CS:4350 Logic in Computer Science

Quantified Boolean Formulas

Cesare Tinelli

Spring 2022

The University of Iowa

Credits

These slides are largely based on slides originally developed by **Andrei Voronkov** at the University of Manchester. Adapted by permission.

Outline

Two-Player Games

Does she have a winning strategy?

Two-Player Games: The Satisfiability Game

Given: a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$

Two-Player Games: The Satisfiability Game

Given: a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$

There are two players: P and Q

Two-Player Games: The Satisfiability Game

Given: a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$

There are two players: P and Q

At round of the game k each player makes a move:

Two-Player Games: The Satisfiability Game

Given: a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$

There are two players: P and Q

At round of the game k each player makes a move:

1. player P can choose a value for variable p_k

Two-Player Games: The Satisfiability Game

Given: a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$

There are two players: P and Q

At round of the game k each player makes a move:

1. player P can choose a value for variable p_k
2. player Q can choose a value for variable q_k

Two-Player Games: The Satisfiability Game

Given: a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$

There are two players: P and Q

At round of the game k each player makes a move:

1. player P can choose a value for variable p_k
2. player Q can choose a value for variable q_k

Player P wins if after n rounds the chosen values satisfy formula G

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

G	Outcome

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	G has no p_i vars, P's choices are immaterial

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	G has no p_i vars, P's choices are immaterial
4.	$q_1 \to q_1$	

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	G has no p_i vars, P's choices are immaterial
4.	$q_1 \to q_1$	G is valid, P always wins!

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	G has no p_i vars, P's choices are immaterial
4.	$q_1 \to q_1$	G is valid, P always wins!
5.	$p_1 \wedge \neg p_1$	

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	G has no p_i vars, P's choices are immaterial
4.	$q_1 \to q_1$	G is valid, P always wins!
5.	$p_1 \wedge \neg p_1$	G is unsatisfiable, Q always wins!

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \rightarrow q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \rightarrow q_2$	G has no p_i vars, P's choices are immaterial
4.	$q_1 \rightarrow q_1$	G is valid, P always wins!
5.	$p_1 \wedge \neg p_1$	G is unsatisfiable, Q always wins!
6.	$p_1 \leftrightarrow q_1$	

Suppose Both Players Make no Mistakes: Who Wins?

Consider several special cases:

	G	Outcome
1.	p_1	P wins with $\{ p_1 \mapsto 1 \}$
2.	$p_1 \to q_1$	P wins with $\{ p_1 \mapsto 0 \}$
3.	$q_1 \to q_2$	G has no p_i vars, P's choices are immaterial
4.	$q_1 \to q_1$	G is valid, P always wins!
5.	$p_1 \wedge \neg p_1$	G is unsatisfiable, Q always wins!
6.	$p_1 \leftrightarrow q_1$	each move by P can be beaten by Q

Winning Strategy

Problem: does P have a winning strategy?

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
  for all moves of Q (values for q_1)

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
for all moves of Q (values for q_1)
there exists a move for P (a value for p_2) such that

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
  for all moves of Q (values for q_1)
    there exists a move for P (a value for p_2) such that
      for all moves of Q (values for q_2)

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
  for all moves of Q (values for q_1)
    there exists a move for P (a value for p_2) such that
      for all moves of Q (values for q_2)
        …

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
  for all moves of Q (values for q_1)
    there exists a move for P (a value for p_2) such that
      for all moves of Q (values for q_2)
        …
          there exists a move for P (a value for p_n) such that

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
  for all moves of Q (values for q_1)
    there exists a move for P (a value for p_2) such that
      for all moves of Q (values for q_2)
        ...
          there exists a move for P (a value for p_n) such that
            for all moves of Q (values for q_n)

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
  for all moves of Q (values for q_1)
    there exists a move for P (a value for p_2) such that
      for all moves of Q (values for q_2)
        …
          there exists a move for P (a value for p_n) such that
            for all moves of Q (values for q_n)
              the formula G is satisfiable

Winning Strategy

Problem: does P have a winning strategy?

P has a winning strategy
iff
there exists a move for P (a value for p_1) such that
for all moves of Q (values for q_1)
there exists a move for P (a value for p_2) such that
for all moves of Q (values for q_2)
...
there exists a move for P (a value for p_n) such that
for all moves of Q (values for q_n)
the formula G is satisfiable

The existence of a winning strategy can be expressed by the *quantified Boolean formula*

$$\exists p_1 \forall q_1 \exists p_2 \forall q_2 \cdots \exists p_n \forall q_n \, G$$

Quantified Boolean Formulas

Propositional Formula:

- Every Boolean variable is a (propositional) formula
- $\top$ and $\bot$ are formulas
- If F is a formula, then $\neg F$ is a formula
- If $F_1, \ldots, F_n$ are formulas, where $n \geq 2$,
  then $F_1 \wedge \cdots \wedge F_n$ and $F_1 \vee \cdots \vee F_n$ are formulas
- If F and G are formulas, then $F \rightarrow G$ and $F \leftrightarrow G$ are formulas

Quantified Boolean Formulas

Propositional Formula:

- Every Boolean variable is a (propositional) formula
- $\top$ and $\bot$ are formulas
- If F is a formula, then $\neg F$ is a formula
- If $F_1, \ldots, F_n$ are formulas, where $n \geq 2$, then $F_1 \wedge \cdots \wedge F_n$ and $F_1 \vee \cdots \vee F_n$ are formulas
- If F and G are formulas, then $F \rightarrow G$ and $F \leftrightarrow G$ are formulas

Quantified Boolean Formulas (QBFs):

- Every propositional formula is a QBF
- If p is a Boolean variable and F is a QBF, then $\forall p\, F$ and $\exists p\, F$ are QBFs

Quantifiers

- $\forall$ is called the *universal quantifier* (symbol)
- $\exists$ is called the *existential quantifier* (symbol)
- $\forall p\, F$ is read as "for all p, F"
- $\exists p\, F$ is read as "there exists p such that F" or "for some p, F"

For every variable p, we treat $\forall p$ and $\exists p$ as unary operators applied to a formula F

$\forall p$ and $\exists p$ have the highest precedence (like $\neg$), e.g.:

$$\forall p\, p \to q \quad \equiv \quad (\forall p\, p) \to q \quad \not\equiv \quad \forall p\, (p \to q)$$

Note: Some texts give quantifiers lower precedence than all Boolean connectives

Quantifiers

- $\forall$ is called the *universal quantifier* (symbol)
- $\exists$ is called the *existential quantifier* (symbol)
- $\forall p\, F$ is read as "for all p, F"
- $\exists p\, F$ is read as "there exists p such that F" or "for some p, F"

For every variable p, we treat $\forall p$ and $\exists p$ as unary operators applied to a formula F

$\forall p$ and $\exists p$ have the highest precedence (like $\neg$), e.g.:

$$\forall p\, p \to q \quad \equiv \quad (\forall p\, p) \to q \quad \not\equiv \quad \forall p\, (p \to q)$$

Note: Some texts give quantifiers lower precedence than all Boolean connectives

Quantifiers

- $\forall$ is called the *universal quantifier* (symbol)
- $\exists$ is called the *existential quantifier* (symbol)
- $\forall p\, F$ is read as "for all p, F"
- $\exists p\, F$ is read as "there exists p such that F" or "for some p, F"

For every variable p, we treat $\forall p$ and $\exists p$ as unary operators applied to a formula F

$\forall p$ and $\exists p$ have the highest precedence (like $\neg$), e.g.:

$$\forall p\, p \to q \quad \equiv \quad (\forall p\, p) \to q \quad \not\equiv \quad \forall p\, (p \to q)$$

Note: Some texts give quantifiers lower precedence than all Boolean connectives

Quantifiers

- $\forall$ is called the *universal quantifier* (symbol)
- $\exists$ is called the *existential quantifier* (symbol)
- $\forall p\, F$ is read as "for all p, F"
- $\exists p\, F$ is read as "there exists p such that F" or "for some p, F"

For every variable p, we treat $\forall p$ and $\exists p$ as unary operators applied to a formula F

$\forall p$ and $\exists p$ have the highest precedence (like $\neg$), e.g.:

$$\forall p\, p \to q \quad \equiv \quad (\forall p\, p) \to q \quad \not\equiv \quad \forall p\, (p \to q)$$

Note: Some texts give quantifiers lower precedence than all Boolean connectives

Changing interpretations pointwise

Let $\mathcal{I}$ be an interpretation

Notation:

$$\mathcal{I}[p \mapsto b](q) \stackrel{\text{def}}{=} \begin{cases} \mathcal{I}(q), & \text{if } p \neq q \\ b, & \text{if } p = q \end{cases}$$

Example: $\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$

$$\begin{array}{rcl} \mathcal{I}[q \mapsto 1] & = & \{p \mapsto 1, q \mapsto 1, r \mapsto 1\} \\ \mathcal{I}[q \mapsto 0] & = & \{p \mapsto 1, q \mapsto 0, r \mapsto 1\} \quad = \quad \mathcal{I} \\ \mathcal{I}[p \mapsto 0] & = & \{p \mapsto 0, q \mapsto 0, r \mapsto 1\} \end{array}$$

Changing interpretations pointwise

Let $\mathcal{I}$ be an interpretation

Notation:

$$\mathcal{I}[p \mapsto b](q) \stackrel{\text{def}}{=} \begin{cases} \mathcal{I}(q), & \text{if } p \neq q \\ b, & \text{if } p = q \end{cases}$$

Example: $\mathcal{I} = \{ p \mapsto 1, q \mapsto 0, r \mapsto 1 \}$

$$\begin{array}{lcll} \mathcal{I}[q \mapsto 1] & = & \{p \mapsto 1, q \mapsto 1, r \mapsto 1\} & \\ \mathcal{I}[q \mapsto 0] & = & \{p \mapsto 1, q \mapsto 0, r \mapsto 1\} & = \ \mathcal{I} \\ \mathcal{I}[p \mapsto 0] & = & \{p \mapsto 0, q \mapsto 0, r \mapsto 1\} & \end{array}$$

QBF Semantics

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(F_1 \wedge \cdots \wedge F_n) = 1$ iff $\mathcal{I}(F_i) = 1$ for all i
3. $\mathcal{I}(F_1 \vee \cdots \vee F_n) = 1$ iff $\mathcal{I}(F_i) = 1$ for some i
4. $\mathcal{I}(\neg F) = 1$ iff $\mathcal{I}(F) = 0$
5. $\mathcal{I}(F \rightarrow G) = 1$ iff $\mathcal{I}(F) = 0$ or $\mathcal{I}(G) = 1$
6. $\mathcal{I}(F \leftrightarrow G) = 1$ iff $\mathcal{I}(F) = \mathcal{I}(G)$
7. $\mathcal{I}(\forall p\, F) = 1$ iff $\mathcal{I}[p \mapsto 0](F) = 1$ and $\mathcal{I}[p \mapsto 1](F) = 1$
8. $\mathcal{I}(\exists p\, F) = 1$ iff $\mathcal{I}[p \mapsto 0](F) = 1$ or $\mathcal{I}[p \mapsto 1](F) = 1$

QBF Semantics

1. $\mathcal{I}(\top) = 1$ and $\mathcal{I}(\bot) = 0$
2. $\mathcal{I}(F_1 \wedge \cdots \wedge F_n) = 1$ iff $\mathcal{I}(F_i) = 1$ for all i
3. $\mathcal{I}(F_1 \vee \cdots \vee F_n) = 1$ iff $\mathcal{I}(F_i) = 1$ for some i
4. $\mathcal{I}(\neg F) = 1$ iff $\mathcal{I}(F) = 0$
5. $\mathcal{I}(F \rightarrow G) = 1$ iff $\mathcal{I}(F) = 0$ or $\mathcal{I}(G) = 1$
6. $\mathcal{I}(F \leftrightarrow G) = 1$ iff $\mathcal{I}(F) = \mathcal{I}(G)$
7. $\mathcal{I}(\forall p\, F) = 1$ iff $\mathcal{I}[p \mapsto 0](F) = 1$ and $\mathcal{I}[p \mapsto 1](F) = 1$
8. $\mathcal{I}(\exists p\, F) = 1$ iff $\mathcal{I}[p \mapsto 0](F) = 1$ or $\mathcal{I}[p \mapsto 1](F) = 1$

Evaluating a formula: and-or trees

How to evaluate $\forall p \, \exists q \, (p \leftrightarrow q)$ in interpretation $\{ \, p \mapsto 1, q \mapsto 0 \, \}$

Evaluating a formula: and-or trees

How to evaluate $\forall p \, \exists q \, (p \leftrightarrow q)$ in interpretation $\{ p \mapsto 1, q \mapsto 0 \}$

Notation: for brevity, let $\mathcal{I}_{v_1 v_2}$ denote the interpretation $\{ p \mapsto v_1, q \mapsto v_2 \}$

Evaluating a formula: and-or trees

How to evaluate $\forall p \, \exists q \, (p \leftrightarrow q)$ in interpretation $\{ \, p \mapsto 1, q \mapsto 0 \, \}$

Notation: for brevity, let $\mathcal{I}_{v_1 v_2}$ denote the interpretation $\{ \, p \mapsto v_1, q \mapsto v_2 \, \}$

$$\mathcal{I}_{10} \models \forall p \, \exists q \, (p \leftrightarrow q)$$

Evaluating a formula: and-or trees

How to evaluate $\forall p\, \exists q\, (p \leftrightarrow q)$ in interpretation $\{\, p \mapsto 1, q \mapsto 0 \,\}$

Notation: for brevity, let $\mathcal{I}_{v_1 v_2}$ denote the interpretation $\{\, p \mapsto v_1, q \mapsto v_2 \,\}$

$$\mathcal{I}_{10} \models \forall p\, \exists q\, (p \leftrightarrow q) \quad \Leftrightarrow \quad \boxed{\begin{array}{ll} \mathcal{I}_{00} \models \exists q\, (p \leftrightarrow q) & \\ & \text{and} \\ \mathcal{I}_{10} \models \exists q\, (p \leftrightarrow q) & \end{array}}$$

Evaluating a formula: and-or trees

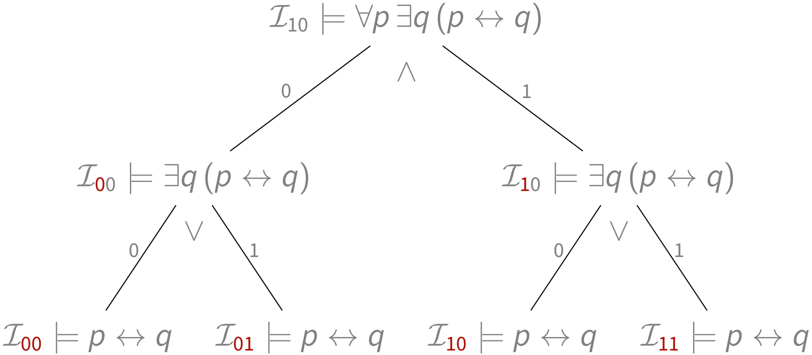
How to evaluate $\forall p\, \exists q\, (p \leftrightarrow q)$ in interpretation $\{\, p \mapsto 1, q \mapsto 0 \,\}$

Notation: for brevity, let $\mathcal{I}_{v_1 v_2}$ denote the interpretation $\{\, p \mapsto v_1, q \mapsto v_2 \,\}$

$$\mathcal{I}_{10} \models \forall p\, \exists q\, (p \leftrightarrow q) \quad \Leftrightarrow \quad \boxed{\begin{array}{ll} \mathcal{I}_{00} \models \exists q\, (p \leftrightarrow q) & \\ & \text{and} \\ \mathcal{I}_{10} \models \exists q\, (p \leftrightarrow q) & \end{array}}$$

$$\Leftrightarrow \quad \boxed{\begin{array}{ll} \boxed{\begin{array}{ll} \mathcal{I}_{00} \models p \leftrightarrow q & \\ & \text{or} \\ \mathcal{I}_{01} \models p \leftrightarrow q & \end{array}} & \\ & \text{and} \\ \boxed{\begin{array}{ll} \mathcal{I}_{10} \models p \leftrightarrow q & \\ & \text{or} \\ \mathcal{I}_{11} \models p \leftrightarrow q & \end{array}} & \end{array}}$$

Evaluating a formula: and-or trees



Evaluating a formula

Notation: Denote any interpretation $\{ p \mapsto b_1, q \mapsto b_2 \}$ by $\mathcal{I}_{b_1 b_2}$
Use wildcards $*$ to denote *any* Boolean value

$$\mathcal{I}_{**} \models \forall p \, \exists q \, (p \leftrightarrow q)$$

Evaluating a formula

Notation: Denote any interpretation $\{\, p \mapsto b_1, q \mapsto b_2 \,\}$ by $\mathcal{I}_{b_1 b_2}$
Use wildcards $*$ to denote *any* Boolean value

$$\mathcal{I}_{**} \models \forall p\, \exists q\, (p \leftrightarrow q) \quad \Leftrightarrow \quad \boxed{\begin{array}{ll} \mathcal{I}_{0*} \models \exists q\, (p \leftrightarrow q) & \\ & \text{and} \\ \mathcal{I}_{1*} \models \exists q\, (p \leftrightarrow q) & \end{array}}$$

Evaluating a formula

Notation: Denote any interpretation $\{\, p \mapsto b_1, q \mapsto b_2 \,\}$ by $\mathcal{I}_{b_1 b_2}$
Use wildcards $*$ to denote *any* Boolean value

$$\mathcal{I}_{**} \models \forall p\, \exists q\, (p \leftrightarrow q) \quad \Leftrightarrow \quad \boxed{\begin{array}{l} \mathcal{I}_{0*} \models \exists q\, (p \leftrightarrow q) \\ \mathcal{I}_{1*} \models \exists q\, (p \leftrightarrow q) \end{array} \quad \text{and}}$$

$$\Leftrightarrow \quad \boxed{\begin{array}{l} \boxed{\begin{array}{l} \mathcal{I}_{00} \models p \leftrightarrow q \\ \mathcal{I}_{01} \models p \leftrightarrow q \end{array} \quad \text{or}} \\ \\ \boxed{\begin{array}{l} \mathcal{I}_{10} \models p \leftrightarrow q \\ \mathcal{I}_{11} \models p \leftrightarrow q \end{array} \quad \text{or}} \end{array} \quad \textit{and}}$$

Evaluating a formula

Notation: Denote any interpretation $\{\, p \mapsto b_1, q \mapsto b_2 \,\}$ by $\mathcal{I}_{b_1 b_2}$
Use wildcards $*$ to denote *any* Boolean value

$$\mathcal{I}_{**} \models \forall p\, \exists q\, (p \leftrightarrow q) \quad \Leftrightarrow \quad \boxed{\begin{array}{ll} \mathcal{I}_{0*} \models \exists q\, (p \leftrightarrow q) & \\ & \text{and} \\ \mathcal{I}_{1*} \models \exists q\, (p \leftrightarrow q) & \end{array}}$$

$$\Leftrightarrow \quad \boxed{\begin{array}{ll} \boxed{\begin{array}{ll} \mathcal{I}_{00} \models p \leftrightarrow q & \\ & \text{or} \\ \mathcal{I}_{01} \models p \leftrightarrow q & \end{array}} & \\ & \textit{and} \\ \boxed{\begin{array}{ll} \mathcal{I}_{10} \models p \leftrightarrow q & \\ & \text{or} \\ \mathcal{I}_{11} \models p \leftrightarrow q & \end{array}} & \end{array}}$$

The variables p and q are *bound* by the quantifiers $\forall p$ and $\exists q$, so
the value of the formula does not depend on the values p and q

Subformula

Propositional formulas:

- F is the immediate subformula of $\neg F$
- $F_1, \ldots, F_n$ are the immediate subformulas of $F_1 \wedge \cdots \wedge F_n$
- $F_1, \ldots, F_n$ are the immediate subformulas of $F_1 \vee \cdots \vee F_n$
- F_1 and F_2 are the immediate subformulas of $F_1 \rightarrow F_2$
- F_1 and F_2 are the immediate subformulas of $F_1 \leftrightarrow F_2$
- …

Subformula

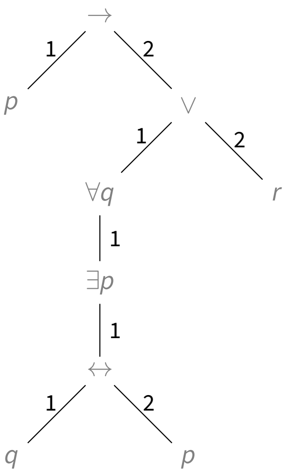
Propositional formulas:

- F is the immediate subformula of $\neg F$
- $F_1, \ldots, F_n$ are the immediate subformulas of $F_1 \wedge \cdots \wedge F_n$
- $F_1, \ldots, F_n$ are the immediate subformulas of $F_1 \vee \cdots \vee F_n$
- F_1 and F_2 are the immediate subformulas of $F_1 \rightarrow F_2$
- F_1 and F_2 are the immediate subformulas of $F_1 \leftrightarrow F_2$
- ...

Quantified Boolean formulas:

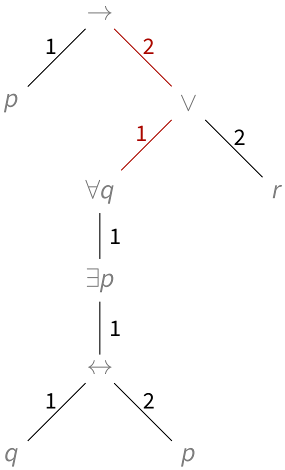
- F is the immediate subformula of $\forall p\, F$ and of $\exists p\, F$

Positions and polarity by example



$p \to \forall q \, \exists p (q \leftrightarrow p) \vee r$

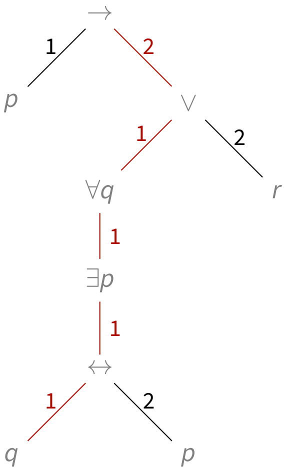
Positions and polarity by example



$p \rightarrow \forall q\, \exists p(q \leftrightarrow p) \vee r$

$F|_{2.1} = \forall q\, \exists p(q \leftrightarrow p)$

Positions and polarity by example



$p \rightarrow \forall q\, \exists p(q \leftrightarrow p) \vee r$

$F|_{2.1} = \forall q\, \exists p(q \leftrightarrow p)$

$F|_{2.1.1.1.1} = q$

Positions and Polarity

Let $F|_\pi = A$

Propositional formulas:

- If A has the form $\neg A_1$,
  then $\pi.1$ is a position in F, $F|_{\pi.1} \stackrel{\text{def}}{=} A_1$ and $pol(F, \pi.1) \stackrel{\text{def}}{=} -pol(F, \pi)$
- If A has the form $A_1 \land \cdots \land A_n$ or $A_1 \lor \cdots \lor A_n$ and $i \in \{ 1, \ldots, n \}$,
  then $\pi.i$ is a position in F and $pol(F, \pi.i) \stackrel{\text{def}}{=} pol(F, \pi)$
- …

Positions and Polarity

Let $F|_\pi = A$

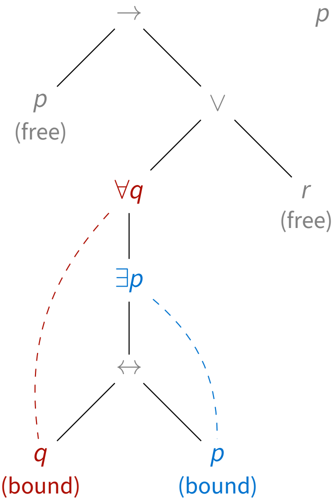
Propositional formulas:

- If A has the form $\neg A_1$,
  then $\pi.1$ is a position in F, $F|_{\pi.1} \stackrel{\text{def}}{=} A_1$ and $pol(F, \pi.1) \stackrel{\text{def}}{=} -pol(F, \pi)$
- If A has the form $A_1 \wedge \cdots \wedge A_n$ or $A_1 \vee \cdots \vee A_n$ and $i \in \{ 1, \ldots, n \}$,
  then $\pi.i$ is a position in F and $pol(F, \pi.i) \stackrel{\text{def}}{=} pol(F, \pi)$
- ...

Quantified Boolean formulas:

- If A has the form $\forall p\, B$ or $\exists p\, B$,
  then $\pi.1$ is a position in F, $F|_{\pi.1} \stackrel{\text{def}}{=} B$ and $pol(F, \pi.1) \stackrel{\text{def}}{=} pol(F, \pi)$

Free and bound variables by example



$p \to \forall q \, \exists p \, (q \leftrightarrow p) \lor r$

Free and bound occurrences in programs

- Free variables in formulas are analogous to global variables in programs
- Bound variables in formulas are analogous to local variables in programs

```
int offset_sym_diff(int i, int j)
{
  int k = i > j ?  i - j :  j - i;
  return a + k
}


sum = i + offset_sym_diff(3,4);
```

Free and bound occurrences in programs

- Free variables in formulas are analogous to global variables in programs
- Bound variables in formulas are analogous to local variables in programs

binding

```
int offset_sym_diff(int i, int j)
{
  int k = i > j ?  i - j :  j - i;
  return a + k
}
```

bound

```
sum = i + offset_sym_diff(3,4);
```

free

Free and bound occurrences of variables

Let F be a QBF and p be atom of at position π

The occurrence of p at position π in F is *bound* if π can be represented as a concatenation of two strings $\pi_1\pi_2$ such that $F|_{\pi_1}$ has the form $\forall p\, G$ or $\exists p\, G$

Free and bound occurrences of variables

Let F be a QBF and p be atom of at position π

The occurrence of p at position π in F is *bound* if π can be represented as a concatenation of two strings $\pi_1\pi_2$ such that $F|_{\pi_1}$ has the form $\forall p\, G$ or $\exists p\, G$

A bound occurrence of p is an occurrence *in the scope of* $\forall p$ *or* $\exists p$

Free and bound occurrences of variables

Let F be a QBF and p be atom of at position π

The occurrence of p at position π in F is *bound* if π can be represented as a concatenation of two strings $\pi_1\pi_2$ such that $F|_{\pi_1}$ has the form $\forall p\, G$ or $\exists p\, G$

A bound occurrence of p is an occurrence *in the scope of* $\forall p$ *or* $\exists p$

Free occurrence: not bound

Free and bound occurrences of variables

Let F be a QBF and p be atom of at position π

The occurrence of p at position π in F is *bound* if π can be represented as a concatenation of two strings $\pi_1\pi_2$ such that $F|_{\pi_1}$ has the form $\forall p\, G$ or $\exists p\, G$

A bound occurrence of p is an occurrence *in the scope of* $\forall p$ *or* $\exists p$

Free occurrence: not bound

Free (bound) variable of a formula: a variable with at least one free (bound) occurrence

Free and bound occurrences of variables

Let F be a QBF and p be atom of at position π

The occurrence of p at position π in F is *bound* if π can be represented as a concatenation of two strings $\pi_1\pi_2$ such that $F|_{\pi_1}$ has the form $\forall p\, G$ or $\exists p\, G$

A bound occurrence of p is an occurrence *in the scope of* $\forall p$ *or* $\exists p$

Free occurrence: not bound

Free (bound) variable of a formula: a variable with at least one free (bound) occurrence

Closed formula: formula with no free variables

Only free variables matter for truth

The truth value of a QBF formula F depends only on the values of its free variables:

Lemma 1

Suppose $\mathcal{I}_1(p) = \mathcal{I}_2(p)$ *for all free variables* p *of* F. *Then*

$$\mathcal{I}_1 \models F \textit{ iff } \mathcal{I}_2 \models F$$

Only free variables matter for truth

The truth value of a QBF formula F depends only on the values of its free variables:

Lemma 1
Suppose $\mathcal{I}_1(p) = \mathcal{I}_2(p)$ for all free variables p of F. Then

$$\mathcal{I}_1 \models F \text{ iff } \mathcal{I}_2 \models F$$

Theorem 2
Let F be a closed formula and let $\mathcal{I}_1, \mathcal{I}_2$ be two interpretations. Then

$$\mathcal{I}_1 \models F \text{ iff } \mathcal{I}_2 \models F$$

Truth, Validity and Satisfiability

Validity and satisfiability are defined as for propositional formulas

Truth, Validity and Satisfiability

Validity and satisfiability are defined as for propositional formulas

There is no difference between these two notions for closed formulas:

Lemma 3

For every interpretation $\mathcal{I}$ and closed formula F the following statements are equivalent: (i) $\mathcal{I} \models F$; (ii) F is satisfiable; and (iii) F is valid.

Truth, Validity and Satisfiability

Validity and satisfiability are defined as for propositional formulas

There is no difference between these two notions for closed formulas:

Lemma 3

For every interpretation $\mathcal{I}$ and closed formula F the following statements are equivalent: (i) $\mathcal{I} \models F$; (ii) F is satisfiable; and (iii) F is valid.

Satisfiability can be expressed through satisfiability/validity of closed formulas:

Lemma 4

Let F be a formula with free variables $p_1, \ldots, p_n$.

- *F is satisfiable iff $\exists p_1 \cdots \exists p_n\, F$ is satisfiable/valid*
- *F is valid iff the formula $\forall p_1 \cdots \forall p_n\, F$ is satisfiable/valid*

Substitutions for propositional formulas

Substitution: F_p^G: denotes the formula obtained from F by replacing all occurrences of variable p by G

Example:

$$((p \vee s) \wedge (q \rightarrow p))_p^{(l \wedge s)} = ((l \wedge s) \vee s) \wedge (q \rightarrow (l \wedge s))$$

Property: Applying any substitution to a valid formula results in a valid formula

Substitutions for propositional formulas

Substitution: F_p^G: denotes the formula obtained from F by replacing all occurrences of variable p by G

Example:

$$((p \vee s) \wedge (q \rightarrow p))_p^{(l \wedge s)} \;=\; ((l \wedge s) \vee s) \wedge (q \rightarrow (l \wedge s))$$

Property: Applying any substitution to a valid formula results in a valid formula

Substitutions for propositional formulas

Substitution: F^G_p: denotes the formula obtained from F by replacing all occurrences of variable p by G

Example:

$$((p \vee s) \wedge (q \rightarrow p))^{(l \wedge s)}_p \; = \; (((l \wedge s) \vee s) \wedge (q \rightarrow (l \wedge s))$$

Property: Applying any substitution to a valid formula results in a valid formula

Substitutions for **quantified** formulas

Some problems …

Substitutions for quantified formulas

Some problems …

Consider $\exists q\, (\neg p \leftrightarrow q)$

Substitutions for quantified formulas

Some problems . . .

Consider $\exists q\,(\neg p \leftrightarrow q)$

We cannot simply replace variables by formulas any more:
$\exists(r \rightarrow r)\,(\neg p \leftrightarrow r \rightarrow r)$??? Ill formed

Substitutions for quantified formulas

Some problems ...

Consider $\exists q\,(\neg p \leftrightarrow q)$

We cannot simply replace variables by formulas any more:
$\exists(r \to r)\,(\neg p \leftrightarrow r \to r)$??? Ill formed

Free variables are parameters: we can only substitute for parameters.
But a variable can have both free and bound occurrences in a formula, e.g.,

$$\forall p\,((p \to q) \lor \neg p) \land (q \lor (q \to p))$$

Renaming bound variables

Notation: $\exists\!\!\forall$: any of $\exists$, $\forall$

Renaming bound variables

Notation: $\exists\!\!\!\forall$: any of $\exists$, $\forall$

Renaming bound variables in $F[\ \exists\!\!\!\forall pG\ ]$:

1. Take a *fresh* variable q (i.e., a variable not occurring in F)
2. Replace all free occurrences of p in G (not in F!) by q, obtaining G'
3. Consider $F[\ \exists\!\!\!\forall qG'\ ]$

Renaming bound variables

Notation: $\exists\!\!\!\forall$: any of $\exists$, $\forall$

Renaming bound variables in $F[\, \exists\!\!\!\forall p G \,]$:

1. Take a *fresh* variable q (i.e., a variable not occurring in F)
2. Replace all free occurrences of p in G (not in F!) by q, obtaining G'
3. Consider $F[\, \exists\!\!\!\forall q G' \,]$

Example:

$\exists r\, (\forall p\, ((p \to r) \wedge p)\,) \vee p$ rename p to q, obtaining

$\exists r\, (\forall q\, ((q \to r) \wedge q)\,) \vee p$

Renaming bound variables

Notation: $\exists\!\!\!\forall$: any of $\exists$, $\forall$

Renaming bound variables in $F[\,\exists\!\!\!\forall p G\,]$:

1. Take a *fresh* variable q (i.e., a variable not occurring in F)
2. Replace all free occurrences of p in G (not in F!) by q, obtaining G'
3. Consider $F[\,\exists\!\!\!\forall q G'\,]$

Example:

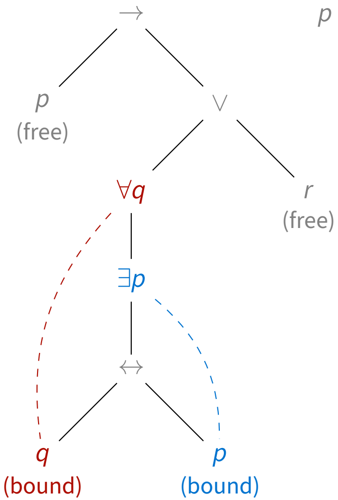
$\exists r\,(\forall p\,((p \to r) \land p))\lor p$ rename p to q, obtaining

$\exists r\,(\forall q\,((q \to r) \land q))\lor p$

Lemma 5

$F[\,\exists\!\!\!\forall p G\,] \equiv F[\,\exists\!\!\!\forall q G'\,]$

Free and bound variables by example



$p \rightarrow \forall q \exists p (q \leftrightarrow p) \vee r$

Rectified formulas

Rectified formula F:

1. no variable appears both free and bound in F
2. for every variable p, there is at most one occurrence of quantifier $\exists\!\!\forall p$ in F

Any formula can be rectified by renaming its bound variables

We can use the usual notation F_p^G for substitutions into a rectified formula F, assuming p occurs only free in F

Rectified formulas

Rectified formula F:

1. no variable appears both free and bound in F
2. for every variable p, there is at most one occurrence of quantifier $\exists\!\!\!\forall p$ in F

Any formula can be rectified by renaming its bound variables

We can use the usual notation F_p^G for substitutions into a rectified formula F, assuming p occurs only free in F

Rectified formulas

Rectified formula F:

1. no variable appears both free and bound in F
2. for every variable p, there is at most one occurrence of quantifier $\exists\!\!\!\forall p$ in F

Any formula can be rectified by renaming its bound variables

We can use the usual notation F_p^G for substitutions into a rectified formula F, assuming p occurs only free in F

Rectification: Example

$$p \to \exists p\,(p \land \forall p\,(p \lor r \to \neg p)) \Rightarrow$$

$$p \to \exists p\,(p \land \forall p_1\,(p_1 \lor r \to \neg p_1)) \Rightarrow$$

$$p \to \exists p_2\,(p_2 \land \forall p_1\,(p_1 \lor r \to \neg p_1))$$

Rectification: Example

$$p \rightarrow \exists p \, (p \wedge \forall p \, (p \vee r \rightarrow \neg p)) \Rightarrow$$

$$p \rightarrow \exists p \, (p \wedge \forall p_1 \, (p_1 \vee r \rightarrow \neg p_1)) \Rightarrow$$

$$p \rightarrow \exists p_2 \, (p_2 \wedge \forall p_1 \, (p_1 \vee r \rightarrow \neg p_1))$$

Rectification: Example

$$p \to \exists p\,(p \wedge \forall p\,(p \vee r \to \neg p)) \Rightarrow$$

$$p \to \exists p\,(p \wedge \forall p_1\,(p_1 \vee r \to \neg p_1)) \Rightarrow$$

$$p \to \exists p_2\,(p_2 \wedge \forall p_1\,(p_1 \vee r \to \neg p_1))$$

Rectification: Example

$$p \to \exists p\,(p \land \boxed{\forall p\,(p \lor r \to \neg p)}\,) \Rightarrow$$

$$p \to \boxed{\exists p\,(p \land \forall p_1\,(p_1 \lor r \to \neg p_1))} \Rightarrow$$

$$p \to \exists p_2\,(p_2 \land \forall p_1\,(p_1 \lor r \to \neg p_1))$$

Rectification: Example

$$p \to \exists p\,(p \land \forall p\,(p \lor r \to \neg p)) \Rightarrow$$

$$p \to \exists p\,(p \land \forall p_1\,(p_1 \lor r \to \neg p_1)) \Rightarrow$$

$$p \to \exists p_2\,(p_2 \land \forall p_1\,(p_1 \lor r \to \neg p_1))$$

Renaming each bound variable to a fresh one preserves equivalence

Another problem

$\exists q\, (\neg p \leftrightarrow q)$ This formula is valid (whatever value p has, choose the opposite for q)

substitute p by q

$\exists q\, (\neg q \leftrightarrow q)$ This formula is unsatisfiable!

Substitutions below a quantifier should not lead to *variable capturing*

Another problem

$\exists q\,(\neg p \leftrightarrow q)$ This formula is valid (whatever value p has, choose the opposite for q)

substitute p by q

$\exists q\,(\neg q \leftrightarrow q)$ This formula is unsatisfiable!

Substitutions below a quantifier should not lead to *variable capturing*

Another problem

$\exists q\,(\neg p \leftrightarrow q)$ This formula is valid (whatever value p has, choose the opposite for q)

substitute p by q

$\exists q\,(\neg q \leftrightarrow q)$ This formula is unsatisfiable!

Substitutions below a quantifier should not lead to *variable capturing*

Another problem

$\exists q\,(\neg p \leftrightarrow q)$ This formula is valid (whatever value p has, choose the opposite for q)

substitute p by q

$\exists q\,(\neg q \leftrightarrow q)$ This formula is unsatisfiable!

Substitutions below a quantifier should not lead to *variable capturing*

Another restriction

Suppose we want to substitute G for p in $F[p]$

Requirement: no free variables in G become bound in F_p^G

(In previous example, $(\exists q\,(\neg p \leftrightarrow q))_p^q$ does not satisfy this requirement)

Uniform solution: renaming of bound variables before application of substitution

Example:

Since $\exists q\,(\neg p \leftrightarrow q) \equiv \exists r(\neg p \leftrightarrow r)$

we can use $(\exists r\,(\neg p \leftrightarrow r))_p^q$ instead of $(\exists q\,(\neg p \leftrightarrow q))_p^q$

Another restriction

Suppose we want to substitute G for p in $F[p]$

Requirement: no free variables in G become bound in F_p^G

(In previous example, $(\exists q\, (\neg p \leftrightarrow q))_p^q$ does not satisfy this requirement)

Uniform solution: renaming of bound variables before application of substitution

Example:

Since $\exists q\, (\neg p \leftrightarrow q) \equiv \exists r (\neg p \leftrightarrow r)$

we can use $(\exists r\, (\neg p \leftrightarrow r))_p^q$ instead of $(\exists q\, (\neg p \leftrightarrow q))_p^q$

Another restriction

Suppose we want to substitute G for p in $F[p]$

Requirement: no free variables in G become bound in F_p^G

(In previous example, $(\exists q\,(\neg p \leftrightarrow q))_p^q$ does not satisfy this requirement)

Uniform solution: renaming of bound variables before application of substitution

Example:

Since $\exists q\,(\neg p \leftrightarrow q) \equiv \exists\, r(\neg p \leftrightarrow r)$
we can use $(\exists r\,(\neg p \leftrightarrow r))_p^q$ instead of $(\exists q\,(\neg p \leftrightarrow q))_p^q$

Another restriction

Suppose we want to substitute G for p in $F[p]$

Requirement: no free variables in G become bound in F_p^G

(In previous example, $(\exists q\,(\neg p \leftrightarrow q))_p^q$ does not satisfy this requirement)

Unifor tution

Examp

Since $\exists$

we can

From now on, we always assume that:

1. formulas are rectified
2. all substitutions satisfy the requirement above

Equivalent replacement

Lemma 6
Let $\mathcal{I}$ be an interpretation and $\mathcal{I} \models F_1 \leftrightarrow F_2$. Then $\mathcal{I} \models G[F_1] \leftrightarrow G[F_2]$.

Theorem 7 (Equivalent Replacement)
Let $F_1 \equiv F_2$. Then $G[F_1] \equiv G[F_2]$.

Equivalent replacement

Lemma 6

Let $\mathcal{I}$ be an interpretation and $\mathcal{I} \models F_1 \leftrightarrow F_2$. Then $\mathcal{I} \models G[F_1] \leftrightarrow G[F_2]$.

Theorem 7 (Equivalent Replacement)

Let $F_1 \equiv F_2$. Then $G[F_1] \equiv G[F_2]$.

More equivalences

Theorem 8

The following holds for all QBFs F:

1. $\forall p_1 \forall p_2\, F \;\equiv\; \forall p_2 \forall p_1\, F$
2. $\exists p_1 \exists p_2\, F \;\equiv\; \exists p_2 \exists p_1\, F$
3. $\exists\!\!\!\forall p\, F \;\equiv\; F$ *if p does not occur free in F*
4. $\forall p\, F \;\equiv\; F_p^{\perp} \wedge F_p^{\top}$
5. $\exists p\, F \;\equiv\; F_p^{\perp} \vee F_p^{\top}$

Note: In general, $\exists p_1 \forall p_2 F \;\not\equiv\; \forall p_2 \exists p_1 F$!

Example:

- $\forall p \exists q\, (p \leftrightarrow q) \;\equiv\; \top$

More equivalences

Theorem 8

The following holds for all QBFs F:

1. $\forall p_1 \forall p_2\, F \;\equiv\; \forall p_2 \forall p_1\, F$
2. $\exists p_1 \exists p_2\, F \;\equiv\; \exists p_2 \exists p_1\, F$
3. $\exists\!\!\!\forall p\, F \;\equiv\; F$ *if* p *does not occur free in* F
4. $\forall p\, F \;\equiv\; F^{\bot}_{p} \wedge F^{\top}_{p}$
5. $\exists p\, F \;\equiv\; F^{\bot}_{p} \vee F^{\top}_{p}$

Note: In general, $\exists p_1 \forall p_2 F \;\not\equiv\; \forall p_2 \exists p_1 F$!

Example:

- $\forall p\, \exists q\, (p \leftrightarrow q) \;\equiv\; \top$

More equivalences

Theorem 8

The following holds for all QBFs F:

1. $\forall p_1 \forall p_2\, F \equiv \forall p_2 \forall p_1\, F$
2. $\exists p_1 \exists p_2\, F \equiv \exists p_2 \exists p_1\, F$
3. $\exists\!\!\forall p\, F \equiv F$ *if p does not occur free in F*
4. $\forall p\, F \equiv F_p^{\perp} \wedge F_p^{\top}$
5. $\exists p\, F \equiv F_p^{\perp} \vee F_p^{\top}$

Note: In general, $\exists p_1 \forall p_2 F \not\equiv \forall p_2 \exists p_1 F$!

Example:

- $\forall p\, \exists q\, (p \leftrightarrow q) \equiv \top$
- $\exists q\, \forall p\, (p \leftrightarrow q) \equiv \perp$

More equivalences

Theorem 8

The following holds for all QBFs F:

1. $\forall p_1 \forall p_2\, F \;\equiv\; \forall p_2 \forall p_1\, F$
2. $\exists p_1 \exists p_2\, F \;\equiv\; \exists p_2 \exists p_1\, F$
3. $\exists\!\!\forall p\, F \;\equiv\; F$ *if p does not occur free in F*
4. $\forall p\, F \;\equiv\; F_p^{\perp} \wedge F_p^{\top}$
5. $\exists p\, F \;\equiv\; F_p^{\perp} \vee F_p^{\top}$

Note: In general, $\exists p_1 \forall p_2 F \;\not\equiv\; \forall p_2 \exists p_1 F$!

Example:

- $\forall p\, \exists q\, (p \leftrightarrow q) \;\equiv\; \top$
- $\exists q\, \forall p\, (p \leftrightarrow q) \;\equiv\; \perp$

Prenex form

Quantifier-free formula: no quantifiers (that is, propositional)

Prenex formula: formula of the form

$$\overbrace{\exists\forall_1 p_1 \cdots \exists\forall_n p_n}^{\text{quantifier prefix}} \underbrace{G}_{\text{matrix}}$$

with G quantifier-free

Outermost prefix of $\exists\forall_1 p_1 \cdots \exists\forall_n p_n G$: the longest subsequence $\exists\forall_1 p_1 \cdots \exists\forall_k p_k$ of $\exists\forall_1 p_1 \cdots \exists\forall_n p_n$ such that $\exists\forall_1 = \cdots = \exists\forall_k$

Prenex form

Quantifier-free formula: no quantifiers (that is, propositional)

Prenex formula: formula of the form

$$\overbrace{\exists\forall_1 p_1 \cdots \exists\forall_n p_n}^{\textit{quantifier prefix}} \; \underbrace{G}_{\textit{matrix}}$$

with G quantifier-free

Outermost prefix of $\exists\forall_1 p_1 \cdots \exists\forall_n p_n G$: the longest subsequence $\exists\forall_1 p_1 \cdots \exists\forall_k p_k$ of $\exists\forall_1 p_1 \cdots \exists\forall_n p_n$ such that $\exists\forall_1 = \cdots = \exists\forall_k$

Prenex form

Quantifier-free formula: no quantifiers (that is, propositional)

Prenex formula: formula of the form

$$\overbrace{\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n}^{\textit{quantifier prefix}} \; \underbrace{G}_{\textit{matrix}}$$

with G quantifier-free

Outermost prefix of $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n G$: the longest subsequence $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_k p_k$ of $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n$ such that $\exists\!\!\!\forall_1 = \cdots = \exists\!\!\!\forall_k$

Prenex form

Quantifier-free formula: no quantifiers (that is, propositional)

Prenex formula: formula of the form

$$\overbrace{\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n}^{\text{quantifier prefix}} \underbrace{G}_{\text{matrix}}$$

with G quantifier-free

Outermost prefix of $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n G$: the longest subsequence $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_k p_k$ of $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n$ such that $\exists\!\!\!\forall_1 = \cdots = \exists\!\!\!\forall_k$

Example

- outermost prefix of $\forall p\, \forall q\, \exists r (r \land p \to q)$: $\forall p\, \forall q$
- outermost prefix of $\exists p\, \forall q\, \exists r (r \land p \to q)$: $\exists p$

Prenex form

Quantifier-free formula: no quantifiers (that is, propositional)

Prenex formula: formula of the form

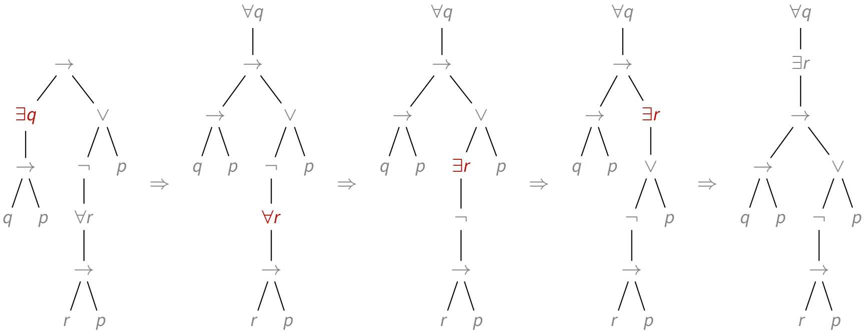
$$\overbrace{\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n}^{\text{quantifier prefix}} \underbrace{G}_{\text{matrix}}$$

with G quantifier-free

Outermost prefix of $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n G$: the longest subsequence $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_k p_k$ of $\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n$ such that $\exists\!\!\!\forall_1 = \cdots = \exists\!\!\!\forall_k$

A formula F is a *prenex form* of a formula G if F is prenex and $F \equiv G$

Conversion to prenex form, Example I



Conversion to prenex form, Example I

Same conversion:

$$
\begin{aligned}
&(\exists q\,(q \rightarrow p)) \rightarrow \neg\forall r\,(r \rightarrow p) \vee p && \Rightarrow\\
&\forall q\,((q \rightarrow p) \rightarrow \neg\forall r\,(r \rightarrow p) \vee p) && \Rightarrow\\
&\forall q\,((q \rightarrow p) \rightarrow \exists r\,\neg(r \rightarrow p) \vee p) && \Rightarrow\\
&\forall q\,((q \rightarrow p) \rightarrow \exists r\,(\neg(r \rightarrow p) \vee p)) && \Rightarrow\\
&\forall q\,\exists r\,((q \rightarrow p) \rightarrow \neg(r \rightarrow p) \vee p)
\end{aligned}
$$

Prenexing rules

$$
\begin{array}{c}
(\exists\!\!\!\forall p\, F_1) \wedge \cdots \wedge F_n \;\Rightarrow\; \exists\!\!\!\forall p\, (F_1 \wedge \cdots \wedge F_n) \\
(\exists\!\!\!\forall p\, F_1) \vee \cdots \vee F_n \;\Rightarrow\; \exists\!\!\!\forall p\, (F_1 \vee \cdots \vee F_n)
\end{array}
$$

$$
\begin{array}{cc}
(\forall p\, F_1) \rightarrow F_2 \;\Rightarrow\; \exists p\, (F_1 \rightarrow F_2) & F_1 \rightarrow (\exists p\, F_2) \;\Rightarrow\; \exists p\, (F_1 \rightarrow F_2) \\
(\exists p\, F_1) \rightarrow F_2 \;\Rightarrow\; \forall p\, (F_1 \rightarrow F_2) & F_1 \rightarrow (\forall p\, F_2) \;\Rightarrow\; \forall p\, (F_1 \rightarrow F_2) \\
\neg \forall p\, F \;\Rightarrow\; \exists p\, \neg F & \neg \exists p\, F \;\Rightarrow\; \forall p\, \neg F
\end{array}
$$

Conversion to prenex form, Example II

$$\exists q\,(q \to p) \to \neg\forall r\,(r \to p) \lor p \qquad \Rightarrow$$
$$\exists q\,(q \to p) \to \exists r\,\neg(r \to p) \lor p \qquad \Rightarrow$$
$$\exists q\,(q \to p) \to \exists r\,(\neg(r \to p) \lor p) \qquad \Rightarrow$$
$$\exists r\,(\exists q\,(q \to p) \to \neg(r \to p) \lor p) \qquad \Rightarrow$$
$$\exists r\,\forall q\,((q \to p) \to \neg(r \to p) \lor p)$$

Checking the satisfiability of QBFs

The algorithms for propositional satisfiability or validity can be adapted to QBF

We will see:

- Splitting
- DPLL

Recall:

1. $F(p_1, \ldots, p_n)$ is satisfiable iff $\exists p_1 \cdots \exists p_n\, F(p_1, \ldots, p_n)$ is satisfiable
2. $F(p_1, \ldots, p_n)$ is valid iff $\forall p_1 \cdots \forall p_n\, F(p_1, \ldots, p_n)$ is satisfiable
3. A closed QBF is either always true (valid) or always false (unsatisfiable)

The algorithms will check whether a closed formula is valid or unsatisfiable

Checking the satisfiability of QBFs

The algorithms for propositional satisfiability or validity can be adapted to QBF

We will see:

- Splitting
- DPLL

Recall:

1. $F(p_1, \ldots, p_n)$ is satisfiable iff $\exists p_1 \cdots \exists p_n \, F(p_1, \ldots, p_n)$ is satisfiable
2. $F(p_1, \ldots, p_n)$ is valid iff $\forall p_1 \cdots \forall p_n \, F(p_1, \ldots, p_n)$ is satisfiable
3. A closed QBF is either always true (valid) or always false (unsatisfiable)

The algorithms will check whether a closed formula is valid or unsatisfiable

Checking the satisfiability of QBFs

The algorithms for propositional satisfiability or validity can be adapted to QBF

We will see:

- Splitting
- DPLL

Recall:

1. $F(p_1, \ldots, p_n)$ is satisfiable iff $\exists p_1 \cdots \exists p_n\, F(p_1, \ldots, p_n)$ is satisfiable
2. $F(p_1, \ldots, p_n)$ is valid iff $\forall p_1 \cdots \forall p_n\, F(p_1, \ldots, p_n)$ is satisfiable
3. A closed QBF is either always true (valid) or always false (unsatisfiable)

The algorithms will check whether a closed formula is valid or unsatisfiable

Splitting: foundations

Lemma 9

- *A closed formula $\forall p\, F$ evaluates to 1 iff both $F_p^{\perp}$ and $F_p^{\top}$ evaluate to 1.*
- *A closed formula $\exists p\, F$ evaluates to true iff either $F_p^{\perp}$ or $F_p^{\top}$ evaluates to 1.*

Splitting

Simplification rules for $\top$:

$$\neg\top \;\Rightarrow\; \bot$$
$$\top \wedge F_1 \wedge \cdots \wedge F_n \;\Rightarrow\; F_1 \wedge \cdots \wedge F_n$$
$$\top \vee F_1 \vee \cdots \vee F_n \;\Rightarrow\; \top$$
$$F \rightarrow \top \;\Rightarrow\; \top \qquad \top \rightarrow F \;\Rightarrow\; F$$
$$F \leftrightarrow \top \;\Rightarrow\; F \qquad \top \leftrightarrow F \;\Rightarrow\; F$$

Simplification rules for $\bot$:

$$\neg\bot \;\Rightarrow\; \top$$
$$\bot \wedge F_1 \wedge \cdots \wedge F_n \;\Rightarrow\; \bot$$
$$\bot \vee F_1 \vee \cdots \vee F_n \;\Rightarrow\; F_1 \vee \cdots \vee F_n$$
$$F \rightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \rightarrow F \;\Rightarrow\; \top$$
$$F \leftrightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \leftrightarrow F \;\Rightarrow\; \neg F$$

Splitting

Simplification rules for $\top$:

$$
\begin{gathered}
\neg\top \;\Rightarrow\; \bot \\
\top \wedge F_1 \wedge \cdots \wedge F_n \;\Rightarrow\; F_1 \wedge \cdots \wedge F_n \\
\top \vee F_1 \vee \cdots \vee F_n \;\Rightarrow\; \top \\
F \rightarrow \top \;\Rightarrow\; \top \qquad \top \rightarrow F \;\Rightarrow\; F \\
F \leftrightarrow \top \;\Rightarrow\; F \qquad \top \leftrightarrow F \;\Rightarrow\; F \\
\forall p\, \top \;\Rightarrow\; \top \\
\exists p\, \top \;\Rightarrow\; \top
\end{gathered}
$$

Simplification rules for $\bot$:

$$
\begin{gathered}
\neg\bot \;\Rightarrow\; \top \\
\bot \wedge F_1 \wedge \cdots \wedge F_n \;\Rightarrow\; \bot \\
\bot \vee F_1 \vee \cdots \vee F_n \;\Rightarrow\; F_1 \vee \cdots \vee F_n \\
F \rightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \rightarrow F \;\Rightarrow\; \top \\
F \leftrightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \leftrightarrow F \;\Rightarrow\; \neg F \\
\forall p\, \bot \;\Rightarrow\; \bot \\
\exists p\, \bot \;\Rightarrow\; \bot
\end{gathered}
$$

Splitting, Example

$$\forall p \, \exists q \, (p \leftrightarrow q)$$

Splitting, Example

$$\begin{tikzcd} & \forall p\,\exists q\,(p \leftrightarrow q) \arrow[dl, "p = 0"'] \\ \exists q\,(\bot \leftrightarrow q) & \wedge \end{tikzcd}$$

Splitting, Example

$$\begin{tikzcd}
& \forall p\,\exists q\,(p \leftrightarrow q) \arrow[dl, "p = 0"'] \\
\exists q\,(\neg q) & \wedge
\end{tikzcd}$$

Splitting, Example

$$
\begin{tikzcd}
\forall p \, \exists q \, (p \leftrightarrow q) \arrow[d, "p = 0"', "\wedge"] \\
\exists q \, (\neg q) \arrow[d, "q = 0"', "\vee"] \\
(\neg \bot)
\end{tikzcd}
$$

Splitting, Example

$$
\begin{tikzcd}[column sep=small]
 & & \forall p\, \exists q\, (p \leftrightarrow q) \arrow[dl, "p = 0"'] \\
 & \exists q\, (\neg q) \arrow[dl, "q = 0"'] & \wedge \\
\boxed{1}\ \top & \vee &
\end{tikzcd}
$$

Splitting, Example

$$
\begin{tikzcd}
 & & \forall p\, \exists q\, (p \leftrightarrow q) \arrow[dl, "p = 0"'] \\
 & & \wedge \\
 & \boxed{1}\ \exists q\, (\neg q) \arrow[dl, "q = 0"'] & \\
 & \vee & \\
\boxed{1}\ \top & &
\end{tikzcd}
$$

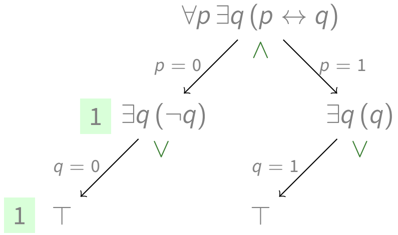
Splitting, Example

$$\begin{tikzcd}
& & \forall p\, \exists q\, (p \leftrightarrow q) \arrow[dl, "p = 0"'] \arrow[dr, "p = 1"] \arrow[d, phantom, "\wedge"] & \\
& \boxed{1}\ \exists q\, (\neg q) \arrow[dl, "q = 0"'] \arrow[d, phantom, "\vee"] & & \exists q\, (\top \leftrightarrow q) \\
\boxed{1}\ \top & & &
\end{tikzcd}$$

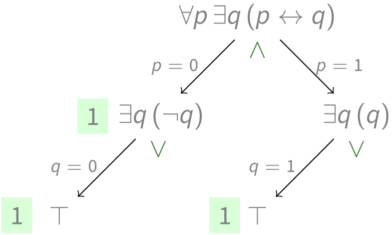
Splitting, Example

$$
\begin{tikzcd}
& & \forall p\,\exists q\,(p \leftrightarrow q) \arrow[dl, "p = 0"'] \arrow[dr, "p = 1"] \arrow[d, phantom, "\wedge"] & \\
& \boxed{1}\ \exists q\,(\neg q) \arrow[dl, "q = 0"'] \arrow[d, phantom, "\vee"] & & \exists q\,(q) \\
\boxed{1}\ \top & & &
\end{tikzcd}
$$

Splitting, Example



Splitting, Example



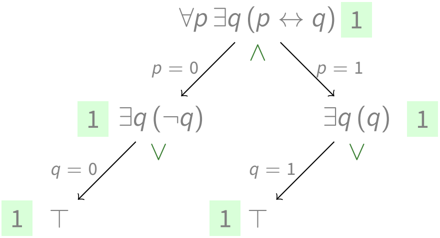
Splitting, Example

$$
\begin{tikzcd}
& & \forall p\, \exists q\, (p \leftrightarrow q) \arrow[dl, "p = 0"'] \arrow[dr, "p = 1"] & & \\
& \boxed{1}\ \exists q\, (\neg q) \arrow[dl, "q = 0"'] & \land & \exists q\, (q)\ \boxed{1} \arrow[dl, "q = 1"'] & \\
\boxed{1}\ \top & \lor & \boxed{1}\ \top & \lor &
\end{tikzcd}
$$

Splitting, Example

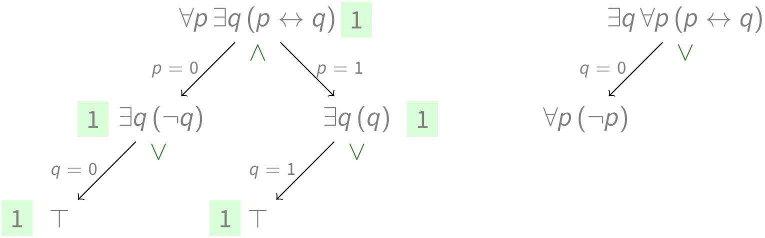
$$
\begin{tikzcd}
 & & & \forall p\, \exists q\, (p \leftrightarrow q)\ \boxed{1} \arrow[dl, "p = 0"'] \arrow[dr, "p = 1"] \arrow[d, phantom, "\wedge"] & \\
 & & \boxed{1}\ \exists q\, (\neg q) \arrow[dl, "q = 0"'] \arrow[d, phantom, "\vee"] & {} & \exists q\, (q)\ \boxed{1} \arrow[dl, "q = 1"'] \arrow[d, phantom, "\vee"] \\
 & \boxed{1}\ \top & {} & \boxed{1}\ \top & {}
\end{tikzcd}
$$

Splitting, Example

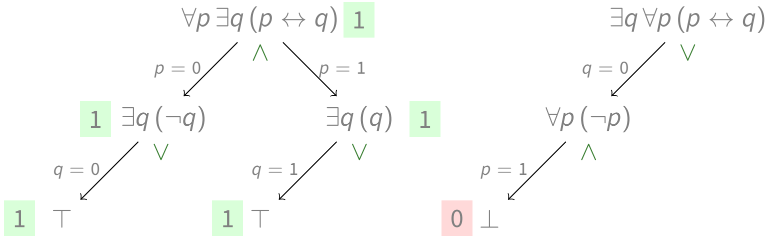


$\exists q \, \forall p \, (p \leftrightarrow q)$

Splitting, Example



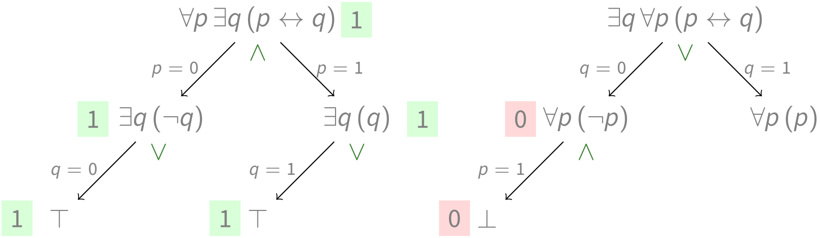
Splitting, Example



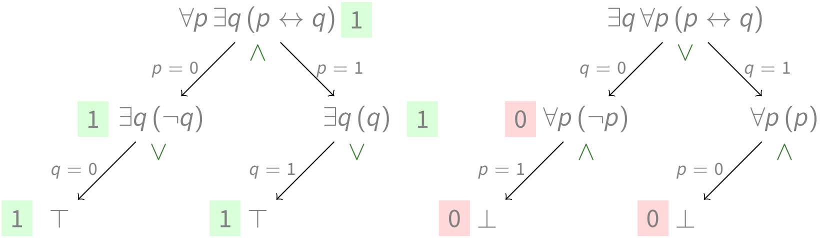
Splitting, Example

$$
\begin{tikzcd}[column sep=small]
& & \forall p\,\exists q\,(p \leftrightarrow q)\ \boxed{1} \arrow[dl, "p=0"'] \arrow[dr, "p=1"] \arrow[d, phantom, "\wedge"] & \\
& \boxed{1}\ \exists q\,(\neg q) \arrow[dl, "q=0"'] \arrow[d, phantom, "\vee"] & \ & \exists q\,(q)\ \boxed{1} \arrow[dl, "q=1"'] \arrow[d, phantom, "\vee"] \\
\boxed{1}\ \top & \ & \boxed{1}\ \top & \
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
& & \exists q\,\forall p\,(p \leftrightarrow q) \arrow[dl, "q=0"'] \arrow[d, phantom, "\vee"] \\
& \boxed{0}\ \forall p\,(\neg p) \arrow[dl, "p=1"'] \arrow[d, phantom, "\wedge"] & \ \\
\boxed{0}\ \bot & \ &
\end{tikzcd}
$$

Splitting, Example



Splitting, Example

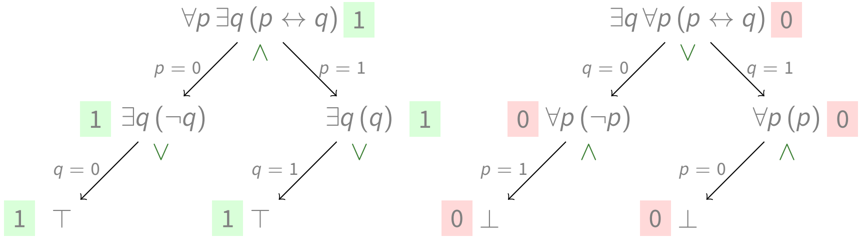


Splitting, Example

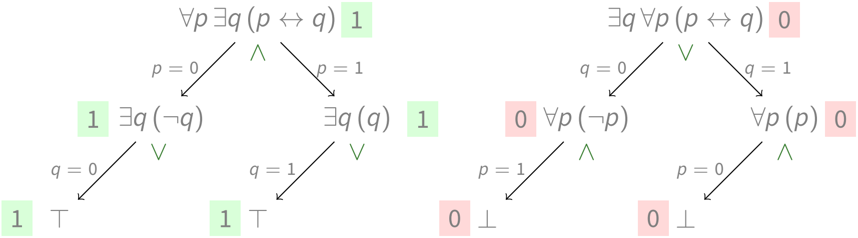
$$
\begin{tikzcd}[column sep=small]
 & & \forall p\, \exists q\, (p \leftrightarrow q)\ \boxed{1} \arrow[dl, "p = 0"'] \arrow[dr, "p = 1"] \arrow[d, phantom, "\wedge"] & & \\
 & \boxed{1}\ \exists q\, (\neg q) \arrow[dl, "q = 0"'] \arrow[d, phantom, "\vee"] & {} & \exists q\, (q)\ \boxed{1} \arrow[dl, "q = 1"'] \arrow[d, phantom, "\vee"] & \\
\boxed{1}\ \top & {} & \boxed{1}\ \top & {} &
\end{tikzcd}
$$

$$
\begin{tikzcd}[column sep=small]
 & & \exists q\, \forall p\, (p \leftrightarrow q) \arrow[dl, "q = 0"'] \arrow[dr, "q = 1"] \arrow[d, phantom, "\vee"] & & \\
 & \boxed{0}\ \forall p\, (\neg p) \arrow[dl, "p = 1"'] \arrow[d, phantom, "\wedge"] & {} & \forall p\, (p)\ \boxed{0} \arrow[dl, "p = 0"'] \arrow[d, phantom, "\wedge"] & \\
\boxed{0}\ \bot & {} & \boxed{0}\ \bot & {} &
\end{tikzcd}
$$

Splitting, Example



Splitting, Example



To minimize search, the selection of variable values is best seen as a two-player game:

- by selecting a value for $\exists q$ one is trying to make the formula true
- by selecting a value for $\forall p$ one is trying to make the formula false

Splitting algorithm

Notation: if $\boldsymbol{p} = (p_1, \ldots, p_k)$ then $\exists\!\forall \boldsymbol{p} F$ denotes $\exists\!\forall p_1 \cdots \exists\!\forall p_k F$

Splitting algorithm

procedure $splitting(F)$
input: closed rectified prenex formula F
output: 0 or 1
parameters: function $select\_variable\_value$ // selects a variable from the outermost prefix
begin // of F as well as a Boolean value for it
$F := simplify(F)$ // apply extended simplification rules to completion
if $F = \bot$ **then return** 0
if $F = \top$ **then return** 1
// else F has the form $\exists\forall \boldsymbol{p}F'$ where $\boldsymbol{p}$ is F's outermost prefix
$(p, b) := select\_variable\_value(F)$
Let G be obtained from F by deleting p from $\boldsymbol{p}$
if $b = 0$ **then** $A := \bot$; $B := \top$ **else** $A := \top$; $B := \bot$
$b := splitting(G_p^A)$
case $(b, \exists\forall)$ **of**
$(0, \forall) \Rightarrow$ **return** 0
$(0, \exists) \Rightarrow$ **return** $splitting(G_p^B)$
$(1, \forall) \Rightarrow$ **return** $splitting(G_p^B)$
$(1, \exists) \Rightarrow$ **return** 1
end

Conjunctive Normal Form

For more efficient algorithms we need QBFs to be in a convenient normal form

Conjunctive Normal Form

For more efficient algorithms we need QBFs to be in a convenient normal form

Our next aim is to modify CNF and DPLL to deal with quantified Boolean formulas

Conjunctive Normal Form

For more efficient algorithms we need QBFs to be in a convenient normal form

Our next aim is to modify CNF and DPLL to deal with quantified Boolean formulas

A quantified Boolean formula F is in *Conjunctive Normal Form (CNF)*, if

- it is either $\bot$, or $\top$, or
- it has the form
$$\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n \, (C_1 \wedge \cdots \wedge C_m)$$
  where $C_1, \ldots, C_m$ are clauses

Conjunctive Normal Form

For more efficient algorithms we need QBFs to be in a convenient normal form

Our next aim is to modify CNF and DPLL to deal with quantified Boolean formulas

A quantified Boolean formula F is in *Conjunctive Normal Form (CNF)*, if

- it is either $\bot$, or $\top$, or
- it has the form

$$\exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_n p_n \, (C_1 \wedge \cdots \wedge C_m)$$

  where $C_1, \ldots, C_m$ are clauses

Example:

$$\forall p \, \exists q \, \exists s \, ((\neg p \vee s \vee q) \wedge (s \vee \neg q) \wedge \neg s)$$

CNF rules

Prenexing rules

+

propositional CNF rules:

$$
\begin{array}{rcl}
F \leftrightarrow G & \Rightarrow & (\neg F \vee G) \wedge (\neg G \vee F) \\
F \rightarrow G & \Rightarrow & \neg F \vee G \\
\neg(F \wedge G) & \Rightarrow & \neg F \vee \neg G \\
\neg(F \vee G) & \Rightarrow & \neg F \wedge \neg G \\
\neg\neg F & \Rightarrow & F \\
(F_1 \wedge \cdots \wedge F_m) \vee G_1 \vee \cdots \vee G_n & \Rightarrow & (F_1 \vee G_1 \vee \cdots \vee G_n) \quad \wedge \\
 & & \cdots \quad \wedge \\
 & & (F_m \vee G_1 \vee \cdots \vee G_n)
\end{array}
$$

DPLL for quantified Boolean formulas

Input:

Q: quantifier sequence $\exists\!\!\forall_1 \boldsymbol{p}_1 \cdots \exists\!\!\forall_n \boldsymbol{p}_n$

$\boldsymbol{S}$: set of clauses with variables from $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n$

Main components:

Unit propagation

Splitting on literals

Unit Propagation

Q: quantifier sequence
$\boldsymbol{S}$: current clause set

Propositional formulas:

For each unit clause L in $\boldsymbol{S}$

1. remove all clauses containing literal L from $\boldsymbol{S}$
2. remove every literal $\overline{L}$ from remaining clauses

Quantified Boolean formulas:

For each unit clause L in $\boldsymbol{S}$ of the form p or $\neg p$

- If Q does not contain p or contains $\exists p$,
  1. remove all clauses containing literal L from $\boldsymbol{S}$
  2. remove every literal $\overline{L}$ from remaining clauses
- otherwise (Q contains $\forall p$), add $\square$ to $\boldsymbol{S}$

Unit Propagation

Q: quantifier sequence
$\boldsymbol{S}$: current clause set

Propositional formulas:

For each unit clause L in $\boldsymbol{S}$

1. remove all clauses containing literal L from $\boldsymbol{S}$
2. remove every literal $\overline{L}$ from remaining clauses

Quantified Boolean formulas:

For each unit clause L in $\boldsymbol{S}$ of the form p or $\neg p$

- If Q does not contain p or contains $\exists p$,
    1. remove all clauses containing literal L from $\boldsymbol{S}$
    2. remove every literal $\overline{L}$ from remaining clauses
- otherwise (Q contains $\forall p$), add $\square$ to $\boldsymbol{S}$

Unit Propagation

Q: quantifier sequence
$\boldsymbol{S}$: current clause set

Propositional formulas:

For each unit clause L in $\boldsymbol{S}$

1. remove all clauses containing literal L from $\boldsymbol{S}$
2. remove every literal $\overline{L}$ from remaining clauses

Quantified Boolean formulas:

For each unit clause L in $\boldsymbol{S}$ of the form p or $\neg p$

- If Q does not contain p or contains $\exists p$,
  1. remove all clauses containing literal L from $\boldsymbol{S}$
  2. remove every literal $\overline{L}$ from remaining clauses
- otherwise (Q contains $\forall p$), add $\square$ to $\boldsymbol{S}$

Unit Propagation

Q: quantifier sequence
$\boldsymbol{S}$: current clause set

Propositional formulas:

For each unit clause L in $\boldsymbol{S}$

1. remove all clauses containing literal L from $\boldsymbol{S}$
2. remove every literal $\overline{L}$ from remaining clauses

Quantified Boolean formulas:

For each unit clause L in $\boldsymbol{S}$ of the form p or $\neg p$

- If Q does not contain p or contains $\exists p$,
    1. remove all clauses containing literal L from $\boldsymbol{S}$
    2. remove every literal $\overline{L}$ from remaining clauses
- otherwise (Q contains $\forall p$), add $\Box$ to $\boldsymbol{S}$

DPLL algorithm

Why do we add $\Box$ to $\boldsymbol{S}$ when Q is $\forall p \, \exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_m \, p_m$ and
$\boldsymbol{S}$ is $\{ \, p, C_1, \ldots, C_n \, \}$?

DPLL algorithm

Why do we add $\Box$ to $\boldsymbol{S}$ when Q is $\forall p \, \exists\forall_1 p_1 \cdots \exists\forall_m \, p_m$ and $\boldsymbol{S}$ is $\{ \, p, C_1, \ldots, C_n \, \}$?

Because

1. The intended input formula is
   $G \; = \; \forall p \, \exists\forall_1 q_1 \cdots \exists\forall_m q_m \, (p \wedge C_1 \wedge \cdots \wedge C_m)$
2. $G \; = \; \exists\forall_1 q_1 \cdots \exists\forall_m q_m \, ((p \wedge C_1 \wedge \cdots \wedge C_m)^{\perp}_{p} \wedge (p \wedge C_1 \wedge \cdots \wedge C_m)^{\top}_{p})$

DPLL algorithm

Why do we add $\Box$ to $\boldsymbol{S}$ when Q is $\forall p \, \exists\!\!\forall_1 p_1 \cdots \exists\!\!\forall_m \, p_m$ and $\boldsymbol{S}$ is $\{ \, p, C_1, \ldots, C_n \, \}$?

Because

1. The intended input formula is
   $G \; = \; \forall p \, \exists\!\!\forall_1 q_1 \cdots \exists\!\!\forall_m q_m \, (p \land C_1 \land \cdots \land C_m)$
2. $G \; \equiv \; \exists\!\!\forall_1 q_1 \cdots \exists\!\!\forall_m q_m \, ((p \land C_1 \land \cdots \land C_m)^{\perp}_{p} \land (p \land C_1 \land \cdots \land C_m)^{\top}_{p})$

DPLL algorithm

Why do we add $\square$ to $\boldsymbol{S}$ when Q is $\forall p\, \exists\forall_1 p_1 \cdots \exists\forall_m\, p_m$ and
$\boldsymbol{S}$ is $\{\, p, C_1, \ldots, C_n \,\}$?

Because

1. The intended input formula is
$$G \;=\; \forall p\, \exists\forall_1 q_1 \cdots \exists\forall_m q_m \,(p \wedge C_1 \wedge \cdots \wedge C_m)$$
2. $$\begin{aligned} G \;&\equiv\; \exists\forall_1 q_1 \cdots \exists\forall_m q_m \,((p \wedge C_1 \wedge \cdots \wedge C_m)_p^{\perp} \wedge (p \wedge C_1 \wedge \cdots \wedge C_m)_p^{\top}) \\ &=\; \exists\forall_1 q_1 \cdots \exists\forall_m q_m \,(\perp \wedge (C_1 \wedge \cdots \wedge C_m)_p^{\perp} \wedge (p \wedge C_1 \wedge \cdots \wedge C_m)_p^{\top}) \end{aligned}$$

DPLL algorithm

Why do we add $\square$ to $\boldsymbol{S}$ when Q is $\forall p\, \exists\!\!\!\forall_1 p_1 \cdots \exists\!\!\!\forall_m\, p_m$ and $\boldsymbol{S}$ is $\{\, p, C_1, \ldots, C_n \,\}$?

Because

1. The intended input formula is
$$G \;=\; \forall p\, \exists\!\!\!\forall_1 q_1 \cdots \exists\!\!\!\forall_m q_m\, (p \wedge C_1 \wedge \cdots \wedge C_m)$$
2. $$\begin{aligned} G &\equiv \exists\!\!\!\forall_1 q_1 \cdots \exists\!\!\!\forall_m q_m \left((p \wedge C_1 \wedge \cdots \wedge C_m)_p^{\bot} \wedge (p \wedge C_1 \wedge \cdots \wedge C_m)_p^{\top} \right) \\ &= \exists\!\!\!\forall_1 q_1 \cdots \exists\!\!\!\forall_m q_m \left(\bot \wedge (C_1 \wedge \cdots \wedge C_m)_p^{\bot} \wedge (p \wedge C_1 \wedge \cdots \wedge C_m)_p^{\top} \right) \\ &\equiv \exists\!\!\!\forall_1 q_1 \cdots \exists\!\!\!\forall_m q_m\, \bot \\ &\equiv \bot \end{aligned}$$

DPLL algorithm

Why do we add $\Box$ to $\boldsymbol{S}$ when Q is $\forall p \, \exists\!\!\forall_1 p_1 \cdots \exists\!\!\forall_m p_m$ and $\boldsymbol{S}$ is $\{ p, C_1, \ldots, C_n \}$?

Alternatively, using the game metaphor, because

the $\forall$-player wants to falsify the formula

DPLL algorithm

Why do we add $\square$ to $\boldsymbol{S}$ when Q is $\forall p \, \exists\!\!\forall_1 p_1 \cdots \exists\!\!\forall_m \, p_m$ and
$\boldsymbol{S}$ is $\{ p, C_1, \ldots, C_n \}$?

Alternatively, using the game metaphor, because

the $\forall$-player wants to falsify the formula

Winning move for the $\forall$-player:

select the value for p that falsifies the unit clause p, and hence the whole CNF

DPLL algorithm

Why do we add $\Box$ to $\boldsymbol{S}$ when Q is $\forall p \, \exists\!\!\forall_1 p_1 \cdots \exists\!\!\forall_m \, p_m$ and
$\boldsymbol{S}$ is $\{\, p, C_1, \ldots, C_n \,\}$?

Alternatively, using the game metaphor, because

the $\forall$-player wants to falsify the formula

Winning move for the $\forall$-player:

select the value for p that falsifies the unit clause p, and hence the whole CNF

(argument is similar for $\{\, \neg p, C_1, \ldots, C_n \,\}$)

DPLL, Example

$\exists p \, \forall q \, \exists r$

$p \vee q \vee \neg r$

$p \vee \neg q \vee r$

$\neg p \vee q \vee r$

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

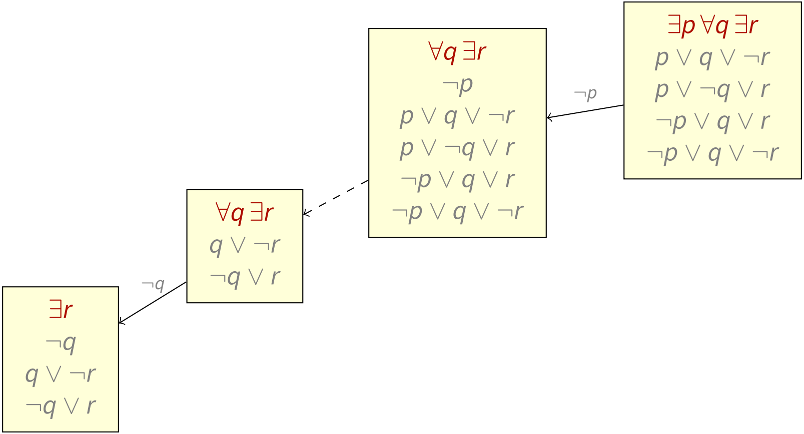
DPLL, Example

$$\begin{tikzcd}
\boxed{\begin{array}{c} \forall q\,\exists r \\ \neg p \\ p \lor q \lor \neg r \\ p \lor \neg q \lor r \\ \neg p \lor q \lor r \\ \neg p \lor q \lor \neg r \end{array}} & \boxed{\begin{array}{c} \exists p\,\forall q\,\exists r \\ p \lor q \lor \neg r \\ p \lor \neg q \lor r \\ \neg p \lor q \lor r \\ \neg p \lor q \lor \neg r \end{array}} \arrow[l, "\neg p"']
\end{tikzcd}$$

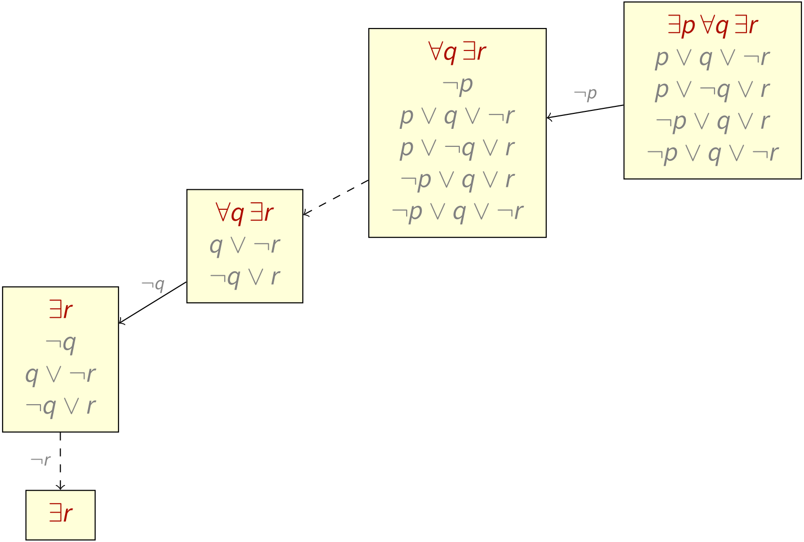
DPLL, Example

$$
\begin{tikzcd}
\boxed{\begin{array}{c} \forall q\, \exists r \\ q \lor \neg r \\ \neg q \lor r \end{array}} & \boxed{\begin{array}{c} \forall q\, \exists r \\ \neg p \\ p \lor q \lor \neg r \\ p \lor \neg q \lor r \\ \neg p \lor q \lor r \\ \neg p \lor q \lor \neg r \end{array}} \arrow[l, dashed] & \boxed{\begin{array}{c} \exists p\, \forall q\, \exists r \\ p \lor q \lor \neg r \\ p \lor \neg q \lor r \\ \neg p \lor q \lor r \\ \neg p \lor q \lor \neg r \end{array}} \arrow[l, "\neg p"']
\end{tikzcd}
$$

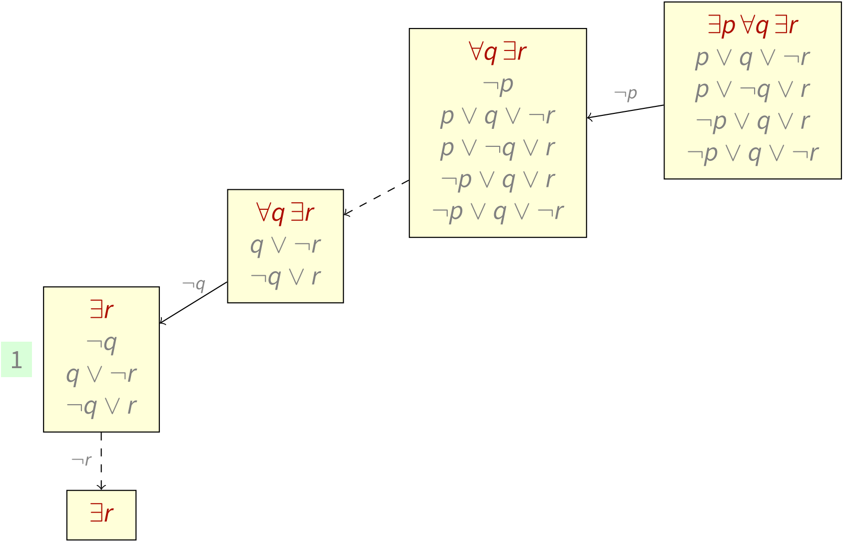
DPLL, Example



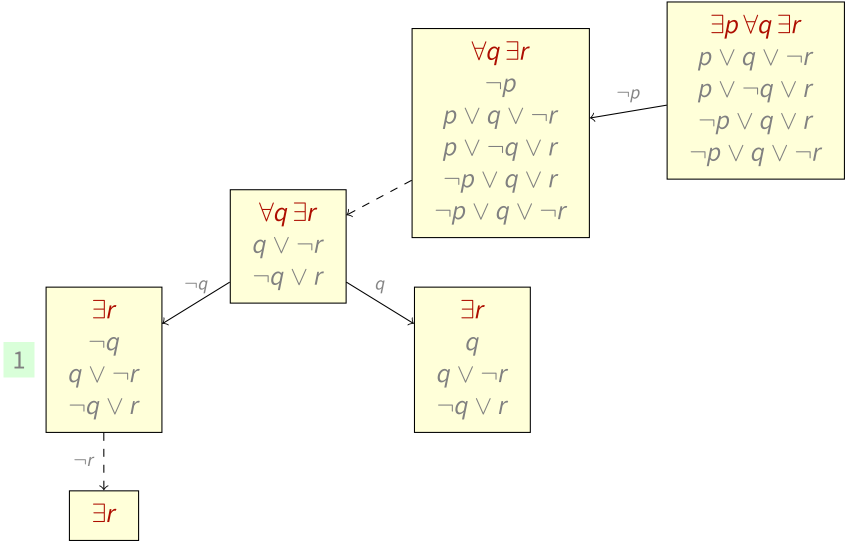
DPLL, Example



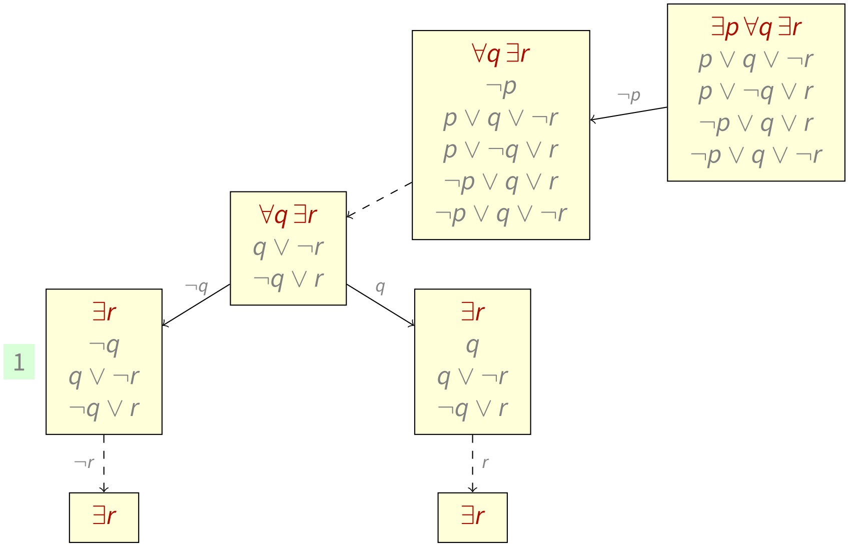
DPLL, Example



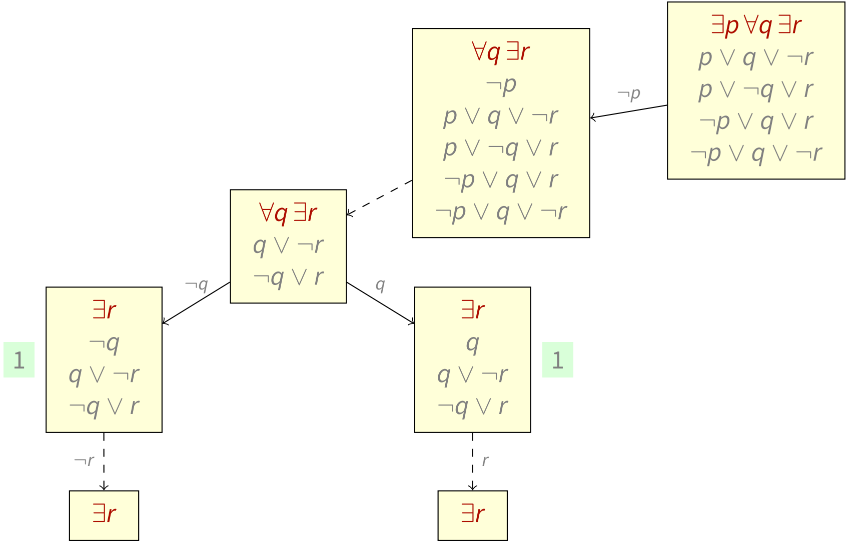
DPLL, Example



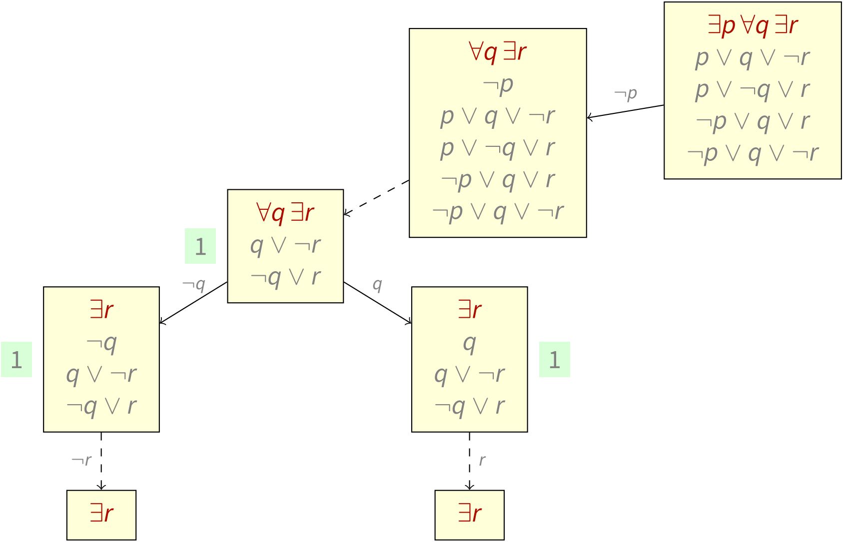
DPLL, Example



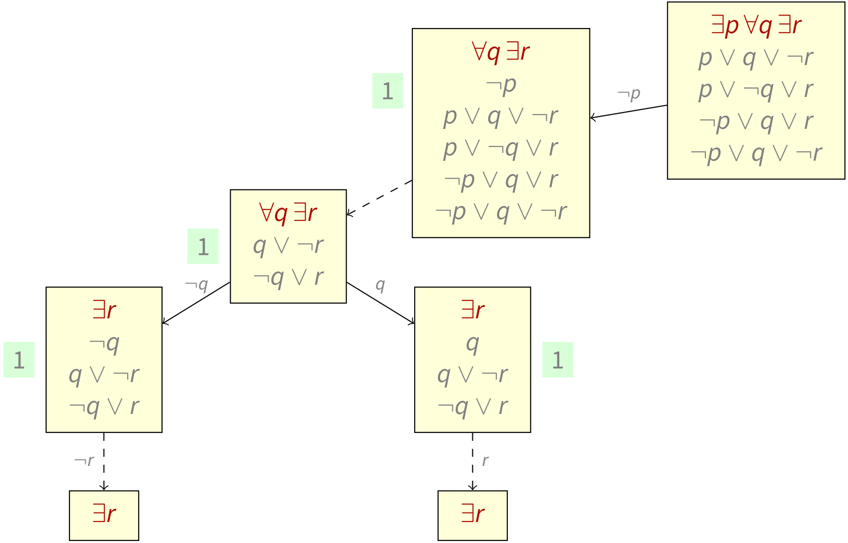
DPLL, Example



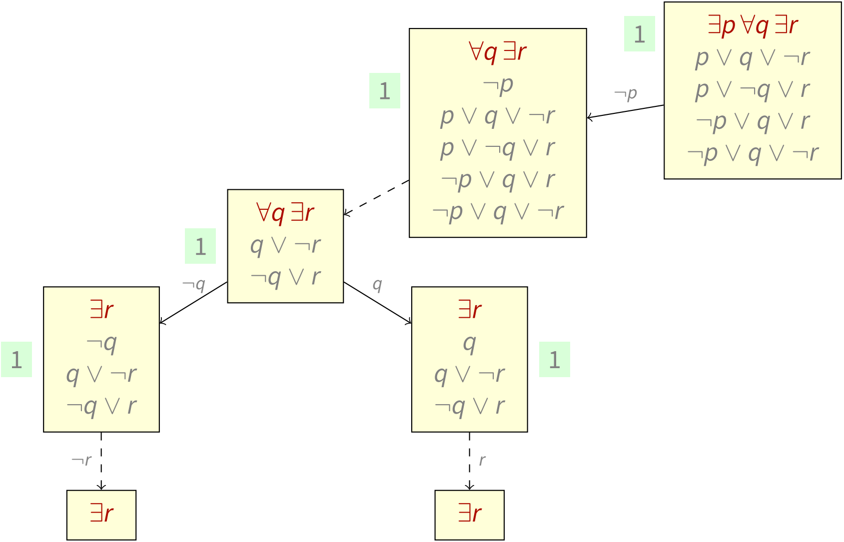
DPLL, Example



DPLL, Example



DPLL, Example



DPLL algorithm

procedure $DPLL(Q, S)$
input: quantifier sequence $Q = \exists\!\forall_1 \boldsymbol{p}_1 \cdots \exists\!\forall_n \boldsymbol{p}_n$,
clause set $\boldsymbol{S}$ with vars from Q
output: 0 or 1
parameters: function $select\_variable\_value$
begin
$\boldsymbol{S} \ := \ unit\_propagate(Q, S)$
if $\boldsymbol{S}$ is empty **then return** 1
if $\boldsymbol{S}$ contains $\Box$ **then return** 0
$(p, b) \ := \ select\_variable\_value(\boldsymbol{p}_1, S)$
Let Q' be obtained from Q by deleting $\exists\!\forall_1 p$ from $\exists\!\forall_1 \boldsymbol{p}_1$
if $b = 0$ **then** $L \ := \ \neg p$
else $L \ := \ p$
case $(DPLL(Q', S \cup \{ L \}), \ \exists\!\forall)$ **of**
$(0, \forall) \Rightarrow$ **return** 0
$(0, \exists) \Rightarrow$ **return** $DPLL(Q', S \cup \{ \bar{L} \})$
$(1, \forall) \Rightarrow$ **return** $DPLL(Q', S \cup \{ \bar{L} \})$
$(1, \exists) \Rightarrow$ **return** 1
end

Improving DPLL with further simplifications

$$\exists p\,\exists q\,\forall r\,\exists s\,((p \lor \neg r) \land (\neg q \lor r) \land (\neg p \lor q \lor s) \land (\neg p \lor q \lor r \lor \neg s))$$

- We can treat $\neg r$ in $p \lor \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s))$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s))$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s))$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s))$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s))$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \wedge (q \vee s) \wedge (q \vee \neg s))$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \lor \neg r) \land (\neg q \lor r) \land (\neg p \lor q \lor s) \land (\neg p \lor q \lor r \lor \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \land (\neg q \lor r) \land (\neg p \lor q \lor s) \land (\neg p \lor q \lor r \lor \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \lor r) \land (q \lor s) \land (q \lor r \lor \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \land (q \lor s) \land (q \lor \neg s))$$

- We can treat $\neg r$ in $p \lor \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \wedge (q \vee s) \wedge (q \vee \neg s)) \Rightarrow$$
$$\exists s\, (s \wedge \neg s)$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Improving DPLL with further simplifications

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \wedge (q \vee s) \wedge (q \vee \neg s)) \Rightarrow$$
$$\exists s\, (s \wedge \neg s) \Rightarrow$$
$$\square$$

- We can treat $\neg r$ in $p \vee \neg r$ as 0 without loss of generality
- We can apply unit propagation
- We can treat r as 0 everywhere without loss of generality
- We can apply unit propagation with $\neg q$
- We can apply unit propagation with s

Pure literal rule

Q: quantifier sequence
$\boldsymbol{S}$: current clause set
L: literal of the form p or $\neg p$

Suppose L is *pure* in $\boldsymbol{S}$ (i.e., $\overline{L}$ does not occur in $\boldsymbol{S}$). Then:

- If p is existentially quantified in Q, we can remove all clauses containing L

Pure literal rule

Q: quantifier sequence
$\boldsymbol{S}$: current clause set
L: literal of the form p or $\neg p$

Suppose L is *pure* in $\boldsymbol{S}$ (i.e., $\overline{L}$ does not occur in $\boldsymbol{S}$). Then:

- If p is existentially quantified in Q, we can remove all clauses containing L
- if p is universally quantified in Q, we can remove L from all clauses

Pure literal rule

Q: quantifier sequence
$\boldsymbol{S}$: current clause set
L: literal of the form p or $\neg p$

Suppose L is *pure* in $\boldsymbol{S}$ (i.e., $\overline{L}$ does not occur in $\boldsymbol{S}$). Then:

- If p is existentially quantified in Q, we can remove all clauses containing L
- if p is universally quantified in Q, we can remove L from all clauses

Why?

Pure literal rule

Q: quantifier sequence
$\boldsymbol{S}$: current clause set
L: literal of the form p or $\neg p$

Suppose L is *pure* in $\boldsymbol{S}$ (i.e., $\overline{L}$ does not occur in $\boldsymbol{S}$). Then:

- If p is existentially quantified in Q, we can remove all clauses containing L
- if p is universally quantified in Q, we can remove L from all clauses

Why?

- The $\exists$-player will make L true (satisfying all clauses containing L)

Pure literal rule

Q: quantifier sequence
$\boldsymbol{S}$: current clause set
L: literal of the form p or $\neg p$

Suppose L is *pure* in $\boldsymbol{S}$ (i.e., $\overline{L}$ does not occur in $\boldsymbol{S}$). Then:

- If p is existentially quantified in Q, we can remove all clauses containing L
- if p is universally quantified in Q, we can remove L from all clauses

Why?

- The $\exists$-player will make L true (satisfying all clauses containing L)
- The $\forall$-player will make L false (so it can be removed from all clauses containing L)

Universal literal deletion

Q: quantifier sequence
$\boldsymbol{S}$: clause set
p, q: variables

- p is *existential in* Q if Q contains $\exists p$
- q is *universal in* Q if Q contains $\forall q$

Universal literal deletion

Q: quantifier sequence
$\boldsymbol{S}$: clause set
p, q: variables

- p is *existential in* Q if Q contains $\exists p$
- q is *universal in* Q if Q contains $\forall q$
- p is *quantified before* a variable q if p occurs before q in Q

Universal literal deletion

Q: quantifier sequence
$\boldsymbol{S}$: clause set
p, q: variables

- p is *existential in* Q if Q contains $\exists p$
- q is *universal in* Q if Q contains $\forall q$
- p is *quantified before* a variable q if p occurs before q in Q

Example: In $Q = \forall q \, \exists p \, \forall r$
q is quantified before both p and r; and p is quantified before r

Universal literal deletion

Q: quantifier sequence
$\boldsymbol{S}$: clause set
p, q: variables

- p is *existential in* Q if Q contains $\exists p$
- q is *universal in* Q if Q contains $\forall q$
- p is *quantified before* a variable q if p occurs before q in Q

Theorem 10
Suppose that

1. C *is a clause in* $\boldsymbol{S}$;
2. *a variable* q *in a literal* L *of* C *is universal in* Q;
3. *all existential variables of* Q *in* C *are quantified before* q.

Then deleting L *from* C *does not change the truth value of* $Q\,S$.

Universal literal deletion

Intuition behind Theorem 10

Consider a clause C from $\boldsymbol{S}$ of the form

$$L_1 \vee \cdots \vee L_n \vee (\neg)q_1 \vee \cdots \vee (\neg)q_m$$

where all existential variables of Q in C are quantified before $q_1, \ldots, q_m$

- If at least one of $L_1, \ldots, L_n$ is true,
  then C is true regardless of the truth value of of $(\neg)q_1, \ldots, (\neg)q_m$
- If all of $L_1, \ldots, L_n$ are false,
  the $\forall$-player will make all $(\neg)q_1, \ldots, (\neg)q_m$ false and win the game

Universal literal deletion

Intuition behind Theorem 10

Consider a clause C from $\boldsymbol{S}$ of the form

$$L_1 \vee \cdots \vee L_n \vee (\neg)q_1 \vee \cdots \vee (\neg)q_m$$

where all existential variables of Q in C are quantified before $q_1, \ldots, q_m$

Consider the position before the $q_1, \ldots, q_m$-moves of the $\forall$-player

- If at least one of $L_1, \ldots, L_n$ is true,
  then C is true regardless of the truth value of of $(\neg)q_1, \ldots, (\neg)q_m$
- If all of $L_1, \ldots, L_n$ are false,
  the $\forall$-player will make all $(\neg)q_1, \ldots, (\neg)q_m$ false and win the game

Universal literal deletion

Intuition behind Theorem 10

Consider a clause C from $\boldsymbol{S}$ of the form

$$L_1 \vee \cdots \vee L_n \vee (\neg)q_1 \vee \cdots \vee (\neg)q_m$$

where all existential variables of Q in C are quantified before $q_1, \ldots, q_m$

Consider the position before the $q_1, \ldots, q_m$-moves of the $\forall$-player

- If at least one of $L_1, \ldots, L_n$ is true,
  then C is true regardless of the truth value of of $(\neg)q_1, \ldots, (\neg)q_m$
- If all of $L_1, \ldots, L_n$ are false,
  the $\forall$-player will make all $(\neg)q_1, \ldots, (\neg)q_m$ false and win the game

Universal literal deletion

Intuition behind Theorem 10

Consider a clause C from $\boldsymbol{S}$ of the form

$$L_1 \vee \cdots \vee L_n \vee (\neg)q_1 \vee \cdots \vee (\neg)q_m$$

where all existential variables of Q in C are quantified before $q_1, \ldots, q_m$

Consider the position before the $q_1, \ldots, q_m$-moves of the $\forall$-player

- If at least one of $L_1, \ldots, L_n$ is true,
  then C is true regardless of the truth value of of $(\neg)q_1, \ldots, (\neg)q_m$
- If all of $L_1, \ldots, L_n$ are false,
  the $\forall$-player will make all $(\neg)q_1, \ldots, (\neg)q_m$ false and win the game

Universal literal deletion

Intuition behind Theorem 10

Consider a clause C from $\boldsymbol{S}$ of the form

$$L_1 \vee \cdots \vee L_n \vee (\neg)q_1 \vee \cdots \vee (\neg)q_m$$

where all existential variables of Q in C are quantified before $q_1, \ldots, q_m$

Consider the position before the $q_1, \ldots, q_m$-moves of the $\forall$-player

- If at least one of $L_1, \ldots, L_n$ is true,
  then C is true regardless of the truth value of of $(\neg)q_1, \ldots, (\neg)q_m$
- If all of $L_1, \ldots, L_n$ are false,
  the $\forall$-player will make all $(\neg)q_1, \ldots, (\neg)q_m$ false and win the game

In either case, the deletion of $(\neg)q_1, \ldots, (\neg)q_m$ will not change the final outcome

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s))$$

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s))$$

- Apply *universal literal deletion* to $p \vee \neg r$

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \lor \neg r) \land (\neg q \lor r) \land (\neg p \lor q \lor s) \land (\neg p \lor q \lor r \lor \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \land (\neg q \lor r) \land (\neg p \lor q \lor s) \land (\neg p \lor q \lor r \lor \neg s))$$

- Apply *universal literal deletion* to $p \lor \neg r$

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s))$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation

Example revisited

$$\exists p \, \exists q \, \forall r \, \exists s \, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p \, \exists q \, \forall r \, \exists s \, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q \, \forall r \, \exists s \, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s))$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s))$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation
- Apply the *pure literal rule* to r

Example revisited

$$\exists p\,\exists q\,\forall r\,\exists s\,((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\,\exists q\,\forall r\,\exists s\,(p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\,\forall r\,\exists s\,((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\,\exists s\,(\neg q \wedge (q \vee s) \wedge (q \vee \neg s))$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation
- Apply the *pure literal rule* to r

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \wedge (q \vee s) \wedge (q \vee \neg s))$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation
- Apply the *pure literal rule* to r
- Apply unit propagation

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \wedge (q \vee s) \wedge (q \vee \neg s)) \Rightarrow$$
$$\exists s\, (s \wedge \neg s)$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation
- Apply the *pure literal rule* to r
- Apply unit propagation

Example revisited

$$\exists p\, \exists q\, \forall r\, \exists s\, ((p \vee \neg r) \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists p\, \exists q\, \forall r\, \exists s\, (p \wedge (\neg q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg p \vee q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \forall r\, \exists s\, ((\neg q \vee r) \wedge (q \vee s) \wedge (q \vee r \vee \neg s)) \Rightarrow$$
$$\exists q\, \exists s\, (\neg q \wedge (q \vee s) \wedge (q \vee \neg s)) \Rightarrow$$
$$\exists s\, (s \wedge \neg s) \Rightarrow$$
$$\square$$

- Apply *universal literal deletion* to $p \vee \neg r$
- Apply unit propagation
- Apply the *pure literal rule* to r
- Apply unit propagation