Outline

Quantified Boolean Formulas

- Syntax and Semantics
- Free and Bound Variables
- Satisfiability Checking
- CNF
- DPLL
- Quantified Boolean Formulas and OBDDs

Two-Player Games

Who is this man?

Two-Player Games

Does Garry Kasparov have a winning strategy?

Two-Player Games

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

Two-Player Games

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

Given a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$.
There are two players: P and Q.
At step k each player makes a move:

Two-Player Games

Given a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$.
There are two players: P and Q.
At step k each player makes a move:

1. the player P can choose a boolean value for the variable p_k;

Two-Player Games

Given a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$.
There are two players: P and Q.
At step k each player makes a move:

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

Two-Player Games

Given a propositional formula G with variables $p_1, q_1, \ldots, p_n, q_n$.
There are two players: P and Q.
At step k each player makes a move:

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

The player P wins if after n steps the chosen values make the formula G true.

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1
2. $p_1 \to q_1$

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1
2. $p_1 \to q_1$
3. $q_1 \to q_1$

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1
2. $p_1 \to q_1$
3. $q_1 \to q_1$ If G is valid, P always wins!

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1
2. $p_1 \to q_1$
3. $q_1 \to q_1$ If G is valid, P always wins!
4. $p_1 \wedge \neg p_1$

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1
2. $p_1 \rightarrow q_1$
3. $q_1 \rightarrow q_1$ If G is valid, P always wins!
4. $p_1 \wedge \neg p_1$ If G is unsatisfiable, Q always wins!

Suppose Both Players Make no Errors: Who Wins?

Consider several special cases

1. p_1
2. $p_1 \rightarrow q_1$
3. $q_1 \rightarrow q_1$ If G is valid, P always wins!
4. $p_1 \wedge \neg p_1$ If G is unsatisfiable, Q always wins!
5. $p_1 \leftrightarrow q_1$

Winning Strategy

Problem: does P have a winning strategy?

Winning Strategy

Problem: does P have a winning strategy?
He has a winning strategy if

- there exists a move for P (a boolean value for for p_1) such that

Winning Strategy

Problem: does P have a winning strategy?
He has a winning strategy if

- there exists a move for P (a boolean value for for p_1) such that
- for all moves of Q (boolean values for for q_1)

Winning Strategy

Problem: does P have a winning strategy?
He has a winning strategy if

- there exists a move for P (a boolean value for for p_1) such that
- for all moves of Q (boolean values for for q_1)
- there exists a move for P (a boolean value for for p_2) such that

Winning Strategy

Problem: does P have a winning strategy?
He has a winning strategy if

- there exists a move for P (a boolean value for for p_1) such that
- for all moves of Q (boolean values for for q_1)
- there exists a move for P (a boolean value for for p_2) such that
- for all moves of Q (boolean values for for q_2)

Winning Strategy

Problem: does P have a winning strategy?
He has a winning strategy if

- there exists a move for P (a boolean value for for p_1) such that
- for all moves of Q (boolean values for for q_1)
- there exists a move for P (a boolean value for for p_2) such that
- for all moves of Q (boolean values for for q_2)
- ...
- for all moves of Q (boolean values for for q_n) the formula G becomes true.

Winning Strategy

Problem: does P have a winning strategy?
He has a winning strategy if

- there exists a move for P (a boolean value for for p_1) such that
- for all moves of Q (boolean values for for q_1)
- there exists a move for P (a boolean value for for p_2) such that
- for all moves of Q (boolean values for for q_2)
- ...
- for all moves of Q (boolean values for for q_n) the formula G becomes true.

The existence of a winning strategy can be expressed by a quantified boolean formula $\exists p_1 \forall q_1 \exists p_2 \forall q_2 \dots \exists p_n \forall q_n G$.

Quantified Boolean Formulas

Propositional formula:

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

Quantified Boolean Formulas

Propositional formula:

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

Quantified boolean formulas:

- If p is a boolean variable and F is a formula, then $\forall p F$ and $\exists p F$ are formulas.

Quantifiers

- $\forall$ is called the universal quantifier.
- $\exists$ is called the existential quantifier.
- Read $\forall pF$ as "for all p, F".
- Read $\exists pF$ as "there exists p such that F" or "for some p, F".

New Notation

Define

$$I_p^b(q) \stackrel{\text{def}}{=} \begin{cases} I(q), & \text{if } p \neq q; \\ b, & \text{if } p = q. \end{cases}$$

Example: let $I = \{p \mapsto 1, q \mapsto 0, r \mapsto 1\}$. Then

$$\begin{array}{lcllll} I_q^1 & = & \{p \mapsto 1, q \mapsto 1, r \mapsto 1\} & & \\ I_q^0 & = & \{p \mapsto 1, q \mapsto 0, r \mapsto 1\} & = & I \\ I_p^0 & = & \{p \mapsto 0, q \mapsto 0, r \mapsto 1\} & & \end{array}$$

Semantics

1. $I(\top) = 1$ and $I(\bot) = 0$.
2. $I(F_1 \wedge \ldots \wedge F_n) = 1$ if and only if $I(F_i) = 1$ for all i.
3. $I(F_1 \vee \ldots \vee F_n) = 1$ if and only if $I(F_i) = 1$ for some i.
4. $I(\neg F) = 1$ if and only if $I(F) = 0$.
5. $I(F \rightarrow G) = 1$ if and only if $I(F) = 0$ or $I(G) = 1$.
6. $I(F \leftrightarrow G) = 1$ if and only if $I(F) = I(G)$.

Semantics

1. $I(\top) = 1$ and $I(\bot) = 0$.
2. $I(F_1 \wedge \ldots \wedge F_n) = 1$ if and only if $I(F_i) = 1$ for all i.
3. $I(F_1 \vee \ldots \vee F_n) = 1$ if and only if $I(F_i) = 1$ for some i.
4. $I(\neg F) = 1$ if and only if $I(F) = 0$.
5. $I(F \rightarrow G) = 1$ if and only if $I(F) = 0$ or $I(G) = 1$.
6. $I(F \leftrightarrow G) = 1$ if and only if $I(F) = I(G)$.
7. $I(\forall p F) = 1$ if and only if $I_p^0(F) = 1$ and $I_p^1(F) = 1$.
8. $I(\exists p F) = 1$ if and only if $I_p^0(F) = 1$ or $I_p^1(F) = 1$.

Evaluating a Formula: and-or trees

Let us evaluate $\forall p \exists q (p \leftrightarrow q)$ on the interpretation $\{p \mapsto 1, q \mapsto 0\}$.

Evaluating a Formula: and-or trees

Let us evaluate $\forall p \exists q(p \leftrightarrow q)$ on the interpretation $\{p \mapsto 1, q \mapsto 0\}$.
Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$.

$$I_{10} \models \forall p \exists q(p \leftrightarrow q)$$

Evaluating a Formula: and-or trees

Let us evaluate $\forall p \exists q (p \leftrightarrow q)$ on the interpretation $\{p \mapsto 1, q \mapsto 0\}$.
Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$.

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

Evaluating a Formula: and-or trees

Let us evaluate $\forall p \exists q (p \leftrightarrow q)$ on the interpretation $\{p \mapsto 1, q \mapsto 0\}$.
Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$.

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

$$\Leftrightarrow \quad \boxed{\begin{array}{ll} \boxed{\begin{array}{ll} I_{00} \models p \leftrightarrow q & \\ I_{01} \models p \leftrightarrow q & \text{or} \end{array}} & \\ & \text{and} \\ & \end{array}}$$

Evaluating a Formula: and-or trees

Let us evaluate $\forall p \exists q(p \leftrightarrow q)$ on the interpretation $\{p \mapsto 1, q \mapsto 0\}$.
Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$.

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

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

Evaluating a formula

$$I_{10} \models \forall p \exists q (p \leftrightarrow q)$$

$$\wedge$$

$$I_{00} \models \exists q (p \leftrightarrow q) \qquad\qquad I_{10} \models \exists q (p \leftrightarrow q)$$

$$\vee \qquad\qquad\qquad\qquad \vee$$

$$I_{00} \models p \leftrightarrow q \qquad I_{01} \models p \leftrightarrow q \qquad I_{10} \models p \leftrightarrow q \qquad I_{11} \models p \leftrightarrow q$$

Evaluating a formula

Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$. Use wildcards $*$ to denote "any" boolean value.

$$I_{**} \models \forall p \exists q (p \leftrightarrow q)$$

Evaluating a formula

Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$. Use wildcards $*$ to denote “any” boolean value.

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

Evaluating a formula

Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$. Use wildcards $*$ to denote "any" boolean value.

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

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

Evaluating a formula

Denote any interpretation $\{p \mapsto b_1, q \mapsto b_2\}$ by $I_{b_1 b_2}$. Use wildcards $*$ to denote "any" boolean value.

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

$$\Leftrightarrow \quad \boxed{\begin{array}{c} \boxed{\begin{array}{l} I_{00} \models p \leftrightarrow q \\ I_{01} \models p \leftrightarrow q \end{array} \text{ or}} \\ \\ \boxed{\begin{array}{l} I_{10} \models p \leftrightarrow q \\ I_{11} \models p \leftrightarrow q \end{array} \text{ or}} \end{array} \text{ and}}$$

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

Subformula

Propositional formulas:

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

Subformula

Propositional formulas:

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

Quantified boolean formulas:

- The formula F_1 is the immediate subformula of the formulas $\forall p F_1$ and $\exists p F_1$.

Positions and Polarity

Let $F|_\pi = G$.
Propositional formulas:

- If G has the form $G_1 \wedge \ldots \wedge G_n$ or $G_1 \vee \ldots \vee G_n$, then for all $i \in \{1, \ldots, n\}$ the position $\pi.i$ is a position in F and $pol(F, \pi.i) \stackrel{\text{def}}{=} pol(F, \pi)$.
- If G has the form $\neg G_1$, then $\pi.1$ is a position in F, $F|_{\pi.1} \stackrel{\text{def}}{=} G_1$ and $pol(F, \pi.1) \stackrel{\text{def}}{=} -pol(F, \pi)$.
- $\ldots$

Positions and Polarity

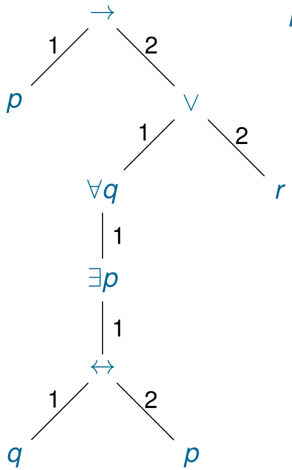
Let $F|_\pi = G$.
Propositional formulas:

- If G has the form $G_1 \wedge \ldots \wedge G_n$ or $G_1 \vee \ldots \vee G_n$, then for all $i \in \{1, \ldots, n\}$ the position $\pi.i$ is a position in F and $pol(F, \pi.i) \stackrel{\text{def}}{=} pol(F, \pi)$.
- If G has the form $\neg G_1$, then $\pi.1$ is a position in F, $F|_{\pi.1} \stackrel{\text{def}}{=} G_1$ and $pol(F, \pi.1) \stackrel{\text{def}}{=} -pol(F, \pi)$.
- ...

Quantified boolean formulas:

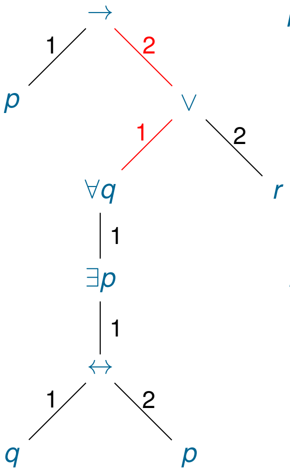
- If G has the form $\forall p G_1$ or $\exists p G_1$, then $\pi.1$ is a position in F, $F|_{\pi.1} \stackrel{\text{def}}{=} G_1$ and $pol(F, \pi.1) \stackrel{\text{def}}{=} pol(F, \pi)$.

Example



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

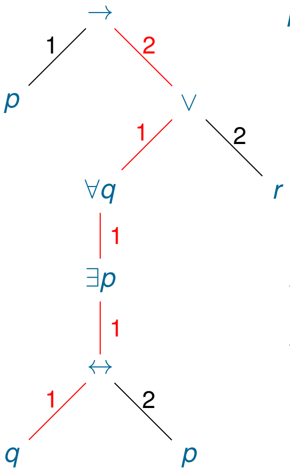
Example



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

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

Example



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

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

$F|_{2.1.1.1.1} = q$

Free and bound occurrences of variables

Let p be a boolean variable and $F|_\pi = p$.

- The occurrence of p at the 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 $\forall pG$ or $\exists pG$ for some G.

Free and bound occurrences of variables

Let p be a boolean variable and $F|_\pi = p$.

- The occurrence of p at the 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 $\forall pG$ or $\exists pG$ for some G.
  In other words, a bound occurrence of p is an occurrence in the scope of $\forall p$ or $\exists p$.

Free and bound occurrences of variables

Let p be a boolean variable and $F|_\pi = p$.

- The occurrence of p at the 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 $\forall pG$ or $\exists pG$ for some G.
  In other words, 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 p be a boolean variable and $F|_\pi = p$.

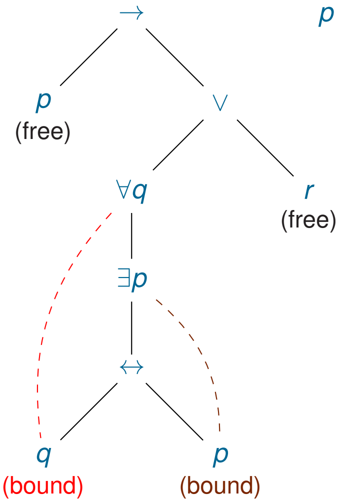
- The occurrence of p at the position π in F is bound if π can be represented as a concatenation of two strings $\pi_1\pi_2$ such that $F|_{\pi_1}$ has the form $\forall pG$ or $\exists pG$ for some G.
  In other words, a bound occurrence of p is an occurrence in the scope of $\forall p$ or $\exists p$.
- 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 p be a boolean variable and $F|_\pi = p$.

- The occurrence of p at the 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 $\forall pG$ or $\exists pG$ for some G.
  In other words, 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.

Example: Free and Bound Variables



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

Only Free Variables Matter

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

Lemma

Let for all free variables p of a formula F we have $I_1(p) = I_2(p)$. Then $I_1 \models F$ if and only if $I_2 \models F$.

Only Free Variables Matter

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

Lemma

Let for all free variables p of a formula F we have $I_1(p) = I_2(p)$. Then $I_1 \models F$ if and only if $I_2 \models F$.

Theorem

Let F be a closed formula and I_1, I_2 be interpretations. Then $I_1 \models F$ if and only if $I_2 \models F$.

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 notions for closed formulas:

Lemma

For every interpretation I and closed formula F the following propositions are equivalent: (i) $I \models F$; (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 notions for closed formulas:

Lemma

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

Validity and satisfiability can be expressed through truth:

Lemma

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

- *F is satisfiable if and only if the formula $\exists p_1 \ldots \exists p_n F$ is satisfiable (true, valid).*
- *F is valid if and only if the formula $\forall p_1 \ldots \forall p_n F$ is valid (true, satisfiable).*

More on free and bound occurrences

```
int symdiff(int i,int j)
{
  return i > j ?  i - j :  j - i;
}

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

More on free and bound occurrences

binding

```
int symdiff(int i,int j)
{
  return i > j ?  i - j :  j - i;
}
```

bound

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

free

More on free and bound occurrences

```
int symdiff(int i,int j)
{
  return i > j ?  i - j :  j - i;
}

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

Renaming bound variables does not change the semantics of the program:

```
int symdiff(int k,int j)
{
  return k > j ?  k - j :  j - k;
}

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

Substitutions for propositional formulas

Substitution: $(F)_p^G$: denotes the formula obtained from F by replacing all occurrences of the variable p by G.

Substitutions for propositional formulas

Substitution: $(F)_p^G$: denotes the formula obtained from F by replacing all occurrences of the variable p by G.

Example:

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

Substitutions for propositional formulas

Substitution: $(F)_p^G$: denotes the formula obtained from F by replacing all occurrences of the variable p by G.

Example:

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

Properties: If we apply any substitution to a valid formula then we also obtain a valid formula.

Substitution for quantified formulas

Some problems...

Substitution for quantified formulas

Some problems...

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

Substitution for quantified formulas

Some problems...

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

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

Substitution for quantified formulas

Some problems...

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

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

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. $(\forall p p \rightarrow q) \wedge (q \vee (q \rightarrow p))$

Renaming bound variables

Notation: $\exists\!\!\!\forall$: any of $\exists$, $\forall$ and $\wedge\!\!\!\!\vee$: any of $\wedge$, $\vee$.

Renaming bound variables

Notation: $\exists\!\forall$: any of $\exists$, $\forall$ and $\wedge\!\!\vee$: any of $\wedge$, $\vee$.

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

1. Take a fresh variable q (that is a variable not occurring in F);
2. Replace all free occurrences of p in G (note: not in F!) by q obtaining G'.
3. So we obtain the $F[\ \exists\!\forall qG'\ ]$ as the result.

Renaming bound variables

Notation: $\exists\forall$: any of $\exists$, $\forall$ and $\wedge\vee$: any of $\wedge$, $\vee$.

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

1. Take a fresh variable q (that is a variable not occurring in F);
2. Replace all free occurrences of p in G (note: not in F!) by q obtaining G'.
3. So we obtain the $F[\ \exists\forall qG'\ ]$ as the result.

Lemma

$F[\ \exists\forall pG\ ] \equiv F[\ \exists\forall qG'\ ]$

Renaming bound variables

Notation: $\exists\!\!\!\forall$: any of $\exists$, $\forall$ and $\wedge\!\!\!\vee$: any of $\wedge$, $\vee$.

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

1. Take a fresh variable q (that is a variable not occurring in F);
2. Replace all free occurrences of p in G (note: not in F!) by q obtaining G'.
3. So we obtain the $F[\ \exists\!\!\!\forall qG'\ ]$ as the result.

Lemma

$F[\ \exists\!\!\!\forall pG\ ] \equiv F[\ \exists\!\!\!\forall qG'\ ]$

Example:
$\exists q(\ \forall p((p \rightarrow q) \wedge p)\ ) \vee p$.

Then we can rename p into r obtaining
$\exists q(\ \forall r((r \rightarrow q) \wedge r)\ ) \vee p$.

Rectified formulas

Rectified formula F:

- no variable appears both free and bound in F;
- for every variable p, the formula F contains at most one occurrence of quantifiers $\exists\forall p$.

Rectified formulas

Rectified formula F:

- no variable appears both free and bound in F;
- for every variable p, the formula F contains at most one occurrence of quantifiers $\exists\forall p$.

Any formula can be transformed into a rectified formula by renaming bound variables.

Rectified formulas

Rectified formula F:

- no variable appears both free and bound in F;
- for every variable p, the formula F contains at most one occurrence of quantifiers $\exists\forall p$.

Any formula can be transformed into a rectified formula by renaming bound variables.

We can use the usual notation $(F)_p^G$ for rectified formulas assuming that p occurs only free.

Rectification: Example

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

Rectification: Example

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

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

Rectification: Example

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

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

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

This formula is rectified and equivalent to the original one.

Another problem

$\exists q(\neg p \leftrightarrow q)$: there exists a truth value equal to the value of $\neg p$. This formula is valid.

Rename p into q.

Another problem

$\exists q(\neg p \leftrightarrow q)$: there exists a truth value equal to the value of $\neg p$. This formula is valid.

Rename p into q.

$\exists q(\neg q \leftrightarrow q)$: there exists a truth value equivalent to its own negation. This formula is unsatisfiable.

Another restriction

Suppose we want to substitute $(F)_p^G$.
Then we require: no free variable in G become bound in $(F)_p^G$.

Another restriction

Suppose we want to substitute $(F)_p^G$.
Then we require: no free variable in G become bound in $(F)_p^G$.

In previous example $\exists q(\neg p \leftrightarrow q)$:
Substitute p by q. ($\exists q(\neg q \leftrightarrow q)$ does not satisfy above)

Another restriction

Suppose we want to substitute $(F)_p^G$.
Then we require: no free variable in G become bound in $(F)_p^G$.

In previous example $\exists q(\neg p \leftrightarrow q)$:
Substitute p by q. ($\exists q(\neg q \leftrightarrow q)$ does not satisfy above)

Uniform solution – renaming of bound variables
$\exists q(\neg p \leftrightarrow q) \equiv \exists r(\neg p \leftrightarrow r)$
Now we can substitute p by q obtaining $\exists r(\neg q \leftrightarrow r)$

Another restriction

Suppose we want to substitute $(F)^G_p$.
Then we require: no free variable in G become bound in $(F)^G_p$.

In previous example $\exists q(\neg p \leftrightarrow q)$:
Substitute p by q. ($\exists q(\neg q \leftrightarrow q)$ does not satisfy above)

Uniform solution – renaming of bound variables
$\exists q(\neg p \leftrightarrow q) \equiv \exists r(\neg p \leftrightarrow r)$
Now we can substitute p by q obtaining $\exists r(\neg q \leftrightarrow r)$

From now on we always assume that:

- formulas are rectified.
- all substitutions satisfy the requirement above

Equivalent replacement

Lemma

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

Theorem (Equivalent Replacement)

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

Prenex form

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

Prenex form

- Quantifier-free formula: no quantifiers (that is, propositional).
- Prenex formula has the form $\exists\!\!\forall_1 p_1 \ldots \exists\!\!\forall_n p_n G$, where G is quantifier-free.

Prenex form

- Quantifier-free formula: no quantifiers (that is, propositional).
- Prenex formula has the form $\exists\!\forall_1 p_1 \ldots \exists\!\forall_n p_n G$, where G is quantifier-free.
- Outermost prefix of $\exists\!\forall_1 p_1 \ldots \exists\!\forall_n p_n G$: the longest subsequence $\exists\!\forall_1 p_1 \ldots \exists\!\forall_k p_k$ of $\exists\!\forall_1 p_1 \ldots \exists\!\forall_n p_n$ such that $\exists\!\forall_1 = \ldots = \exists\!\forall_k$.

Prenex form

- Quantifier-free formula: no quantifiers (that is, propositional).
- Prenex formula has the form $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_n p_n G$, where G is quantifier-free.
- Outermost prefix of $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_n p_n G$: the longest subsequence $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_k p_k$ of $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_n p_n$ such that $\exists\!\!\!\forall_1 = \ldots = \exists\!\!\!\forall_k$.
- A formula F is a prenex form of a formula G if F is prenex and $F \equiv G$.

Prenexing rules

Prenexing rules:

$$\rlap{$\exists$}\forall p F_1 \mathbin{\rlap{$\wedge$}\vee} \ldots \mathbin{\rlap{$\wedge$}\vee} F_n \;\Rightarrow\; \rlap{$\exists$}\forall p (F_1 \mathbin{\rlap{$\wedge$}\vee} \ldots \mathbin{\rlap{$\wedge$}\vee} F_n)$$

$$\begin{array}{llll}
\forall p F_1 \to F_2 \;\Rightarrow\; \exists p (F_1 \to F_2) & & \exists p F_1 \to F_2 \;\Rightarrow\; \forall p (F_1 \to F_2) \\
F_1 \to \forall p F_2 \;\Rightarrow\; \forall p (F_1 \to F_2) & & F_1 \to \exists p F_2 \;\Rightarrow\; \exists p (F_1 \to F_2)
\end{array}$$

$$\neg \forall p F \;\Rightarrow\; \exists p \neg F \qquad\qquad \neg \exists p F \;\Rightarrow\; \forall p \neg F$$

Prenexing. Example I

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

Prenexing. Example II

$$
\begin{array}{l}
\exists q(q \rightarrow p) \rightarrow \neg\forall r(r \rightarrow p) \vee p \Rightarrow \\
\exists q(q \rightarrow p) \rightarrow \exists r\neg(r \rightarrow p) \vee p \Rightarrow \\
\exists q(q \rightarrow p) \rightarrow \exists r(\neg(r \rightarrow p) \vee p) \Rightarrow \\
\exists r(\exists q(q \rightarrow p) \rightarrow \neg(r \rightarrow p) \vee p) \Rightarrow \\
\exists r\forall q((q \rightarrow p) \rightarrow \neg(r \rightarrow p) \vee p).
\end{array}
$$

What's next

Algorithms for satisfiability, validity of QBF:

- Splitting
- DPLL

Reminder:
(i) $F(p_1, \ldots, p_n)$ is satisfiable iff $\exists p_1 \ldots \exists p_n F(p_1, \ldots, p_n)$ is true/satisfiable.
(ii) $F(p_1, \ldots, p_n)$ is valid iff $\forall p_1 \ldots \forall p_n F(p_1, \ldots, p_n)$ is true/satisfiable.
Algorithms will check whether a closed formula is true or false.

Splitting: foundations

Lemma

- *A closed formula $\forall pF$ is true if and only if the formulas $F_p^\perp$ and $F_p^\top$ are true.*
- *A closed formula $\exists pF$ is true if and only if at least one of the formulas $F_p^\perp$ or $F_p^\top$ is true.*

Splitting

Simplification rules for $\top$:

$$\begin{array}{c} \neg\top \;\Rightarrow\; \bot \\ \top \wedge F_1 \wedge \ldots \wedge F_n \;\Rightarrow\; F_1 \wedge \ldots \wedge F_n \\ \top \vee F_1 \vee \ldots \vee F_n \;\Rightarrow\; \top \\ F \rightarrow \top \;\Rightarrow\; \top \qquad \top \rightarrow F \;\Rightarrow\; F \\ F \leftrightarrow \top \;\Rightarrow\; F \qquad \top \leftrightarrow F \;\Rightarrow\; F \end{array}$$

Simplification rules for $\bot$:

$$\begin{array}{c} \neg\bot \;\Rightarrow\; \top \\ \bot \wedge F_1 \wedge \ldots \wedge F_n \;\Rightarrow\; \bot \\ \bot \vee F_1 \vee \ldots \vee F_n \;\Rightarrow\; F_1 \vee \ldots \vee F_n \\ F \rightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \rightarrow F \;\Rightarrow\; \top \\ F \leftrightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \leftrightarrow F \;\Rightarrow\; \neg F \end{array}$$

Splitting

Simplification rules for $\top$:

$$
\begin{array}{c}
\neg\top \;\Rightarrow\; \bot \\
\top \wedge F_1 \wedge \ldots \wedge F_n \;\Rightarrow\; F_1 \wedge \ldots \wedge F_n \\
\top \vee F_1 \vee \ldots \vee F_n \;\Rightarrow\; \top \\
F \rightarrow \top \;\Rightarrow\; \top \qquad \top \rightarrow F \;\Rightarrow\; F \\
F \leftrightarrow \top \;\Rightarrow\; F \qquad \top \leftrightarrow F \;\Rightarrow\; F \\
\forall p \top \;\Rightarrow\; \top \\
\exists p \top \;\Rightarrow\; \top
\end{array}
$$

Simplification rules for $\bot$:

$$
\begin{array}{c}
\neg\bot \;\Rightarrow\; \top \\
\bot \wedge F_1 \wedge \ldots \wedge F_n \;\Rightarrow\; \bot \\
\bot \vee F_1 \vee \ldots \vee F_n \;\Rightarrow\; F_1 \vee \ldots \vee F_n \\
F \rightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \rightarrow F \;\Rightarrow\; \top \\
F \leftrightarrow \bot \;\Rightarrow\; \neg F \qquad \bot \leftrightarrow F \;\Rightarrow\; \neg F \\
\forall p \bot \;\Rightarrow\; \bot \\
\exists p \bot \;\Rightarrow\; \bot
\end{array}
$$

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 of F and a boolean value for it)
begin
$F \ := \ \mathit{simplify}(F)$
if $F = \bot$ **then return** 0
if $F = \top$ **then return** 1
Let F have the form $\exists\!\!\!\forall p_1 \ldots \exists\!\!\!\forall p_k F_1$
$(p, b) \ := \ \mathit{select\_variable\_value}(F)$
Let F' be obtained from F by deleting $\exists\!\!\!\forall p$ from its outermost prefix
if $b = 0$ **then** $(G_1, G_2) \ := \ (\bot, \top)$
else $(G_1, G_2) \ := \ (\top, \bot)$
case $(\mathit{splitting}((F')_p^{G_1}), \ \exists\!\!\!\forall)$ **of**
$(0, \forall) \Rightarrow$ **return** 0
$(0, \exists) \Rightarrow$ **return** $\mathit{splitting}((F')_p^{G_2})$
$(1, \forall) \Rightarrow$ **return** $\mathit{splitting}((F')_p^{G_2})$
$(1, \exists) \Rightarrow$ **return** 1
end

Splitting: examples

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

Splitting: examples

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

Splitting: examples

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

Splitting: examples

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

Splitting: examples

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

Splitting: examples

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

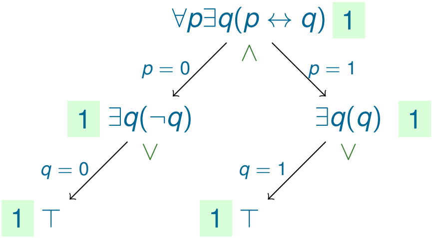
Splitting: examples

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

Splitting: examples

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

Splitting: examples



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

Splitting: examples

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

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

Splitting: examples

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

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

Splitting: examples

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

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

Splitting: examples

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

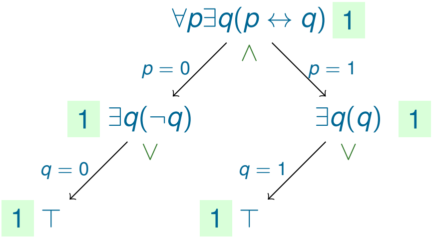
$$
\begin{tikzcd}[column sep=small]
& & \exists q \forall p (p \leftrightarrow q) \arrow[dl, "q=0"'] \arrow[dr, "q=1"] \arrow[d, phantom, "\vee" near start] & \\
& \boxed{0}\ \forall p(\neg p) \arrow[dl, "p=1"'] \arrow[d, phantom, "\wedge" near start] & & \forall p(p) \\
\boxed{0}\ \bot & & &
\end{tikzcd}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

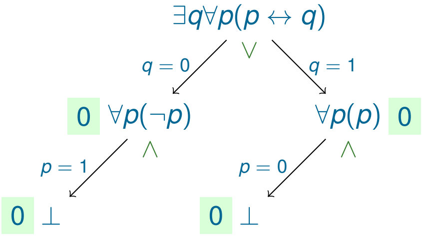
Splitting: examples

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

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

Splitting: examples





Splitting: examples

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

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

Splitting: examples

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

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

Note: selection of variable values is best understood as two-player games: 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 it false.

CNF

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

CNF

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

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

CNF

For more efficient algorithms we need formulas 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 CNF, if it is either $\bot$, or $\top$, or has the form $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_n p_n (C_1 \wedge \ldots \wedge C_m)$, where $C_1, \ldots, C_m$ are clauses.

CNF

For more efficient algorithms we need formulas 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 CNF, if it is either $\perp$, or $\top$, or has the form $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_n p_n (C_1 \wedge \ldots \wedge C_m)$, 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:

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

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\!\!\forall_1 p_1 \ldots\ldots \exists\!\!\!\forall_n p_n$
- S: a set of clauses

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\!\!\forall_1 p_1 \ldots\ldots \exists\!\!\!\forall_n p_n$
- S: a set of clauses

Main simplification – unit propagation with respect to Q, S:
if S contains a unit clause, i.e. a clause consisting of one literal L of the form p or $\neg p$ then

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\!\!\forall_1 p_1 \ldots\ldots \exists\!\!\!\forall_n p_n$
- S: a set of clauses

Main simplification – unit propagation with respect to Q, S:
if S contains a unit clause, i.e. a clause consisting of one literal L of the form p or $\neg p$ then

- if Q contains $\exists p$ or p does not occur in Q
  1. remove from S every clause of the form $L \vee C'$;
  2. replace in S every clause of the form $\overline{L} \vee C'$ by the clause C'.

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\!\!\forall_1 p_1 \ldots\ldots \exists\!\!\!\forall_n p_n$
- S: a set of clauses

Main simplification – unit propagation with respect to Q, S:
if S contains a unit clause, i.e. a clause consisting of one literal L of the form p or $\neg p$ then

- if Q contains $\exists p$ or p does not occur in Q
  1. remove from S every clause of the form $L \vee C'$;
  2. replace in S every clause of the form $\overline{L} \vee C'$ by the clause C'.
- if Q contains $\forall p$, then replace S by the set $\{\Box\}$;

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\!\!\forall_1 p_1 \ldots\ldots \exists\!\!\!\forall_n p_n$
- S: a set of clauses

Main simplification – unit propagation with respect to Q, S:
if S contains a unit clause, i.e. a clause consisting of one literal L of the form p or $\neg p$ then

- if Q contains $\exists p$ or p does not occur in Q
  1. remove from S every clause of the form $L \vee C'$;
  2. replace in S every clause of the form $\overline{L} \vee C'$ by the clause C'.
- if Q contains $\forall p$, then replace S by the set $\{\square\}$;

Why different for universal quantifiers? Use intuition from games!

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\forall_1 p_1 \ldots \ldots \exists\!\forall_n p_n$
- S: a set of clauses

Main simplification – unit propagation with respect to Q, S:
if S contains a unit clause, i.e. a clause consisting of one literal L of the form p or $\neg p$ then

- if Q contains $\exists p$ or p does not occur in Q
  1. remove from S every clause of the form $L \vee C'$;
  2. replace in S every clause of the form $\overline{L} \vee C'$ by the clause C'.
- if Q contains $\forall p$, then replace S by the set $\{\square\}$;

Why different for universal quantifiers? Use intuition from games!

The player playing $\forall$ wants to make the formula false.

Unit Propagation (DPLL)

Input of DPLL:

- Q: quantifier sequence $\exists\!\!\forall_1 p_1 \ldots\ldots \exists\!\!\forall_n p_n$
- S: a set of clauses

Main simplification – unit propagation with respect to Q, S:
if S contains a unit clause, i.e. a clause consisting of one literal L of the form p or $\neg p$ then

- if Q contains $\exists p$ or p does not occur in Q
  1. remove from S every clause of the form $L \lor C'$;
  2. replace in S every clause of the form $\overline{L} \lor C'$ by the clause C'.
- if Q contains $\forall p$, then replace S by the set $\{\Box\}$;

Why different for universal quantifiers? Use intuition from games!

The player playing $\forall$ wants to make the formula false. So, when it is his turn to make a move $\forall p$, he has a winning move: to select the value for p which makes the unit clause false (and hence the conjunction of clauses false too).

DPLL algorithm

procedure $DPLL(Q, S)$
input: quantifier sequence $Q = \exists\!\forall_1 p_1 \ldots \exists\!\forall_n p_n$, set of clauses S
output: 0 or 1
parameters: function $select\_variable\_value$
begin
  $S \; := \; unit\_propagate(Q, S)$
  if S is empty **then** **return** 1
  if S contains $\square$ **then** **return** 0
  $(p, b) \; := \; select\_variable\_value(Q, S)$
  Let Q' be obtained from Q by deleting $\exists\!\forall p$ from its outermost prefix
  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** $DPLL(Q', S \cup \{\overline{L}\})$
    $(1, \forall) \Rightarrow$ **return** $DPLL(Q', S \cup \{\overline{L}\})$
    $(1, \exists) \Rightarrow$ **return** 1
end

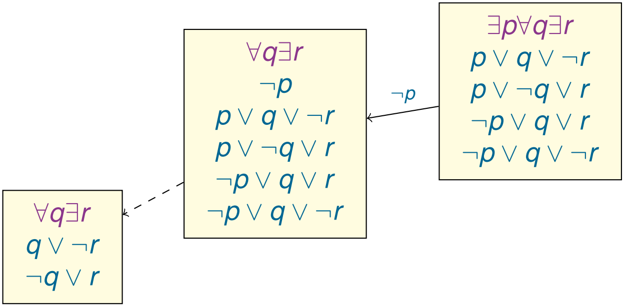
Example

$$\exists p \forall q \exists r$$
$$p \vee q \vee \neg r$$
$$p \vee \neg q \vee r$$
$$\neg p \vee q \vee r$$
$$\neg p \vee q \vee \neg r$$

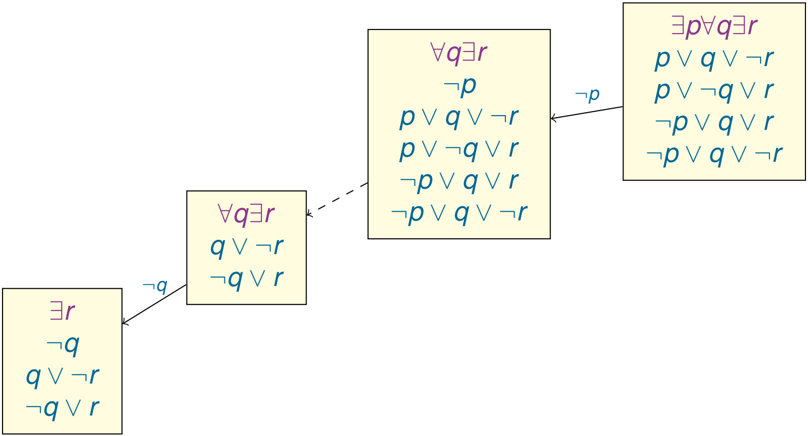
Example

$$
\boxed{\begin{array}{c} \forall q \exists r \\ \neg p \\ p \vee q \vee \neg r \\ p \vee \neg q \vee r \\ \neg p \vee q \vee r \\ \neg p \vee q \vee \neg r \end{array}}
\xleftarrow{\neg p}
\boxed{\begin{array}{c} \exists p \forall q \exists r \\ p \vee q \vee \neg r \\ p \vee \neg q \vee r \\ \neg p \vee q \vee r \\ \neg p \vee q \vee \neg r \end{array}}
$$

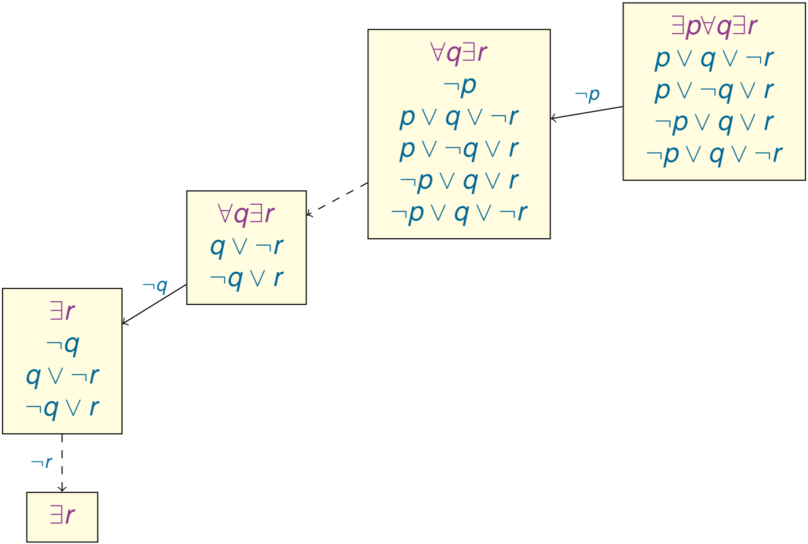
Example


$\exists p \forall q \exists r$
$p \lor q \lor \neg r$
$p \lor \neg q \lor r$
$\neg p \lor q \lor r$
$\neg p \lor q \lor \neg r$
$\neg p$
$\forall q \exists r$
$\neg p$
$p \lor q \lor \neg r$
$p \lor \neg q \lor r$
$\neg p \lor q \lor r$
$\neg p \lor q \lor \neg r$
$\forall q \exists r$
$q \lor \neg r$
$\neg q \lor r$

Example

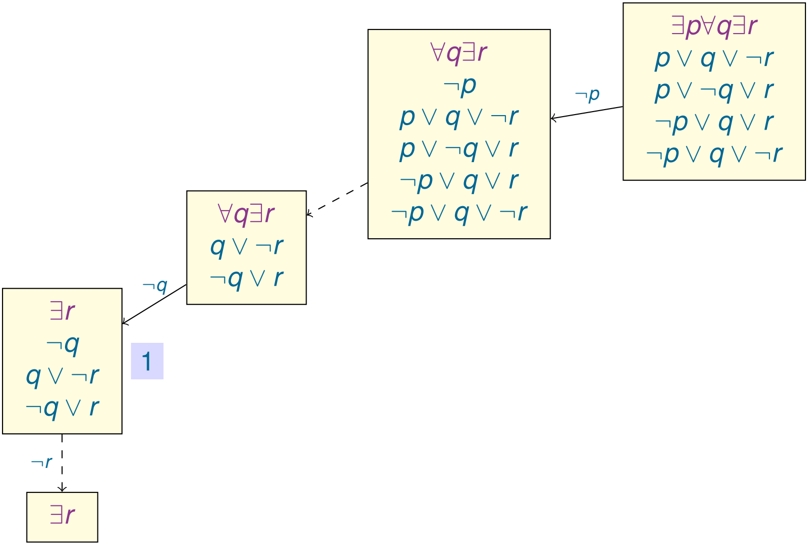

$\exists p \forall q \exists r$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\neg p$
$\forall q \exists r$
$\neg p$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\forall q \exists r$
$q \vee \neg r$
$\neg q \vee r$
$\neg q$
$\exists r$
$\neg q$
$q \vee \neg r$
$\neg q \vee r$

Example

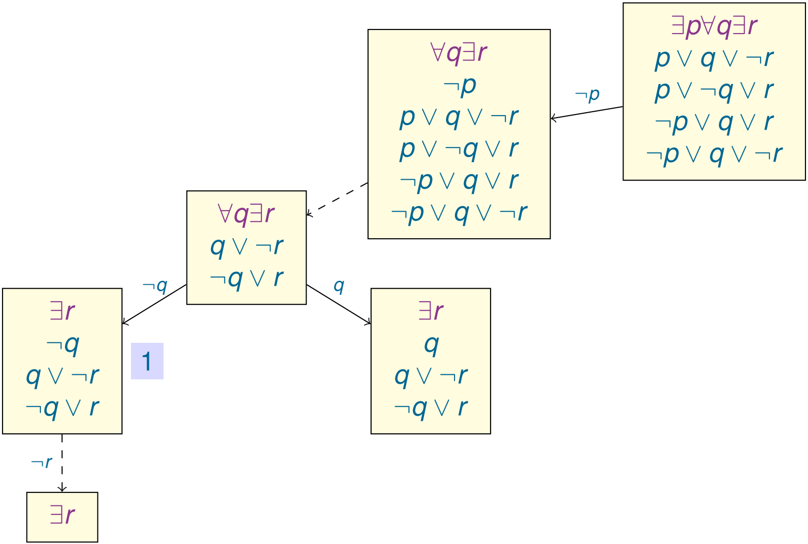


$\exists p \forall q \exists r$
$p \lor q \lor \neg r$
$p \lor \neg q \lor r$
$\neg p \lor q \lor r$
$\neg p \lor q \lor \neg r$
$\neg p$
$\forall q \exists r$
$\neg p$
$p \lor q \lor \neg r$
$p \lor \neg q \lor r$
$\neg p \lor q \lor r$
$\neg p \lor q \lor \neg r$
$\forall q \exists r$
$q \lor \neg r$
$\neg q \lor r$
$\neg q$
$\exists r$
$\neg q$
$q \lor \neg r$
$\neg q \lor r$
$\neg r$
$\exists r$

Example

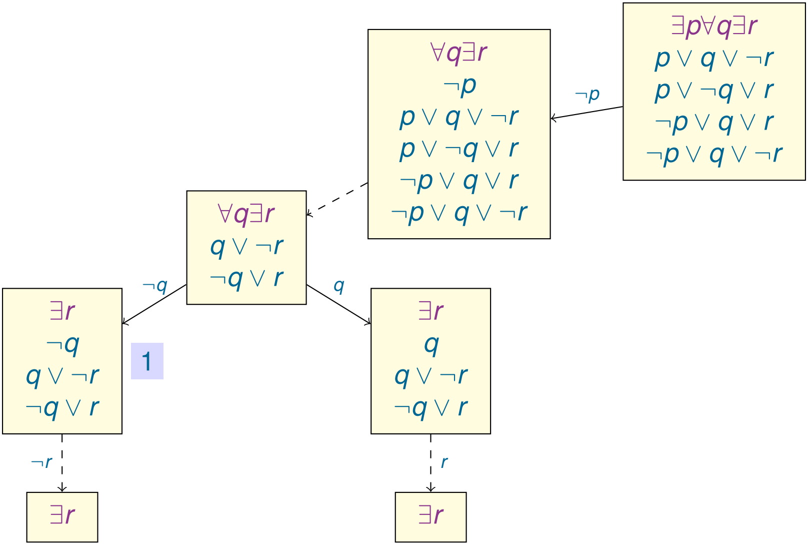


Example

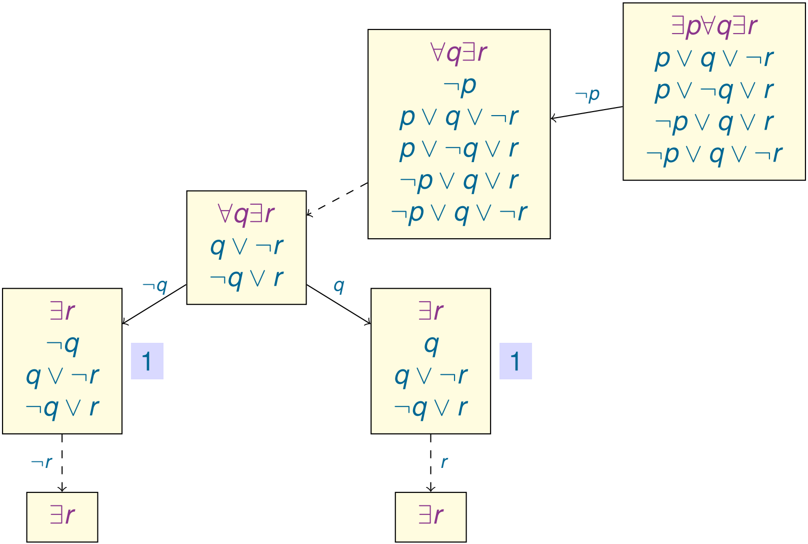


$\exists p \forall q \exists r$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\neg p$
$\forall q \exists r$
$\neg p$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\forall q \exists r$
$q \vee \neg r$
$\neg q \vee r$
$\neg q$
$\exists r$
$\neg q$
$q \vee \neg r$
$\neg q \vee r$
1
q
$\exists r$
q
$q \vee \neg r$
$\neg q \vee r$
$\neg r$
$\exists r$

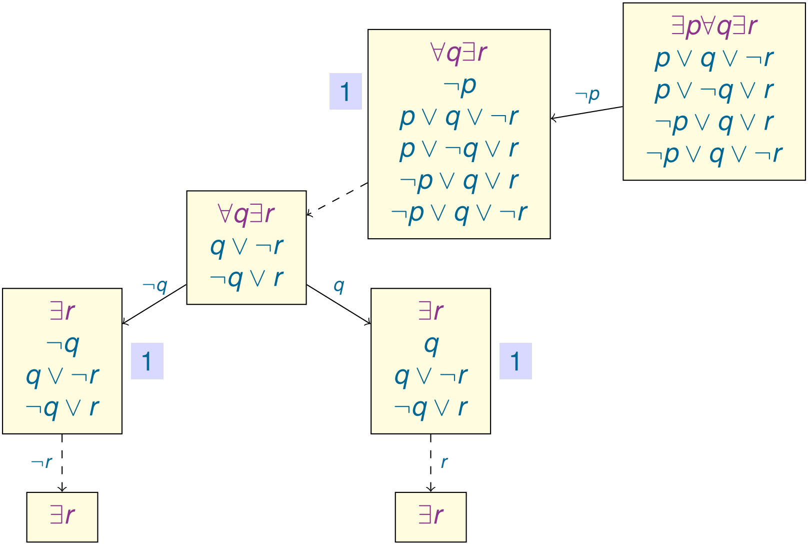
Example


$\exists p \forall q \exists r$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\neg p$
$\forall q \exists r$
$\neg p$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\forall q \exists r$
$q \vee \neg r$
$\neg q \vee r$
$\neg q$
q
$\exists r$
$\neg q$
$q \vee \neg r$
$\neg q \vee r$
1
$\exists r$
q
$q \vee \neg r$
$\neg q \vee r$
$\neg r$
r
$\exists r$
$\exists r$

Example

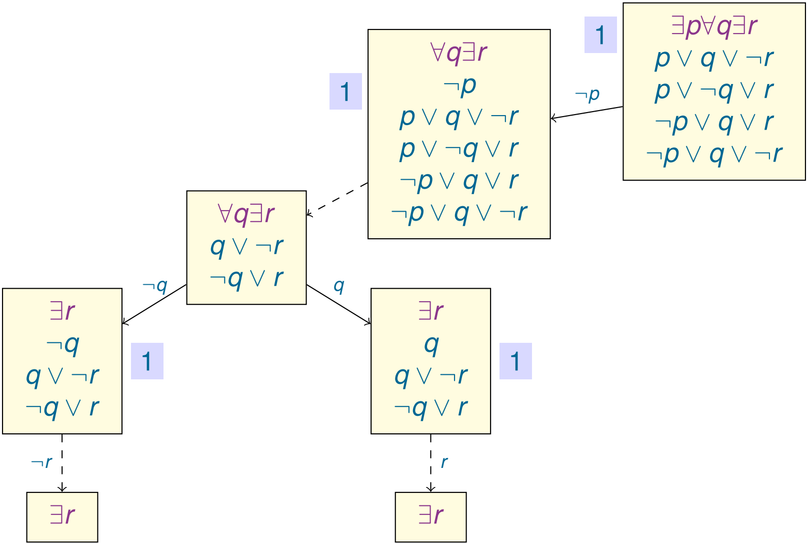

$\exists p \forall q \exists r$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\neg p$
$\forall q \exists r$
$\neg p$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\forall q \exists r$
$q \vee \neg r$
$\neg q \vee r$
$\neg q$
q
$\exists r$
$\neg q$
$q \vee \neg r$
$\neg q \vee r$
1
$\exists r$
q
$q \vee \neg r$
$\neg q \vee r$
1
$\neg r$
r
$\exists r$
$\exists r$

Example



$\exists p \forall q \exists r$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\neg p$
1
$\forall q \exists r$
$\neg p$
$p \vee q \vee \neg r$
$p \vee \neg q \vee r$
$\neg p \vee q \vee r$
$\neg p \vee q \vee \neg r$
$\forall q \exists r$
$q \vee \neg r$
$\neg q \vee r$
$\neg q$
q
$\exists r$
$\neg q$
$q \vee \neg r$
$\neg q \vee r$
1
$\exists r$
q
$q \vee \neg r$
$\neg q \vee r$
1
$\neg r$
r
$\exists r$
$\exists r$

Example



∃p∀q∃r
p ∨ q ∨ ¬r
p ∨ ¬q ∨ r
¬p ∨ q ∨ r
¬p ∨ q ∨ ¬r
1
¬p
∀q∃r
¬p
p ∨ q ∨ ¬r
p ∨ ¬q ∨ r
¬p ∨ q ∨ r
¬p ∨ q ∨ ¬r
1
∀q∃r
q ∨ ¬r
¬q ∨ r
¬q
q
∃r
¬q
q ∨ ¬r
¬q ∨ r
1
∃r
q
q ∨ ¬r
¬q ∨ r
1
¬r
r
∃r
∃r

Pure literal rule

Let Q be quantifier prefix and S set of clauses.
Let literal L be pure in S (i.e. $\overline{L}$ does not occur in S) then:

- If the variable of L is existentially quantified in Q then we can remove all clauses in which L occurs.

Pure literal rule

Let Q be quantifier prefix and S set of clauses.
Let literal L be pure in S (i.e. $\overline{L}$ does not occur in S) then:

- If the variable of L is existentially quantified in Q then we can remove all clauses in which L occurs.
- If the variable of L is universally quantified then we can remove L from all clauses where L occurs.

Pure literal rule

Let Q be quantifier prefix and S set of clauses.
Let literal L be pure in S (i.e. $\overline{L}$ does not occur in S) then:

- If the variable of L is existentially quantified in Q then we can remove all clauses in which L occurs.
- If the variable of L is universally quantified then we can remove L from all clauses where L occurs.

Why?

Pure literal rule

Let Q be quantifier prefix and S set of clauses.
Let literal L be pure in S (i.e. $\overline{L}$ does not occur in S) then:

- If the variable of L is existentially quantified in Q then we can remove all clauses in which L occurs.
- If the variable of L is universally quantified then we can remove L from all clauses where L occurs.

Why?

- The $\exists$-player will make the literal true (so all clauses containing this literal will be satisfied).

Pure literal rule

Let Q be quantifier prefix and S set of clauses.
Let literal L be pure in S (i.e. $\overline{L}$ does not occur in S) then:

- If the variable of L is existentially quantified in Q then we can remove all clauses in which L occurs.
- If the variable of L is universally quantified then we can remove L from all clauses where L occurs.

Why?

- The $\exists$-player will make the literal true (so all clauses containing this literal will be satisfied).
- The $\forall$-player will make the literal false (so it can be removed from all clauses containing this literal).

Universal literal deletion

Consider a quantifier prefix Q and a conjunction of clauses S.

- a variable p is existential in Q, if Q contains $\exists p$.
- a variable q is universal in Q, if Q contains $\forall q$.

Universal literal deletion

Consider a quantifier prefix Q and a conjunction of clauses S.

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

Universal literal deletion

Consider a quantifier prefix Q and a conjunction of clauses S.

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

Example: If Q is $\forall q \exists p \forall r$ then q is quantified before both p and r; and p is quantified before r (in Q).

Universal literal deletion

Consider a quantifier prefix Q and a conjunction of clauses S.

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

Example: If Q is $\forall q \exists p \forall r$ then q is quantified before both p and r; and p is quantified before r (in Q).

Theorem

Let Q be a quantifier prefix and S a conjunction of clauses. Suppose that

1. *C is a clause in S;*
2. *a variable q in C is universal in Q;*
3. *all existential variables in C are quantified before q.*

Then the deletion of the literal containing q from C does not change the truth value of QS.

Universal literal deletion

Let $q_1, \ldots, q_m$ be all universal variables of C such that all existential variables are quantified before them. Then C has the form:

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

Universal literal deletion

Let $q_1, \ldots, q_m$ be all universal variables of C such that all existential variables are quantified before them. Then C has the form:

$$L_1 \lor \ldots \lor L_n \lor (\neg)q_1 \lor \ldots \lor (\neg)q_m$$

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

Universal literal deletion

Let $q_1, \ldots, q_m$ be all universal variables of C such that all existential variables are quantified before them. Then C has the form:

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

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

- If at least one of the literals $L_1, \ldots, L_n$ is true, deletion of $(\neg) q_1, \ldots, (\neg) q_m$ will not change the outcome of the game, since after any assignment to $q_1, \ldots, q_m$ the clause will be true.

Universal literal deletion

Let $q_1, \ldots, q_m$ be all universal variables of C such that all existential variables are quantified before them. Then C has the form:

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

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

- If at least one of the literals $L_1, \ldots, L_n$ is true, deletion of $(\neg)q_1, \ldots, (\neg)q_m$ will not change the outcome of the game, since after any assignment to $q_1, \ldots, q_m$ the clause will be true.
- If all of the literals $L_1, \ldots, L_n$ are false, the $\forall$-player will make all $(\neg)q_1, \ldots, (\neg)q_m$ false and win the game, so deletion of these literals will not change the outcome of the game either.

Example

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

Example

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

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

Example

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

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

Example

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

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

Example

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

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

Example

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

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

Example

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

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

Example

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

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

Example

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

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

Example

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

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

End of Lecture 15

Slides for lecture 15 end here . . .

QBF and OBDD

We know how to apply boolean operations to OBDDs. Can we also apply quantification to OBDDs in a straightforward way?

QBF and OBDD

We know how to apply boolean operations to OBDDs. Can we also apply quantification to OBDDs in a straightforward way?

Quantification: given an OBDD representing a formula F, find an OBDD representing $\exists\!\!\forall_1 p_1 \ldots \exists\!\!\forall_n p_n F$

QBF and OBDD

We know how to apply boolean operations to OBDDs. Can we also apply quantification to OBDDs in a straightforward way?

Quantification: given an OBDD representing a formula F, find an OBDD representing $\exists\!\!\!\forall_1 p_1 \ldots \exists\!\!\!\forall_n p_n F$

There is no simple algorithm for quantification in general, but there is one when $\exists\!\!\!\forall_1 \ldots \exists\!\!\!\forall_n$ are the same quantifier.

Quantification for OBDDs

We can use the following properties of QBFs:

- $\exists p\,(\textit{ if } p \textit{ then } F \textit{ else } G) \equiv F \lor G$;
- $\forall p\,(\textit{ if } p \textit{ then } F \textit{ else } G) \equiv F \land G$;

Quantification for OBDDs

We can use the following properties of QBFs:

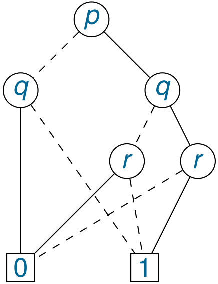
- $\exists p\,(\textit{ if } p \textit{ then } F \textit{ else } G) \equiv F \vee G$;
- $\forall p\,(\textit{ if } p \textit{ then } F \textit{ else } G) \equiv F \wedge G$;
- If $p \neq q$, then
  $\exists\!\!\!\forall p\,(\textit{ if } q \textit{ then } F \textit{ else } G) \equiv \textit{ if } q \textit{ then } \exists\!\!\!\forall pF \textit{ else } \exists\!\!\!\forall pG$

$\exists$-quantification algorithm for OBDDs

procedure $\exists quant(\{p_1, \ldots, p_k\}, \{n_1, \ldots, n_m\})$
parameters: global dag D
input: nodes $n_1, \ldots, n_m$ representing $F_1, \ldots, F_m$ in D
output: a node n representing $\exists p_1 \ldots \exists p_k (F_1 \vee \ldots \vee F_m)$ in (modified) D
begin
  if $m = 0$ **then return** $\boxed{0}$
  if some n_i is $\boxed{1}$ **then return** $\boxed{1}$
  if some n_i is $\boxed{0}$ **then**
    return $\exists quant(\{p_1, \ldots, p_k\}, \{n_1, \ldots, n_{i-1}, n_{i+1}, \ldots, n_m\})$
  $p \ := \ max\_var(n_1, \ldots, n_m)$
  forall $i = 1 \ldots m$
    if n_i is labelled by p
      then $(l_i, r_i) \ := \ (neg(n_i), pos(n_i))$
      else $(l_i, r_i) \ := \ (n_i, n_i)$
  if $p \in \{p_1, \ldots, p_k\}$
    then return $\exists quant(\{p_1, \ldots, p_k\} - \{p\}, \{l_1, \ldots, l_m, r_1, \ldots, r_m\})$
    else
      $k_1 \ := \ \exists quant(\{p_1, \ldots, p_k\}, \{l_1, \ldots, l_m\})$
      $k_2 \ := \ \exists quant(\{p_1, \ldots, p_k\}, \{r_1, \ldots, r_m\})$
      return $integrate(k_1, p, k_2, D)$
end

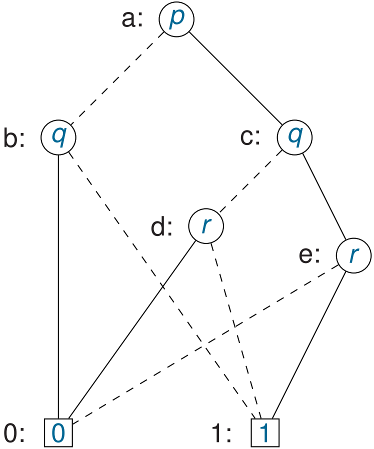
Example

Take the order $p > q > r$ and the formula $\exists p \exists r (p \leftrightarrow ((p \rightarrow r) \leftrightarrow q))$.



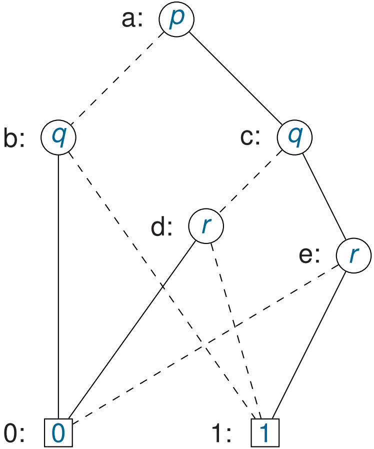
Example

$\exists quant(\{p, r\}, \{a\})$

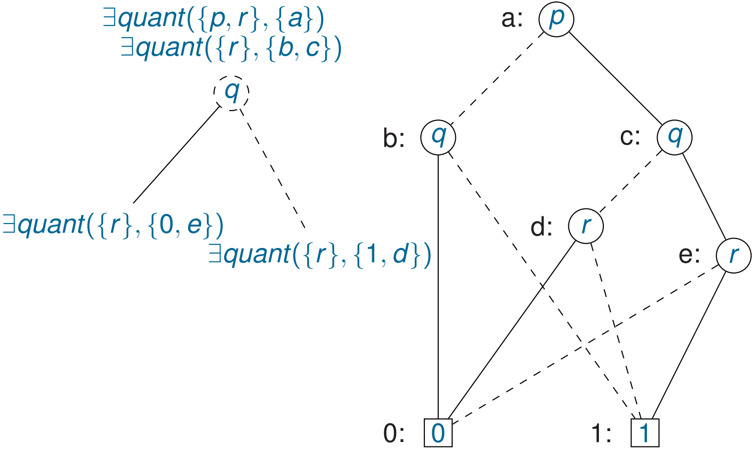


Example

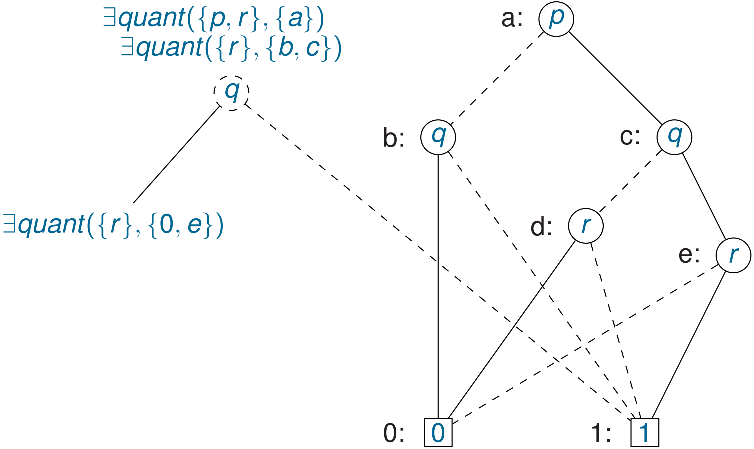
$\exists quant(\{p, r\}, \{a\})$
$\quad \exists quant(\{r\}, \{b, c\})$



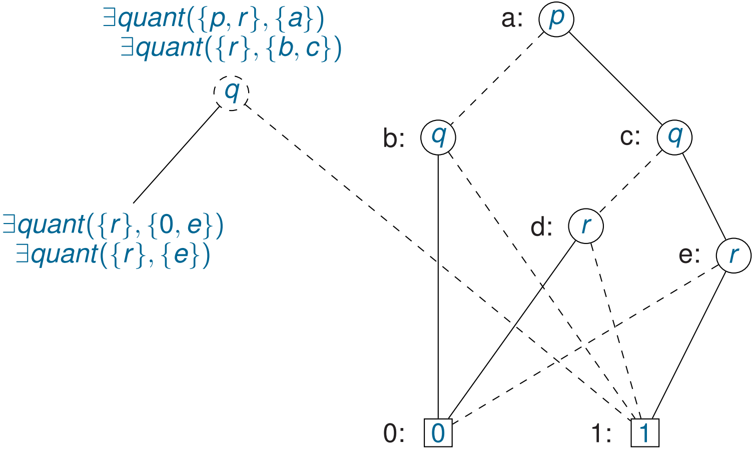
Example



Example

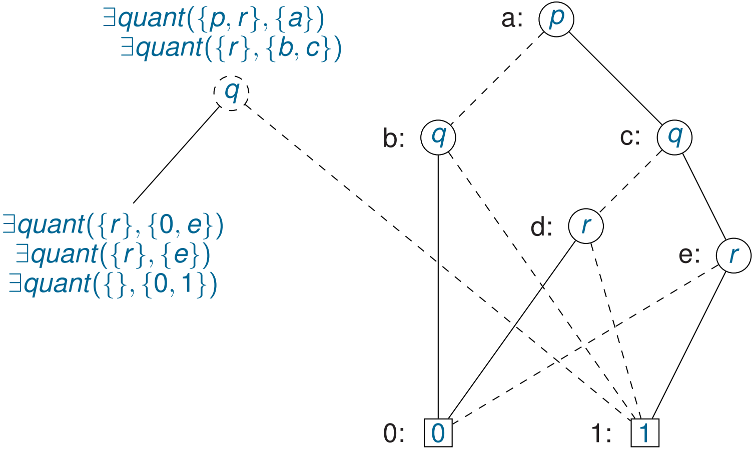

$\exists quant(\{p, r\}, \{a\})$
$\exists quant(\{r\}, \{b, c\})$
q
$\exists quant(\{r\}, \{0, e\})$
a: p
b: q
c: q
d: r
e: r
0: 0
1: 1

Example



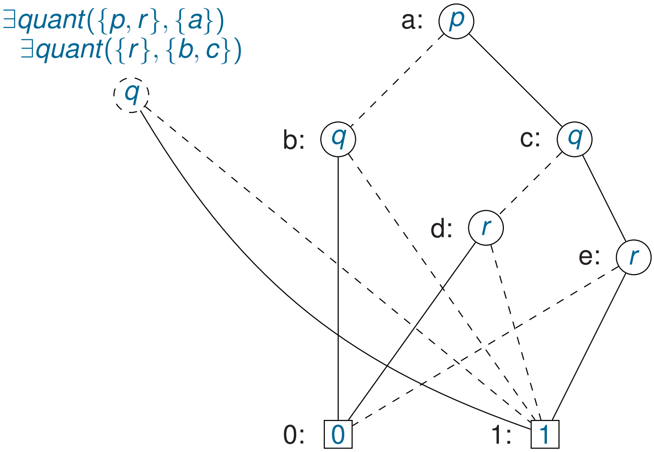
$\exists quant(\{p, r\}, \{a\})$
$\exists quant(\{r\}, \{b, c\})$
q
$\exists quant(\{r\}, \{0, e\})$
$\exists quant(\{r\}, \{e\})$
a: p
b: q
c: q
d: r
e: r
0: 0
1: 1

Example



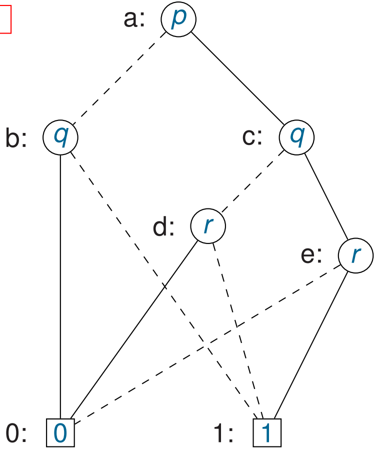
∃quant({p, r}, {a})
∃quant({r}, {b, c})
q
∃quant({r}, {0, e})
∃quant({r}, {e})
∃quant({}, {0, 1})
a: p
b: q
c: q
d: r
e: r
0: 0
1: 1

Example



Example

$\exists quant(\{p, r\}, \{a\}) = \boxed{1}$



$\exists$-quantification algorithm for OBDDs

procedure $\exists quant(\{p_1, \ldots, p_k\}, \{n_1, \ldots, n_m\})$
parameters: global dag D
input: nodes $n_1, \ldots, n_m$ representing $F_1, \ldots, F_m$ in D
output: a node n representing $\exists p_1 \ldots \exists p_k(F_1 \vee \ldots \vee F_m)$ in (modified) D
begin

```
  if m = 0 then return [0]
  if some n_i is [1] then return [1]
  if some n_i is [0] then
    return ∃quant({p_1, ..., p_k}, {n_1, ..., n_{i-1}, n_{i+1}, ..., n_m})
  p := max_var(n_1, ..., n_m)
  forall i = 1 ... m
    if n_i is labelled by p
      then (l_i, r_i) := (neg(n_i), pos(n_i))
      else (l_i, r_i) := (n_i, n_i)
  if p ∈ {p_1, ..., p_k}
    then return ∃quant({p_1, ..., p_k} − {p}, {l_1, ..., l_m, r_1, ..., r_m})
    else
      k_1 := ∃quant({p_1, ..., p_k}, {l_1, ..., l_m})
      k_2 := ∃quant({p_1, ..., p_k}, {r_1, ..., r_m})
      return integrate(k_1, p, k_2, D)
```

end

$\forall$-quantification algorithm for OBDDs

procedure $\forall quant(\{p_1, \ldots, p_k\}, \{n_1, \ldots, n_m\})$
parameters: global dag D
input: nodes $n_1, \ldots, n_m$ representing $F_1, \ldots, F_m$ in D
output: a node n representing $\forall p_1 \ldots \forall p_k (F_1 \wedge \ldots \wedge F_m)$ in (modified) D
begin
  if $m = 0$ **then return** $\boxed{1}$
  if some n_i is $\boxed{0}$ **then return** $\boxed{0}$
  if some n_i is $\boxed{1}$ **then**
    return $\forall quant(\{p_1, \ldots, p_k\}, \{n_1, \ldots, n_{i-1}, n_{i+1}, \ldots, n_m\})$
  $p \;:=\; max\_var(n_1, \ldots, n_m)$
  forall $i = 1 \ldots m$
    if n_i is labelled by p
      then $(l_i, r_i) \;:=\; (neg(n_i), pos(n_i))$
      else $(l_i, r_i) \;:=\; (n_i, n_i)$
  if $p \in \{p_1, \ldots, p_k\}$
    then return $\forall quant(\{p_1, \ldots, p_k\} - \{p\}, \{l_1, \ldots, l_m, r_1, \ldots, r_m\})$
    else
      $k_1 \;:=\; \forall quant(\{p_1, \ldots, p_k\}, \{l_1, \ldots, l_m\})$
      $k_2 \;:=\; \forall quant(\{p_1, \ldots, p_k\}, \{r_1, \ldots, r_m\})$
      return $integrate(k_1, p, k_2, D)$
end