形式化方法导引

第 4 章 逻辑问题求解

4.2 理论 - (3) CNF 与 Horn Clauses

黄文超

http://staff.ustc.edu.cn/~huangwc/fm.html

回顾：SAT 求解所遇到的问题:
Provable equivalence:

$$\neg(p \wedge q) \dashv\vdash \neg q \vee \neg p \qquad \neg(p \vee q) \dashv\vdash \neg q \wedge \neg p$$
$$p \to q \dashv\vdash \neg q \to \neg p \qquad p \to q \dashv\vdash \neg p \vee q$$
$$p \wedge q \to p \dashv\vdash r \vee \neg r \qquad p \wedge q \to r \dashv\vdash p \to (q \to r).$$

rules 太多: 推演过于复杂, 符号也有冗余

- 减少冗余的符号，设计自动推演算法

问题: 如何减少冗余的符号，设计自动推演算法?
先给部分结果:

- CNF (conjunctive normal form) 合取范式
  - 取如下 (一元、二元) 符号
    - $\{\wedge, \vee, \neg\}$
- Horn clauses 霍恩子句
  - 取如下 (一元、二元) 符号
    - $\{\wedge, \to\}$

回顾：SAT 求解所遇到的问题:
Provable equivalence:

$$\neg(p \wedge q) \dashv\vdash \neg q \vee \neg p \qquad \neg(p \vee q) \dashv\vdash \neg q \wedge \neg p$$
$$p \rightarrow q \dashv\vdash \neg q \rightarrow \neg p \qquad p \rightarrow q \dashv\vdash \neg p \vee q$$
$$p \wedge q \rightarrow p \dashv\vdash r \vee \neg r \qquad p \wedge q \rightarrow r \dashv\vdash p \rightarrow (q \rightarrow r).$$

rules 太多: 推演过于复杂, 符号也有冗余

- 减少冗余的符号，设计自动推演算法

问题: 如何减少冗余的符号，设计自动推演算法?
先给部分结果：

- CNF (conjunctive normal form) 合取范式
  - 取如下 (一元、二元) 符号
    - $\{\wedge, \vee, \neg\}$
- Horn clauses 霍恩子句
  - 取如下 (一元、二元) 符号
    - $\{\wedge, \rightarrow\}$

回顾: SAT 的一种求解思路:

Two Problems:

- Problem 1: Checking SAT of a proposition formula
- Problem 2: Checking SAT of a CNF formula

How to solve problem 1?

- Step 1: Transform Problem 1 to Problem 2
- Step 2: Solve Problem 2.

Step 1 (one way by applying the following rules):

- $\neg, \vee, \wedge$: Do nothing
- $\rightarrow$: $p \rightarrow q \equiv \neg p \vee q$
- $\leftrightarrow$: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Step 1 (another clever way): *Tseitin transformation*.

Step 2: 已解

回顾: SAT 的一种求解思路:

Two Problems:

- Problem 1: Checking SAT of a proposition formula
- Problem 2: Checking SAT of a CNF formula

How to solve problem 1?

- Step 1: Transform Problem 1 to Problem 2
- Step 2: Solve Problem 2.

Step 1 (one way by applying the following rules):

- $\neg, \vee, \wedge$: Do nothing
- $\rightarrow$: $p \rightarrow q \equiv \neg p \vee q$
- $\leftrightarrow$: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Step 1 (another clever way): *Tseitin transformation*.

Step 2: 已解

回顾: SAT 的一种求解思路:

Two Problems:

- Problem 1: Checking SAT of a proposition formula
- Problem 2: Checking SAT of a CNF formula

How to solve problem 1?

- Step 1: Transform Problem 1 to Problem 2
- Step 2: Solve Problem 2.

Step 1 (one way by applying the following rules):

- $\neg, \vee, \wedge$: Do nothing
- $\rightarrow$: $p \rightarrow q \equiv \neg p \vee q$
- $\leftrightarrow$: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Step 1 (another clever way): *Tseitin transformation.*

Step 2: 已解

回顾: SAT 的一种求解思路:

Two Problems:

- Problem 1: Checking SAT of a proposition formula
- Problem 2: Checking SAT of a CNF formula

How to solve problem 1?

- Step 1: Transform Problem 1 to Problem 2
- Step 2: Solve Problem 2.

Step 1 (one way by applying the following rules):

- $\neg, \vee, \wedge$: Do nothing
- $\rightarrow$: $p \rightarrow q \equiv \neg p \vee q$
- $\leftrightarrow$: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Step 1 (another clever way): *Tseitin transformation.*

Step 2: 已解

回顾: SAT 的一种求解思路:

Two Problems:

- Problem 1: Checking SAT of a proposition formula
- Problem 2: Checking SAT of a CNF formula

How to solve problem 1?

- Step 1: Transform Problem 1 to Problem 2
- Step 2: Solve Problem 2.

Step 1 (one way by applying the following rules):

- $\neg, \vee, \wedge$: Do nothing
- $\rightarrow$: $p \rightarrow q \equiv \neg p \vee q$
- $\leftrightarrow$: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Step 1 (another clever way): *Tseitin transformation*.

Step 2: 已解

回顾: SAT 的一种求解思路:

Two Problems:

- Problem 1: Checking SAT of a proposition formula
- Problem 2: Checking SAT of a CNF formula

How to solve problem 1?

- Step 1: Transform Problem 1 to Problem 2
- Step 2: Solve Problem 2.

Step 1 (one way by applying the following rules):

- $\neg, \vee, \wedge$: Do nothing
- $\rightarrow$: $p \rightarrow q \equiv \neg p \vee q$
- $\leftrightarrow$: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

Step 1 (another clever way): *Tseitin transformation*.

Step 2: 已解

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
    - show how it is possible
    - why a naive solution may blow up
- *Tseitin transformation*
    - linear in the size of the formula
    - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

How to solve SAT based on *Horn clauses* instead of CNF?

For any formula ϕ we can make its truth table

For *any 0* in this truth table, we can make a correpsonding clause

p	q	r	ϕ	$p \vee \neg q \vee r$	$\neg p \vee q \vee \neg r$	$\neg p \vee \neg q \vee \neg r$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	0	0	1	1
0	1	1	1	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	0	1
1	1	0	1	1	1	1
1	1	1	0	1	1	0

Now the conjunction $(p \vee \neg q \vee r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee \neg r)$ of these clauses has the same truth table as ϕ, so it is logically equivalent to ϕ

For any formula ϕ we can make its truth table

For *any 0* in this truth table, we can make a correpsonding clause

p	q	r	ϕ	$p \vee \neg q \vee r$	$\neg p \vee q \vee \neg r$	$\neg p \vee \neg q \vee \neg r$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	0	0	1	1
0	1	1	1	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	0	1
1	1	0	1	1	1	1
1	1	1	0	1	1	0

Now the conjunction $(p \vee \neg q \vee r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee \neg r)$ of these clauses has the same truth table as ϕ, so it is logically equivalent to ϕ

For any formula ϕ we can make its truth table

For *any 0* in this truth table, we can make a correpsonding clause

p	q	r	ϕ	$p \vee \neg q \vee r$	$\neg p \vee q \vee \neg r$	$\neg p \vee \neg q \vee \neg r$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	0	0	1	1
0	1	1	1	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	0	1
1	1	0	1	1	1	1
1	1	1	0	1	1	0

Now the conjunction $(p \vee \neg q \vee r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee \neg r)$ of these clauses has the same truth table as ϕ, so it is logically equivalent to ϕ

For any formula ϕ we can make its truth table

For *any 0* in this truth table, we can make a correpsonding clause

p	q	r	ϕ	$p \vee \neg q \vee r$	$\neg p \vee q \vee \neg r$	$\neg p \vee \neg q \vee \neg r$
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	0	0	1	1
0	1	1	1	1	1	1
1	0	0	1	1	1	1
1	0	1	0	1	0	1
1	1	0	1	1	1	1
1	1	1	0	1	1	0

Now the conjunction $(p \vee \neg q \vee r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee \neg r)$ of these clauses has the same truth table as ϕ, so it is logically equivalent to ϕ

This approach always works: if the truth table of ϕ *contains k 0's*, then we obtain a CNF consisting of *k clauses*

Drawback: this k may be very large

- consider the case: n variables in ϕ
  - How many clauses for constructing ϕ?
  - How many literals for each clause?

Good case: A smaller CNF logically equivalent to ϕ may exist, have clauses of $\leq n$ literals

- Example: $p \wedge (\neg q \vee r)$ is a CNF with 2 clauses, having 5 0's in truth table of 8 rows

Bad case: For some formulas the *exponential* number of clauses is *unavoidable* (见下页)

This approach always works: if the truth table of ϕ *contains k 0's*, then we obtain a CNF consisting of *k clauses*

Drawback: this k may be very large

- consider the case: n variables in ϕ
  - How many clauses for constructing ϕ?
  - How many literals for each clause?

Good case: A smaller CNF logically equivalent to ϕ may exist, have clauses of $\leq n$ literals

- Example: $p \wedge (\neg q \vee r)$ is a CNF with 2 clauses, having 5 0's in truth table of 8 rows

Bad case: For some formulas the *exponential* number of clauses is *unavoidable* (见下页)

This approach always works: if the truth table of ϕ *contains k 0's*, then we obtain a CNF consisting of *k clauses*

Drawback: this k may be very large

- consider the case: n variables in ϕ
  - How many clauses for constructing ϕ?
  - How many literals for each clause?

Good case: A smaller CNF logically equivalent to ϕ may exist, have clauses of $\leq n$ literals

- Example: $p \wedge (\neg q \vee r)$ is a CNF with 2 clauses, having 5 0's in truth table of 8 rows

Bad case: For some formulas the *exponential* number of clauses is *unavoidable* (见下页)

This approach always works: if the truth table of ϕ *contains k 0's*, then we obtain a CNF consisting of *k clauses*

Drawback: this k may be very large

- consider the case: n variables in ϕ
  - How many clauses for constructing ϕ?
  - How many literals for each clause?

Good case: A smaller CNF logically equivalent to ϕ may exist, have clauses of $\leq n$ literals

- Example: $p \wedge (\neg q \vee r)$ is a CNF with 2 clauses, having 5 0's in truth table of 8 rows

Bad case: For some formulas the *exponential* number of clauses is *unavoidable* (见下页)

This approach always works: if the truth table of ϕ *contains k 0's*, then we obtain a CNF consisting of *k clauses*

Drawback: this k may be very large

- consider the case: n variables in ϕ
  - How many clauses for constructing ϕ?
  - How many literals for each clause?

Good case: A smaller CNF logically equivalent to ϕ may exist, have clauses of $\leq n$ literals

- Example: $p \wedge (\neg q \vee r)$ is a CNF with 2 clauses, having 5 0's in truth table of 8 rows

Bad case: For some formulas the *exponential* number of clauses is *unavoidable* (见下页)

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

This formula yields true iff an *even number* of $p_i's$ has the value false

命题

Let X be a CNF satisfying $\Phi \equiv X$

Then every clause C in X contains exactly n literals

证明:

Assume not, then some p_i does not occur in a clause C of X

Then you can give values to the remaining variables such that C is false, and X is false too, independent of the value of p_i

Swapping values of p_i does not swap values of X, contradicting $\Phi \equiv X$

□

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$
This formula yields true iff an *even number* of $p_i' s$ has the value false

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

证明:
Assume not, then some p_i does not occur in a clause C of X

Then you can give values to the remaining variables such that C is false, and X is false too, independent of the value of p_i

Swapping values of p_i does not swap values of X, contradicting $\Phi \equiv X$
□

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$
This formula yields true iff an *even number* of $p_i's$ has the value false

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

证明:
Assume not, then some p_i does not occur in a clause C of X

Then you can give values to the remaining variables such that C is false, and X is false too, independent of the value of p_i

Swapping values of p_i does not swap values of X, contradicting $\Phi \equiv X$
□

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$
This formula yields true iff an *even number* of $p_i's$ has the value false

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

证明:

Assume not, then some p_i does not occur in a clause C of X

Then you can give values to the remaining variables such that C is false, and X is false too, independent of the value of p_i

Swapping values of p_i does not swap values of X, contradicting $\Phi \equiv X$
□

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$
This formula yields true iff an *even number* of $p_i's$ has the value false

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

证明:
Assume not, then some p_i does not occur in a clause C of X

Then you can give values to the remaining variables such that C is false, and X is false too, independent of the value of p_i

Swapping values of p_i does not swap values of X, contradicting $\Phi \equiv X$
□

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

This formula yields true iff an *even number* of $p_i's$ has the value false

命题

Let X be a CNF satisfying $\Phi \equiv X$

Then every clause C in X contains exactly n literals

证明:

Assume not, then some p_i does not occur in a clause C of X

Then you can give values to the remaining variables such that C is false, and X is false too, independent of the value of p_i

Swapping values of p_i does not swap values of X, contradicting $\Phi \equiv X$

$\square$

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

The truth table of Φ contains 2^n rows, half of which containing 0

So exactly 2^{n-1} rows contain 0

Every clause of exactly n literals has one 0 in its truth table

So we need 2^{n-1} *such clauses* to obtain the truth table of Φ

So for this Φ the *exponential* size is unavoidable

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

The truth table of Φ contains 2^n rows, half of which containing 0

So exactly 2^{n-1} rows contain 0

Every clause of exactly n literals has one 0 in its truth table

So we need 2^{n-1} *such clauses* to obtain the truth table of Φ

So for this Φ the *exponential* size is unavoidable

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

The truth table of Φ contains 2^n rows, half of which containing 0

So exactly 2^{n-1} rows contain 0

Every clause of exactly n literals has one 0 in its truth table

So we need 2^{n-1} *such clauses* to obtain the truth table of Φ

So for this Φ the *exponential* size is unavoidable

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

The truth table of Φ contains 2^n rows, half of which containing 0

So exactly 2^{n-1} rows contain 0

Every clause of exactly n literals has one 0 in its truth table

So we need 2^{n-1} *such clauses* to obtain the truth table of Φ

So for this Φ the *exponential* size is unavoidable

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

The truth table of Φ contains 2^n rows, half of which containing 0

So exactly 2^{n-1} rows contain 0

Every clause of exactly n literals has one 0 in its truth table

So we need 2^{n-1} *such clauses* to obtain the truth table of Φ

So for this Φ the *exponential* size is unavoidable

Example: $\Phi : (\cdots((p_1 \leftrightarrow p_2) \leftrightarrow p_3) \cdots \leftrightarrow p_n)$

命题

Let X be a CNF satisfying $\Phi \equiv X$
Then every clause C in X contains exactly n literals

The truth table of Φ contains 2^n rows, half of which containing 0

So exactly 2^{n-1} rows contain 0

Every clause of exactly n literals has one 0 in its truth table

So we need 2^{n-1} *such clauses* to obtain the truth table of Φ

So for this Φ the *exponential* size is unavoidable

Summarizing the *challenge*:

- For *any* propositional formula ϕ, it is possible to find a logically equivalent CNF
- Bad case: but the size of this CNF may be *exponential*

新方法: *Tseitin transformation* (见下页)

Summarizing the *challenge*:

- For *any* propositional formula ϕ, it is possible to find a logically equivalent CNF
- Bad case: but the size of this CNF may be *exponential*

新方法: *Tseitin transformation* (见下页)

Summarizing the *challenge*:

- For *any* propositional formula ϕ, it is possible to find a logically equivalent CNF
- Bad case: but the size of this CNF may be *exponential*

新方法: *Tseitin transformation* (见下页)

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

Tseitin transformation

- *Linear* transformation of *arbitrary* propositional formula to CNF

思路: Give a name to every subformula (except literals) and use this name as a *fresh* variable

- For every formula ϕ on ≤ 3 variables there is a small CNF *cnf*$(\phi) \equiv \phi$
- Transform a *big* formula ϕ to the conjunction of *cnf*(ϕ_i) for many small formulas ϕ_i obtained from ϕ, one for each subformula

More precisely, for every subformula ψ, we define

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$

cnf$(n_\phi \leftrightarrow (B \leftrightarrow C)) \wedge$

cnf$(B \leftrightarrow (\neg s \wedge p)) \wedge$

cnf$(C \leftrightarrow (D \vee \neg p)) \wedge$

cnf$(D \leftrightarrow (q \rightarrow r))$

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$

cnf$(n_\phi \leftrightarrow (B \leftrightarrow C)) \wedge$

cnf$(B \leftrightarrow (\neg s \wedge p)) \wedge$

cnf$(C \leftrightarrow (D \vee \neg p)) \wedge$

cnf$(D \leftrightarrow (q \rightarrow r))$

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$

cnf$(n_\phi \leftrightarrow (B \leftrightarrow C)) \wedge$

cnf$(B \leftrightarrow (\neg s \wedge p)) \wedge$

cnf$(C \leftrightarrow (D \vee \neg p)) \wedge$

cnf$(D \leftrightarrow (q \rightarrow r))$

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$
cnf$(n_\phi \leftrightarrow (B \leftrightarrow C)) \wedge$
cnf$(B \leftrightarrow (\neg s \wedge p)) \wedge$
cnf$(C \leftrightarrow (D \vee \neg p)) \wedge$
cnf$(D \leftrightarrow (q \rightarrow r))$

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$

cnf$(n_\phi \leftrightarrow (B \leftrightarrow C)) \wedge$

cnf$(B \leftrightarrow (\neg s \wedge p)) \wedge$

cnf$(C \leftrightarrow (D \vee \neg p)) \wedge$

cnf$(D \leftrightarrow (q \rightarrow r))$

2. 理论

2.3 CNF and Horn Clauses | Tseitin transformation

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$

cnf$(n_\phi \leftrightarrow (B \leftrightarrow C)) \wedge$

cnf$(B \leftrightarrow (\neg s \wedge p)) \wedge$

cnf$(C \leftrightarrow (D \vee \neg p)) \wedge$

cnf$(D \leftrightarrow (q \rightarrow r))$

More precisely, for every subformula ψ

- $n_\psi = \psi$, if ψ is a literal
- $n_\psi =$ the name of ψ, otherwise

The Tseitin transformation $T(\phi)$ of ϕ, is defined to be the CNF consisting of:

- n_ψ
- *cnf*$(q \leftrightarrow \neg n_\psi)$ for every non-literal subformula of the shape $\neg\psi$ having name q
- *cnf*$(q \leftrightarrow (n_{\psi_1} \diamond n_{\psi_2}))$ for every subformula of the shape $\psi_1 \diamond \psi_2$ having name q, for

$$\diamond \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$$

Example ϕ:

$$\underbrace{(\neg s \wedge p)}_{B} \leftrightarrow \underbrace{(\underbrace{(q \rightarrow r)}_{D} \vee \neg p)}_{C}$$

yields $T(\phi)$:

$n_\phi \wedge$

cnf$(n_\phi \leftrightarrow (B \leftrightarrow C))\wedge$

cnf$(B \leftrightarrow (\neg s \wedge p))\wedge$

cnf$(C \leftrightarrow (D \vee \neg p))\wedge$

cnf$(D \leftrightarrow (q \rightarrow r))$

定理

ϕ is satisfiable if and only if $T(\phi)$ is satisfiable

证明：略（若感兴趣，可见 note）

剩下的问题：We still need to compute the formula *cnf*$(n_\psi \leftrightarrow \cdots)$

$$cnf(p \leftrightarrow \neg q) = (p \vee q) \wedge (\neg p \vee \neg q)$$

$$cnf(p \leftrightarrow (q \vee r)) = (\neg p \vee q \vee r) \wedge (p \vee \neg q) \wedge (p \vee \neg r)$$

$$cnf(p \leftrightarrow (q \wedge r)) = (p \vee \neg q \vee \neg r) \wedge (\neg p \vee q) \wedge (\neg p \vee r)$$

$$cnf(p \leftrightarrow (q \leftrightarrow r)) = (p \vee q \vee r) \wedge (p \vee \neg q \vee \neg r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee r)$$

定理

ϕ is satisfiable if and only if $T(\phi)$ is satisfiable

证明: 略（若感兴趣，可见 note）

剩下的问题: We still need to compute the formula *cnf*$(n_\psi \leftrightarrow \cdots)$

$$\mathit{cnf}(p \leftrightarrow \neg q) = (p \vee q) \wedge (\neg p \vee \neg q)$$

$$\mathit{cnf}(p \leftrightarrow (q \vee r)) = (\neg p \vee q \vee r) \wedge (p \vee \neg q) \wedge (p \vee \neg r)$$

$$\mathit{cnf}(p \leftrightarrow (q \wedge r)) = (p \vee \neg q \vee \neg r) \wedge (\neg p \vee q) \wedge (\neg p \vee r)$$

$$\mathit{cnf}(p \leftrightarrow (q \leftrightarrow r)) = (p \vee q \vee r) \wedge (p \vee \neg q \vee \neg r) \wedge (\neg p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee r)$$

Concluding

- For every propositional formula ϕ its Tseitin transformation $T(\phi)$ is easily computed
- Size of $T(\phi)$ is linear in size of ϕ
- Preserves satisfiability
- Not only CNF, even *3-CNF*
- Used in current SAT solvers

Concluding

- For every propositional formula ϕ its Tseitin transformation $T(\phi)$ is easily computed
- Size of $T(\phi)$ is linear in size of ϕ
- Preserves satisfiability
- Not only CNF, even *3-CNF*
- Used in current SAT solvers

Concluding

- For every propositional formula ϕ its Tseitin transformation $T(\phi)$ is easily computed
- Size of $T(\phi)$ is linear in size of ϕ
- Preserves satisfiability
- Not only CNF, even *3-CNF*
- Used in current SAT solvers

Concluding

- For every propositional formula ϕ its Tseitin transformation $T(\phi)$ is easily computed
- Size of $T(\phi)$ is linear in size of ϕ
- Preserves satisfiability
- Not only CNF, even *3-CNF*
- Used in current SAT solvers

Concluding

- For every propositional formula ϕ its Tseitin transformation $T(\phi)$ is easily computed
- Size of $T(\phi)$ is linear in size of ϕ
- Preserves satisfiability
- Not only CNF, even *3-CNF*
- Used in current SAT solvers

Concluding

- For every propositional formula ϕ its Tseitin transformation $T(\phi)$ is easily computed
- Size of $T(\phi)$ is linear in size of ϕ
- Preserves satisfiability
- Not only CNF, even *3-CNF*
- Used in current SAT solvers

回顾本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

问题: How to solve SAT based on *Horn clauses* instead of CNF?

回顾本节内容:

How to transform a propositional formula to CNF?

- *challenge*:
  - show how it is possible
  - why a naive solution may blow up
- *Tseitin transformation*
  - linear in the size of the formula
  - used in current SAT solvers

问题: How to solve SAT based on *Horn clauses* instead of CNF?

定义: Horn clause

A *Horn formula* is a formula ϕ propositional logic if it can be generated as instance of H in this grammar:

$$P ::== \bot \mid \top \mid p$$
$$A ::== P \mid P \wedge A$$
$$C ::== A \to P$$
$$H ::== C \mid C \wedge H$$

We call each instance of C a *Horn clause*.

定义: Horn clause

A *Horn formula* is a formula ϕ propositional logic if it can be generated as instance of H in this grammar:

$$P ::== \bot \mid \top \mid p$$
$$A ::== P \mid P \wedge A$$
$$C ::== A \to P$$
$$H ::== C \mid C \wedge H$$

We call each instance of C a *Horn clause*.

Recall that the logical constants:

- $\bot$ denotes an unsatisfiable formula
- $\top$ denotes a tautology

定义: Horn clause

A *Horn formula* is a formula ϕ propositional logic if it can be generated as instance of H in this grammar:

$$P ::== \bot \mid \top \mid p$$
$$A ::== P \mid P \wedge A$$
$$C ::== A \to P$$
$$H ::== C \mid C \wedge H$$

We call each instance of C a *Horn clause*.

例: Examples of *Horn formulas*:

$$(p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$$
$$(p \wedge q \wedge s \to \bot) \wedge (q \wedge r \to p) \wedge (\top \to s)$$
$$(p_2 \wedge p_3 \wedge p_5 \to p_{13}) \wedge (\top \to p_5) \wedge (p_5 \wedge p_{11} \to \bot)$$

定义: Horn clause

A *Horn formula* is a formula ϕ propositional logic if it can be generated as instance of H in this grammar:

$$
\begin{aligned}
P &::== \bot \mid \top \mid p \\
A &::== P \mid P \wedge A \\
C &::== A \to P \\
H &::== C \mid C \wedge H
\end{aligned}
$$

We call each instance of C a *Horn clause*.

例: Examples of formulas which are *not Horn formulas*:

$$
\begin{aligned}
&(p \wedge q \wedge s \to \neg p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s) \\
&(p \wedge q \wedge s \to \bot) \wedge (\neg q \wedge r \to p) \wedge (\top \to s) \\
&(p_2 \wedge p_3 \wedge p_5 \to p_{13} \wedge p_{27}) \wedge (\top \to p_5) \wedge (p_5 \wedge p_{11} \vee \bot)
\end{aligned}
$$

算法: HORN(ϕ)

begin function

  mark all occurrences of $\top$ in ϕ;

  while there is a conjunct $P_1 \wedge P_2 \wedge \cdots \wedge P_{k_i} \to P'$ of ϕ such that

    all P_j are marked but P' isn't

   do mark P'

  end while

  if $\bot$ is marked

   then return 'unsatisfiable'

   else return 'satisfiable'

end function

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \land q \land w \to \bot) \land (t \to \bot) \land (r \to p) \land (\top \to r) \land (\top \to q) \land (u \to s) \land (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \land q \land w \to \bot) \land (t \to \bot) \land (r \to p) \land (\top \to r) \land (\top \to q) \land (r \land u \to w) \land (u \to s) \land (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \land q \land s \to p) \land (q \land r \to p) \land (p \land s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \rightarrow \bot) \wedge (t \rightarrow \bot) \wedge (r \rightarrow p) \wedge (\top \rightarrow r) \wedge (\top \rightarrow q) \wedge (r \wedge u \rightarrow w) \wedge (u \rightarrow s) \wedge (\top \rightarrow u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \rightarrow p) \wedge (q \wedge r \rightarrow p) \wedge (p \wedge s \rightarrow s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \wedge q \wedge w \to \bot) \wedge (t \to \bot) \wedge (r \to p) \wedge (\top \to r) \wedge (\top \to q) \wedge (r \wedge u \to w) \wedge (u \to s) \wedge (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \wedge q \wedge s \to p) \wedge (q \wedge r \to p) \wedge (p \wedge s \to s)$

Marked: None... return 'satisfiable'

例 1: Horn (ϕ)

$\phi = (p \land q \land w \to \bot) \land (t \to \bot) \land (r \to p) \land (\top \to r) \land (\top \to q) \land (u \to s) \land (\top \to u)$

Marked: $\top$ r q u p s return 'satisfiable'

例 2: Horn (ϕ)

$\phi = (p \land q \land w \to \bot) \land (t \to \bot) \land (r \to p) \land (\top \to r) \land (\top \to q) \land (r \land u \to w) \land (u \to s) \land (\top \to u)$

Marked: $\top$ r q u p w $\bot$ return 'unsatisfiable'

例 3: Horn (ϕ)

$\phi = (p \land q \land s \to p) \land (q \land r \to p) \land (p \land s \to s)$

Marked: None... return 'satisfiable'

Concluding

- There are practically important subclasses, e.g., *Horn formulas*, which have much more *efficient ways* of deciding their satisfiability
- *Horn clauses* have been applied to many classical formal verifiers, e.g., the protocol verifier ProVerif.
- How to transform propositional formulas to Horn formulas?

Concluding

- There are practically important subclasses, e.g., *Horn formulas*, which have much more *efficient ways* of deciding their satisfiability
- *Horn clauses* have been applied to many classical formal verifiers, e.g., the protocol verifier ProVerif.
- How to transform propositional formulas to Horn formulas?

Concluding

- There are practically important subclasses, e.g., *Horn formulas*, which have much more *efficient ways* of deciding their satisfiability
- *Horn clauses* have been applied to many classical formal verifiers, e.g., the protocol verifier ProVerif.
- How to transform propositional formulas to Horn formulas?

Concluding

- There are practically important subclasses, e.g., *Horn formulas*, which have much more *efficient ways* of deciding their satisfiability
- *Horn clauses* have been applied to many classical formal verifiers, e.g., the protocol verifier ProVerif.
- How to transform propositional formulas to Horn formulas?

作业

1. Construct a formula in CNF based on each of the following truth tables:

p	q	r	ϕ
1	1	1	1
1	1	0	0
1	0	1	0
0	1	1	1
1	0	0	0
0	1	0	0
0	0	1	1
0	0	0	0

2. Apply algorithm HORN to each of these Horn formulas:
   1. $(p \wedge q \wedge s \to \bot) \wedge (q \wedge r \to p) \wedge (\top \to s)$
   2. $(p_5 \to p_{11}) \wedge (p_2 \wedge p_3 \wedge p_5 \to p_{13}) \wedge (\top \to p_5) \wedge (p_5 \wedge p_{11} \to \bot)$
   3. $(\top \to q) \wedge (\top \to s) \wedge (w \to \bot) \wedge (p \wedge q \wedge s \to \bot) \wedge (v \to s) \wedge (\top \to s) \wedge (r \to p)$