (NNF)
- Disjunctive normal form (DNF)
- Conjunctive normal form (CNF)

Every formula of PL can be converted to an equivalent formula in one of these forms

Negation normal form (NNF)

- Only logical connectives: $\wedge$, $\vee$, and $\neg$
- $\neg$ only appears in literals

Grammar

$$
\begin{aligned}
\langle \text{Atom} \rangle &::= \top \mid \bot \mid \langle \text{Variable} \rangle \\
\langle \text{Literal} \rangle &::= \langle \text{Atom} \rangle \mid \neg \langle \text{Atom} \rangle \\
\langle \text{Formula} \rangle &::= \langle \text{Literal} \rangle \mid \langle \text{Formula} \rangle \vee \langle \text{Formula} \rangle \mid \langle \text{Formula} \rangle \wedge \langle \text{Formula} \rangle
\end{aligned}
$$

Negation normal form (NNF)

- Only logical connectives: $\wedge$, $\vee$, and $\neg$
- $\neg$ only appears in literals

Grammar

$$
\begin{aligned}
\langle \text{Atom} \rangle &:= \top \mid \bot \mid \langle \text{Variable} \rangle \\
\langle \text{Literal} \rangle &:= \langle \text{Atom} \rangle \mid \neg \langle \text{Atom} \rangle \\
\langle \text{Formula} \rangle &:= \langle \text{Literal} \rangle \mid \langle \text{Formula} \rangle \vee \langle \text{Formula} \rangle \mid \langle \text{Formula} \rangle \wedge \langle \text{Formula} \rangle
\end{aligned}
$$

NNF transformation

Repeatedly apply the following rewrites ($\longrightarrow$) to the formula and its subformulas, in any order, to *completion*[1]

- Eliminate double implications: $\alpha \Leftrightarrow \beta \longrightarrow (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$
- Eliminate implications: $\alpha \Rightarrow \beta \longrightarrow (\neg\alpha \vee \beta)$
- Push negation inside conjunctions: $\neg(\alpha \wedge \beta) \longrightarrow (\neg\alpha \vee \neg\beta)$
- Push negation inside disjunctions: $\neg(\alpha \vee \beta) \longrightarrow (\neg\alpha \wedge \neg\beta)$
- Eliminate double negations: $\neg\neg\alpha \longrightarrow \alpha$
- Eliminate negated bottom: $\neg\bot \longrightarrow \top$
- Eliminate negated top: $\neg\top \longrightarrow \bot$

[1] I.e., until none applies anymore

NNF transformation

Repeatedly apply the following rewrites ($\longrightarrow$) to the formula and its subformulas, in any order, to *completion*[1]

- Eliminate double implications: $\alpha \Leftrightarrow \beta \longrightarrow (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$
- Eliminate implications: $\alpha \Rightarrow \beta \longrightarrow (\neg\alpha \vee \beta)$
- Push negation inside conjunctions: $\neg(\alpha \wedge \beta) \longrightarrow (\neg\alpha \vee \neg\beta)$
- Push negation inside disjunctions: $\neg(\alpha \vee \beta) \longrightarrow (\neg\alpha \wedge \neg\beta)$
- Eliminate double negations: $\neg\neg\alpha \longrightarrow \alpha$
- Eliminate negated bottom: $\neg\bot \longrightarrow \top$
- Eliminate negated top: $\neg\top \longrightarrow \bot$

[1] I.e., until none applies anymore

NNF transformation properties

Theorem 1

Every wff α not containing double implications ($\Leftrightarrow$) can be transformed into an equivalent NNF α' with a linear increase in the size[a] of the formula

[a] E.g., the number of variable occurrences or, equivalently, the number of subformulas

NNF transformation properties

Unfortunately, the NNF of formulas containing $\Leftrightarrow$ can grow exponentially larger in the worst case!

Example

$$(a_1 \Leftrightarrow a_2) \Leftrightarrow (a_3 \Leftrightarrow a_4) \qquad \text{4 vars}$$

$$\downarrow$$

$$(a_1 \Leftrightarrow a_2) \Rightarrow (a_3 \Leftrightarrow a_4) \wedge (a_3 \Leftrightarrow a_4) \Rightarrow (a_1 \Leftrightarrow a_2) \qquad \text{8 vars}$$

$$\downarrow$$

$$\vdots$$

$$\downarrow$$

$$((a_1 \Rightarrow a_2) \wedge (a_2 \Rightarrow a_1)) \Rightarrow ((a_3 \Rightarrow a_4) \wedge (a_4 \Rightarrow a_3))$$
$$\wedge \qquad \text{16 vars}$$
$$((a_3 \Rightarrow a_4) \wedge (a_4 \Rightarrow a_3)) \Rightarrow ((a_1 \Rightarrow a_2) \wedge (a_2 \Rightarrow a_1))$$

NNF transformation properties

Unfortunately, the NNF of formulas containing $\Leftrightarrow$ can grow exponentially larger in the worst case!

Example

$$(a_1 \Leftrightarrow a_2) \Leftrightarrow (a_3 \Leftrightarrow a_4) \qquad \text{4 vars}$$

$$\downarrow$$

$$(a_1 \Leftrightarrow a_2) \Rightarrow (a_3 \Leftrightarrow a_4) \land (a_3 \Leftrightarrow a_4) \Rightarrow (a_1 \Leftrightarrow a_2) \qquad \text{8 vars}$$

$$\downarrow$$

$$\vdots$$

$$\downarrow$$

$$\begin{array}{c} ((a_1 \Rightarrow a_2) \land (a_2 \Rightarrow a_1)) \Rightarrow ((a_3 \Rightarrow a_4) \land (a_4 \Rightarrow a_3)) \\ \land \\ ((a_3 \Rightarrow a_4) \land (a_4 \Rightarrow a_3)) \Rightarrow ((a_1 \Rightarrow a_2) \land (a_2 \Rightarrow a_1)) \end{array} \qquad \text{16 vars}$$

Disjunctive normal form (DNF)

- Formula is in NNF
- Formula is a disjunction of conjunctions of literals, i.e., of the form:

$$\bigvee_i \left(\bigwedge_j l_{ij} \right)$$

Grammar

$$\begin{aligned}
\langle \text{Atom} \rangle &:= \top \mid \bot \mid \langle \text{Variable} \rangle \\
\langle \text{Literal} \rangle &:= \langle \text{Atom} \rangle \mid \neg \langle \text{Atom} \rangle \\
\langle \text{Cube} \rangle &:= \langle \text{Literal} \rangle \mid \langle \text{Literal} \rangle \wedge \langle \text{Cube} \rangle \\
\langle \text{Formula} \rangle &:= \langle \text{Cube} \rangle \mid \langle \text{Cube} \rangle \vee \langle \text{Formula} \rangle
\end{aligned}$$

Disjunctive normal form (DNF)

- Formula is in NNF
- Formula is a disjunction of conjunctions of literals, i.e., of the form:

$$\bigvee_i \left(\bigwedge_j l_{ij} \right)$$

Grammar

$$\begin{aligned}
\langle \text{Atom} \rangle &:= \top \mid \bot \mid \langle \text{Variable} \rangle \\
\langle \text{Literal} \rangle &:= \langle \text{Atom} \rangle \mid \neg \langle \text{Atom} \rangle \\
\langle \text{Cube} \rangle &:= \langle \text{Literal} \rangle \mid \langle \text{Literal} \rangle \wedge \langle \text{Cube} \rangle \\
\langle \text{Formula} \rangle &:= \langle \text{Cube} \rangle \mid \langle \text{Cube} \rangle \vee \langle \text{Formula} \rangle
\end{aligned}$$

DNF transformation

Apply the following rewrites, in any order, to completion

- Apply NNF transformation rewrites
- Distribute $\wedge$ over $\vee$ (another source of exponential increase):
  - $\alpha \wedge (\beta \vee \gamma) \longrightarrow (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$
  - $(\alpha \vee \beta) \wedge \gamma \longrightarrow (\alpha \wedge \gamma) \vee (\beta \wedge \gamma)$
- Normalize nested conjunctions and disjunctions
  - $(\alpha \wedge \beta) \wedge \gamma \longrightarrow \alpha \wedge (\beta \wedge \gamma)$
  - $(\alpha \vee \beta) \vee \gamma \longrightarrow \alpha \vee (\beta \vee \gamma)$

Note: Instead of having nested conjunctions or disjunctions, it is convenient to treat $\wedge$ and $\vee$ as n-ary operators for any $n > 1$ (e.g., we treat $\alpha_1 \vee (\alpha_2 \vee (\alpha_3 \vee \alpha_4))$ as $\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4$)

DNF transformation

Apply the following rewrites, in any order, to completion

- Apply NNF transformation rewrites
- Distribute $\wedge$ over $\vee$ (another source of exponential increase):
  - $\alpha \wedge (\beta \vee \gamma) \longrightarrow (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$
  - $(\alpha \vee \beta) \wedge \gamma \longrightarrow (\alpha \wedge \gamma) \vee (\beta \wedge \gamma)$
- Normalize nested conjunctions and disjunctions
  - $(\alpha \wedge \beta) \wedge \gamma \longrightarrow \alpha \wedge (\beta \wedge \gamma)$
  - $(\alpha \vee \beta) \vee \gamma \longrightarrow \alpha \vee (\beta \vee \gamma)$

Note: Instead of having nested conjunctions or disjunctions, it is convenient to treat $\wedge$ and $\vee$ as n-ary operators for any $n > 1$ (e.g., we treat $a_1 \vee (a_2 \vee (a_3 \vee a_4))$ as $a_1 \vee a_2 \vee a_3 \vee a_4$)

DNF transformation

Theorem 2

Every wff α can be transformed into a logically equivalent DNF α', with a potentially exponential increase in the size of the formula

Note: The exponential increase can occur even in the absence of $\Leftrightarrow$

DNF transformation

Theorem 2
Every wff α can be transformed into a logically equivalent DNF α', with a potentially exponential increase in the size of the formula

Note: The exponential increase can occur even in the absence of $\Leftrightarrow$

Exercise

Transform each of these formulas (separately) into DNF:

$$\neg((p \lor \neg q) \Rightarrow r) \qquad \neg(a \Rightarrow (\neg b \Rightarrow a))$$

NNF transformation rewrites:

1. $\alpha \Leftrightarrow \beta \longrightarrow (\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)$
2. $\alpha \Rightarrow \beta \longrightarrow \neg\alpha \lor \beta$
3. $\neg(\alpha \lor \beta) \longrightarrow (\neg\alpha \land \neg\beta)$
4. $\neg(\alpha \land \beta) \longrightarrow (\neg\alpha \lor \neg\beta)$
5. $\neg\neg\alpha \longrightarrow \alpha$
6. $\neg\top \longrightarrow \bot$
7. $\neg\bot \longrightarrow \top$

DNF transformation rewrites:

1. $\alpha \land (\beta \lor \gamma) \longrightarrow (\alpha \land \beta) \lor (\alpha \land \gamma)$
2. $(\alpha \lor \beta) \land \gamma \longrightarrow (\alpha \land \gamma) \lor (\beta \land \gamma)$
3. $(\alpha \land \beta) \land \gamma \longrightarrow \alpha \land (\beta \land \gamma)$
4. $(\alpha \lor \beta) \lor \gamma \longrightarrow \alpha \lor (\beta \lor \gamma)$

Conjunctive normal form (CNF)

- Formula is in NNF
- Formula is a conjunction of disjunctions of literals, i.e., of the form:

$$\bigwedge_i \left(\bigvee_j l_{ij} \right)$$

Grammar

$$\begin{aligned}
\langle \text{Atom} \rangle &:= \top \mid \bot \mid \langle \text{Variable} \rangle \\
\langle \text{Literal} \rangle &:= \langle \text{Atom} \rangle \mid \neg \langle \text{Atom} \rangle \\
\langle \text{Clause} \rangle &:= \langle \text{Literal} \rangle \mid \langle \text{Literal} \rangle \vee \langle \text{Clause} \rangle \\
\langle \text{Formula} \rangle &:= \langle \text{Clause} \rangle \mid \langle \text{Clause} \rangle \wedge \langle \text{Formula} \rangle
\end{aligned}$$

Conjunctive normal form (CNF)

- Formula is in NNF
- Formula is a conjunction of disjunctions of literals, i.e., of the form:

$$\bigwedge_i \left(\bigvee_j l_{ij} \right)$$

Grammar

$$
\begin{aligned}
\langle \text{Atom} \rangle &:= \top \mid \bot \mid \langle \text{Variable} \rangle \\
\langle \text{Literal} \rangle &:= \langle \text{Atom} \rangle \mid \neg \langle \text{Atom} \rangle \\
\langle \text{Clause} \rangle &:= \langle \text{Literal} \rangle \mid \langle \text{Literal} \rangle \vee \langle \text{Clause} \rangle \\
\langle \text{Formula} \rangle &:= \langle \text{Clause} \rangle \mid \langle \text{Clause} \rangle \wedge \langle \text{Formula} \rangle
\end{aligned}
$$

CNF transformation

Apply the following rewrites, in any order, to completion

- Apply NNF transformation rewrites
- Distribute $\vee$ over $\wedge$ (another source of exponential increase):
  - $\alpha \vee (\beta \wedge \gamma) \longrightarrow (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$
  - $(\alpha \wedge \beta) \vee \gamma \longrightarrow (\alpha \vee \gamma) \wedge (\beta \vee \gamma)$
- Normalize nested conjunctions and disjunctions
  - $(\alpha \wedge \beta) \wedge \gamma \longrightarrow \alpha \wedge (\beta \wedge \gamma)$
  - $(\alpha \vee \beta) \vee \gamma \longrightarrow \alpha \vee (\beta \vee \gamma)$

Exercise

Transform each of these formulas (separately) into CNF:

$$\neg((p \lor \neg q) \Rightarrow r) \qquad \neg(a \Rightarrow (\neg b \Rightarrow a))$$

NNF transformation rewrites:

1. $\alpha \Leftrightarrow \beta \longrightarrow (\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha)$
2. $\alpha \Rightarrow \beta \longrightarrow \neg\alpha \lor \beta$
3. $\neg(\alpha \lor \beta) \longrightarrow (\neg\alpha \land \neg\beta)$
4. $\neg(\alpha \land \beta) \longrightarrow (\neg\alpha \lor \neg\beta)$
5. $\neg\neg\alpha \longrightarrow \alpha$
6. $\neg\top \longrightarrow \bot$
7. $\neg\bot \longrightarrow \top$

CNF transformation rewrites:

1. $\alpha \lor (\beta \land \gamma) \longrightarrow (\alpha \lor \beta) \land (\alpha \lor \gamma)$
2. $(\alpha \land \beta) \lor \gamma \longrightarrow (\alpha \lor \gamma) \land (\beta \lor \gamma)$
3. $(\alpha \land \beta) \land \gamma \longrightarrow \alpha \land (\beta \land \gamma)$
4. $(\alpha \lor \beta) \lor \gamma \longrightarrow \alpha \lor (\beta \lor \gamma)$

CNF transformation

Theorem 3

Every wff α can be transformed into a logically equivalent CNF α', with a potentially exponential increase in the size of the formula

Note: The size increase can occur even in the absence of $\Leftrightarrow$

CNF transformation

Theorem 3

Every wff α can be transformed into a logically equivalent CNF α', with a potentially exponential increase in the size of the formula

Note: The size increase can occur even in the absence of $\Leftrightarrow$

CNF transformation can be exponential

There are formulas whose shortest CNF has an exponential size

Is there any way to avoid exponential blowup? Yes!

CNF transformation can be exponential

There are formulas whose shortest CNF has an exponential size

Is there any way to avoid exponential blowup? Yes!

CNF transformation can be exponential

There are formulas whose shortest CNF has an exponential size

Is there any way to avoid exponential blowup? Yes!

A space-efficient CNF transformation

Using so-called *naming*, *definition introduction*, or *Tseitin's transformation*

1. Take a non-literal subformula α of formula φ
2. Introduce a new *name* n for it, i.e., a fresh propositional variable
3. Add a *definition for* n, i.e., a formula stating that n is equivalent to α

$$\begin{aligned} \varphi \quad = \quad & p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}^{\alpha})))) \\ & n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{aligned}$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Using so-called *naming*, *definition introduction*, or *Tseitin's transformation*

1. Take a non-literal subformula α of formula φ
2. Introduce a new *name* n for it, i.e., a fresh propositional variable
3. Add a *definition for n*, i.e., a formula stating that n is equivalent to α

$$\begin{aligned}\varphi \quad = \quad & p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}^{\alpha})))) \\ & n \Leftrightarrow (p_5 \Leftrightarrow p_6)\end{aligned}$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Using so-called *naming*, *definition introduction*, or *Tseitin's transformation*

1. Take a non-literal subformula α of formula φ
2. Introduce a new *name* n for it, i.e., a fresh propositional variable
3. Add a *definition for n*, i.e., a formula stating that n is equivalent to α

$$\varphi \quad = \quad p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}^{\alpha}))))$$

$$n \Leftrightarrow (p_5 \Leftrightarrow p_6)$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Using so-called *naming*, *definition introduction*, or *Tseitin's transformation*

1. Take a non-literal subformula α of formula φ
2. Introduce a new *name* n for it, i.e., a fresh propositional variable
3. Add a *definition for* n, i.e., a formula stating that n is equivalent to α

$$
\begin{aligned}
\varphi \quad = \quad & p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}^{\alpha})))) \\
& n \Leftrightarrow (p_5 \Leftrightarrow p_6)
\end{aligned}
$$

$$
S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}
$$

A space-efficient CNF transformation

Using so-called *naming*, *definition introduction*, or *Tseitin's transformation*

1. Take a non-literal subformula α of formula φ
2. Introduce a new *name* n for it, i.e., a fresh propositional variable
3. Add a *definition for* n, i.e., a formula stating that n is equivalent to α

$$\begin{aligned} \varphi \quad &= \quad p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}^{\alpha})))) \\ & \quad\; n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{aligned}$$

4. Replace α in φ by its name n:

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Note: The new set S of formulas and the original formula φ are not equivalent but they are *equisatisfiable*:

$$\varphi = p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}))))$$

$$n \Leftrightarrow (p_5 \Leftrightarrow p_6)$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Note: The new set S of formulas and the original formula φ are not equivalent but they are *equisatisfiable*:

1. every interpretation satisfying S satisfies φ as well, and
2. every interpretation satisfying φ can be extended to one that satisfies S (by assigning to n the value of $p_5 \Leftrightarrow p_6$)

$$\varphi \;=\; p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}))))$$

$$n \Leftrightarrow (p_5 \Leftrightarrow p_6)$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Note: The new set S of formulas and the original formula φ are not equivalent but they are *equisatisfiable*:

1. every interpretation satisfying S satisfies φ as well, and
2. every interpretation satisfying φ can be extended to one that satisfies S (by assigning to n the value of $p_5 \Leftrightarrow p_6$)

$$\varphi = p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}^{n}))))$$

$$n \Leftrightarrow (p_5 \Leftrightarrow p_6)$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

A space-efficient CNF transformation

Note: The new set S of formulas and the original formula φ are not equivalent but they are *equisatisfiable*:

1. every interpretation satisfying S satisfies φ as well, and
2. every interpretation satisfying φ can be extended to one that satisfies S (by assigning to n the value of $p_5 \Leftrightarrow p_6$)

$$\varphi = p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow (\overbrace{p_5 \Leftrightarrow p_6}))))$$
$$n \Leftrightarrow (p_5 \Leftrightarrow p_6)$$

$$S = \left\{ \begin{array}{l} p_1 \Leftrightarrow (p_2 \Leftrightarrow (p_3 \Leftrightarrow (p_4 \Leftrightarrow n))) \\ n \Leftrightarrow (p_5 \Leftrightarrow p_6) \end{array} \right\}$$

After several steps

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

$$\begin{aligned} p_1 &\Leftrightarrow (p_2 \Leftrightarrow n_3) \\ n_3 &\Leftrightarrow (p_3 \Leftrightarrow n_4) \\ n_4 &\Leftrightarrow (p_4 \Leftrightarrow n_5) \\ n_5 &\Leftrightarrow (p_5 \Leftrightarrow p_6) \end{aligned}$$

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

After several steps

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

$$\begin{aligned} p_1 &\Leftrightarrow (p_2 \Leftrightarrow n_3) \\ n_3 &\Leftrightarrow (p_3 \Leftrightarrow n_4) \\ n_4 &\Leftrightarrow (p_4 \Leftrightarrow n_5) \\ n_5 &\Leftrightarrow (p_5 \Leftrightarrow p_6) \end{aligned}$$

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

After several steps

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

$$p_1 \Leftrightarrow (p_2 \Leftrightarrow n_3)$$
$$n_3 \Leftrightarrow (p_3 \Leftrightarrow n_4)$$
$$n_4 \Leftrightarrow (p_4 \Leftrightarrow n_5)$$
$$n_5 \Leftrightarrow (p_5 \Leftrightarrow p_6)$$

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 α: a set S_α of clauses which is satisfiable iff α is satisfiable

Clausal form of a set S of formulas: a set S' of clauses which is satisfiable iff so is S

Big advantage of clausal normal form over CNF:

we can convert any formula to a set of clauses in almost linear time

Clausal Form

Clausal form of a formula α: a set S_α of clauses which is satisfiable iff α is satisfiable

Clausal form of a set S *of formulas:* a set S' of clauses which is satisfiable iff so is S

Big advantage of clausal normal form over CNF:

we can convert any formula to a set of clauses in almost linear time

Clausal Form

Clausal form of a formula α: a set S_α of clauses which is satisfiable iff α is satisfiable

Clausal form of a set S of formulas: a set S' of clauses which is satisfiable iff so is S

Big advantage of clausal normal form over CNF:

we can convert any formula to a set of clauses in almost linear time

Definitional Clause Form Transformation

How to convert a formula α into a set S of clauses that is a clausal normal form of α:

1. If α has the form $C_1 \wedge \cdots \wedge C_n$, where $n \geq 1$ and each C_i is a clause, then

$$S := \{ C_1, \ldots, C_n \}$$

2. Otherwise, introduce a name for each subformula β of α that is not a literal, and use this name instead of β

Definitional Clause Form Transformation

How to convert a formula α into a set S of clauses that is a clausal normal form of α:

1. If α has the form $C_1 \wedge \cdots \wedge C_n$, where $n \geq 1$ and each C_i is a clause, then

$$S := \{ C_1, \ldots, C_n \}$$

2. Otherwise, introduce a name for each subformula β of α that is not a literal, and use this name instead of β

Definitional Clause Form Transformation

How to convert a formula α into a set S of clauses that is a clausal normal form of α:

1. If α has the form $C_1 \wedge \cdots \wedge C_n$, where $n \geq 1$ and each C_i is a clause, then

$$S := \{ C_1, \ldots, C_n \}$$

2. Otherwise, introduce a name for each subformula β of α that is not a literal, and use this name instead of β

Converting a formula to clausal form, Example

	non-literal subformula	definition	clauses
	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$		n_1
n_1	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$	$n_1 \Leftrightarrow \neg n_2$	$\neg n_1 \vee \neg n_2$
			$n_1 \vee n_2$
n_2	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r)$	$n_2 \Leftrightarrow (n_3 \Rightarrow n_7)$	$\neg n_2 \vee \neg n_3 \vee n_7$
			$n_3 \vee n_2$
			$\neg n_7 \vee n_2$
n_3	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r)$	$n_3 \Leftrightarrow (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$
			$\neg n_3 \vee n_5$
			$\neg n_4 \vee \neg n_5 \vee n_3$
n_4	$p \Rightarrow q$	$n_4 \Leftrightarrow (p \Rightarrow q)$	$\neg n_4 \vee \neg p \vee q$
			$p \vee n_4$
			$\neg q \vee n_4$
n_5	$p \wedge q \Rightarrow r$	$n_5 \Leftrightarrow (n_6 \Rightarrow r)$	$\neg n_5 \vee \neg n_6 \vee r$
			$n_6 \vee n_5$
			$\neg r \vee n_5$
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	$p \Rightarrow r$	$n_7 \Leftrightarrow (p \Rightarrow r)$	$\neg n_7 \vee \neg p \vee r$
			$p \vee n_7$
			$\neg r \vee n_7$

Converting a formula to clausal form, Example

	non-literal subformula	definition	clauses
	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$		n_1
n_1	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$	$n_1 \Leftrightarrow \neg n_2$	$\neg n_1 \vee \neg n_2$ $n_1 \vee n_2$
n_2	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r)$	$n_2 \Leftrightarrow (n_3 \Rightarrow n_7)$	$\neg n_2 \vee \neg n_3 \vee n_7$ $n_3 \vee n_2$ $\neg n_7 \vee n_2$
n_3	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r)$	$n_3 \Leftrightarrow (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$ $\neg n_3 \vee n_5$ $\neg n_4 \vee \neg n_5 \vee n_3$
n_4	$p \Rightarrow q$	$n_4 \Leftrightarrow (p \Rightarrow q)$	$\neg n_4 \vee \neg p \vee q$ $p \vee n_4$ $\neg q \vee n_4$
n_5	$p \wedge q \Rightarrow r$	$n_5 \Leftrightarrow (n_6 \Rightarrow r)$	$\neg n_5 \vee \neg n_6 \vee r$ $n_6 \vee n_5$ $\neg r \vee n_5$
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	$p \Rightarrow r$	$n_7 \Leftrightarrow (p \Rightarrow r)$	$\neg n_7 \vee \neg p \vee r$ $p \vee n_7$ $\neg r \vee n_7$

Consider all subformulas that are not literals

Converting a formula to clausal form, Example

	non-literal subformula	definition	clauses
	$\neg((p \Rightarrow q) \land (p \land q \Rightarrow r) \Rightarrow (p \Rightarrow r))$		n_1
n_1	$\neg((p \Rightarrow q) \land (p \land q \Rightarrow r) \Rightarrow (p \Rightarrow r))$	$n_1 \Leftrightarrow \neg n_2$	$\neg n_1 \lor \neg n_2$
			$n_1 \lor n_2$
n_2	$(p \Rightarrow q) \land (p \land q \Rightarrow r) \Rightarrow (p \Rightarrow r)$	$n_2 \Leftrightarrow (n_3 \Rightarrow n_7)$	$\neg n_2 \lor \neg n_3 \lor n_7$
			$n_3 \lor n_2$
			$\neg n_7 \lor n_2$
n_3	$(p \Rightarrow q) \land (p \land q \Rightarrow r)$	$n_3 \Leftrightarrow (n_4 \land n_5)$	$\neg n_3 \lor n_4$
			$\neg n_3 \lor n_5$
			$\neg n_4 \lor \neg n_5 \lor n_3$
n_4	$p \Rightarrow q$	$n_4 \Leftrightarrow (p \Rightarrow q)$	$\neg n_4 \lor \neg p \lor q$
			$p \lor n_4$
			$\neg q \lor n_4$
n_5	$p \land q \Rightarrow r$	$n_5 \Leftrightarrow (n_6 \Rightarrow r)$	$\neg n_5 \lor \neg n_6 \lor r$
			$n_6 \lor n_5$
			$\neg r \lor n_5$
n_6	$p \land q$	$n_6 \Leftrightarrow (p \land q)$	$\neg n_6 \lor p$
			$\neg n_6 \lor q$
			$\neg p \lor \neg q \lor n_6$
n_7	$p \Rightarrow r$	$n_7 \Leftrightarrow (p \Rightarrow r)$	$\neg n_7 \lor \neg p \lor r$
			$p \lor n_7$
			$\neg r \lor n_7$

Introduce names for these formulas

Converting a formula to clausal form, Example

	non-literal subformula	definition	clauses
	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$		n_1
n_1	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$	$n_1 \Leftrightarrow \neg n_2$	$\neg n_1 \vee \neg n_2$ $n_1 \vee n_2$
n_2	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r)$	$n_2 \Leftrightarrow (n_3 \Rightarrow n_7)$	$\neg n_2 \vee \neg n_3 \vee n_7$ $n_3 \vee n_2$ $\neg n_7 \vee n_2$
n_3	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r)$	$n_3 \Leftrightarrow (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$ $\neg n_3 \vee n_5$ $\neg n_4 \vee \neg n_5 \vee n_3$
n_4	$p \Rightarrow q$	$n_4 \Leftrightarrow (p \Rightarrow q)$	$\neg n_4 \vee \neg p \vee q$ $p \vee n_4$ $\neg q \vee n_4$
n_5	$p \wedge q \Rightarrow r$	$n_5 \Leftrightarrow (n_6 \Rightarrow r)$	$\neg n_5 \vee \neg n_6 \vee r$ $n_6 \vee n_5$ $\neg r \vee n_5$
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	$p \Rightarrow r$	$n_7 \Leftrightarrow (p \Rightarrow r)$	$\neg n_7 \vee \neg p \vee r$ $p \vee n_7$ $\neg r \vee n_7$

Introduce definitions

Converting a formula to clausal form, Example

	non-literal subformula	definition	clauses
	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$		n_1
n_1	$\neg((p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r))$	$n_1 \Leftrightarrow \neg n_2$	$\neg n_1 \vee \neg n_2$ $n_1 \vee n_2$
n_2	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r) \Rightarrow (p \Rightarrow r)$	$n_2 \Leftrightarrow (n_3 \Rightarrow n_7)$	$\neg n_2 \vee \neg n_3 \vee n_7$ $n_3 \vee n_2$ $\neg n_7 \vee n_2$
n_3	$(p \Rightarrow q) \wedge (p \wedge q \Rightarrow r)$	$n_3 \Leftrightarrow (n_4 \wedge n_5)$	$\neg n_3 \vee n_4$ $\neg n_3 \vee n_5$ $\neg n_4 \vee \neg n_5 \vee n_3$
n_4	$p \Rightarrow q$	$n_4 \Leftrightarrow (p \Rightarrow q)$	$\neg n_4 \vee \neg p \vee q$ $p \vee n_4$ $\neg q \vee n_4$
n_5	$p \wedge q \Rightarrow r$	$n_5 \Leftrightarrow (n_6 \Rightarrow r)$	$\neg n_5 \vee \neg n_6 \vee r$ $n_6 \vee n_5$ $\neg r \vee n_5$
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	$p \Rightarrow r$	$n_7 \Leftrightarrow (p \Rightarrow r)$	$\neg n_7 \vee \neg p \vee r$ $p \vee n_7$ $\neg r \vee n_7$

Convert the definition formulas to CNF in the standard way

DNF vs. CNF for satisfiability checking

DNF

- Satisfiability is decidable in linear time, with one traversal of the cubes
  - The DNF is unsatisfiable iff every cube contains both a literal and its complement
- However, converting to an equivalent DNF may result in exponential size increase

CNF

- Deciding satisfiability is hard (NP-hard)
- Converting to an equivalent CNF may result in exponential size increase
- However, converting into an *equisatisfiable* CNF can be done with only a linear size increase

DNF vs. CNF for satisfiability checking

DNF

- Satisfiability is decidable in linear time, with one traversal of the cubes
    - The DNF is unsatisfiable iff every cube contains both a literal and its complement
- However, converting to an equivalent DNF may result in exponential size increase

CNF

- Deciding satisfiability is hard (NP-hard)
- Converting to an equivalent CNF may result in exponential size increase
- However, converting into an *equisatisfiable* CNF can be done with only a linear size increase

DNF vs. CNF for satisfiability checking

DNF

- Satisfiability is decidable in linear time, with one traversal of the cubes
  - The DNF is unsatisfiable iff every cube contains both a literal and its complement
- However, converting to an equivalent DNF may result in exponential size increase

CNF

- Deciding satisfiability is hard (NP-hard)
- Converting to an equivalent CNF may result in exponential size increase
- However, converting into an *equisatisfiable* CNF can be done with only a linear size increase

DNF vs. CNF for satisfiability checking

DNF

- Satisfiability is decidable in linear time, with one traversal of the cubes
  - The DNF is unsatisfiable iff every cube contains both a literal and its complement
- However, converting to an equivalent DNF may result in exponential size increase

CNF

- Deciding satisfiability is hard (NP-hard)
- Converting to an equivalent CNF may result in exponential size increase
- However, converting into an *equisatisfiable* CNF can be done with only a linear size increase

DNF vs. CNF for satisfiability checking

DNF

- Satisfiability is decidable in linear time, with one traversal of the cubes
  - The DNF is unsatisfiable iff every cube contains both a literal and its complement
- However, converting to an equivalent DNF may result in exponential size increase

CNF

- Deciding satisfiability is hard (NP-hard)
- Converting to an equivalent CNF may result in exponential size increase
- However, converting into an *equisatisfiable* CNF can be done with only a linear size increase

DNF vs. CNF for satisfiability checking

DNF

- Satisfiability is decidable in linear time, with one traversal of the cubes
  - The DNF is unsatisfiable iff every cube contains both a literal and its complement
- However, converting to an equivalent DNF may result in exponential size increase

CNF

- Deciding satisfiability is hard (NP-hard)
- Converting to an equivalent CNF may result in exponential size increase
- However, converting into an *equisatisfiable* CNF can be done with only a linear size increase

DNF vs. CNF for satisfiability checking

Modern satisfiability checkers for PL expect CNF-like input

They choose to tackle the hardness of the satisfiability problem at runtime rather than at transformation time

DNF vs. CNF for satisfiability checking

Modern satisfiability checkers for PL expect CNF-like input

They choose to tackle the hardness of the satisfiability problem at runtime rather than at transformation time