# Current Research in Proof Complexity: Lecture 1 Introduction, Course Overview, and Practicalities

Jakob Nordström

KTH Royal Institute of Technology

Monday October 24, 2011

# The Subject Matter of This Course

- What is a proof?
- Which (logical) statements have efficient proofs?
- How can we find such proofs? (Can we?)
- What are good methods of reasoning about logical statements?
- What are natural notions of "efficiency" of proofs? (size, complexity, et cetera)
- How are these notions related?

# So What Is a Proof?

Claim: 25957 is the product of two primes.
True or false? What kind of proof would convince us?

- "I told you so. Just factor and check it yourself!"
  Not much of a proof.

- $25957 \equiv 1 \pmod 2$ $\quad$ $25957 \equiv 0 \pmod{101}$
  $25957 \equiv 1 \pmod 3$ $\quad$ $25957 \equiv 1 \pmod{103}$
  $25957 \equiv 2 \pmod 5$ $\quad$ $\vdots$
  $\vdots$ $\quad$ $25957 \equiv 0 \pmod{257}$
  $25957 \equiv 19 \pmod{99}$ $\quad$ $\vdots$
  OK, but maybe even a bit of overkill.

- "$25957 = 101 \cdot 257$; check yourself that these are primes."

Key demand: A proof should be efficiently verifiable.

# So What Is a Proof?

Claim: 25957 is the product of two primes.
True or false? What kind of proof would convince us?

- "I told you so. Just factor and check it yourself!"
  Not much of a proof.
- $25957 \equiv 1 \pmod 2$ $\quad$ $25957 \equiv 0 \pmod{101}$
  $25957 \equiv 1 \pmod 3$ $\quad$ $25957 \equiv 1 \pmod{103}$
  $25957 \equiv 2 \pmod 5$ $\quad$ $\vdots$
  $\vdots$ $\quad$ $25957 \equiv 0 \pmod{257}$
  $25957 \equiv 19 \pmod{99}$ $\quad$ $\vdots$
  OK, but maybe even a bit of overkill.
- "$25957 = 101 \cdot 257$; check yourself that these are primes."

Key demand: A proof should be efficiently verifiable.

# So What Is a Proof?

Claim: 25957 is the product of two primes.
True or false? What kind of proof would convince us?

- "I told you so. Just factor and check it yourself!"
  Not much of a proof.

- $25957 \equiv 1 \pmod 2$ $\quad$ $25957 \equiv 0 \pmod{101}$
  $25957 \equiv 1 \pmod 3$ $\quad$ $25957 \equiv 1 \pmod{103}$
  $25957 \equiv 2 \pmod 5$ $\quad$ $\vdots$
  $\vdots$ $\quad$ $25957 \equiv 0 \pmod{257}$
  $25957 \equiv 19 \pmod{99}$ $\quad$ $\vdots$
  OK, but maybe even a bit of overkill.

- "$25957 = 101 \cdot 257$; check yourself that these are primes."

Key demand: A proof should be efficiently verifiable.

# So What Is a Proof?

Claim: 25957 is the product of two primes.
True or false? What kind of proof would convince us?

- "I told you so. Just factor and check it yourself!"
  Not much of a proof.

- $25957 \equiv 1 \pmod 2$ $25957 \equiv 0 \pmod{101}$
  $25957 \equiv 1 \pmod 3$ $25957 \equiv 1 \pmod{103}$
  $25957 \equiv 2 \pmod 5$ $\vdots$
  $\vdots$ $25957 \equiv 0 \pmod{257}$
  $25957 \equiv 19 \pmod{99}$ $\vdots$
  OK, but maybe even a bit of overkill.

- "$25957 = 101 \cdot 257$; check yourself that these are primes."

Key demand: A proof should be efficiently verifiable.

# Proof system

Proof system for a language $L$ (adapted from [Cook & Reckhow '79]):

Deterministic algorithm $P(x, \pi)$ that runs in time polynomial in $|x|$ and $|\pi|$ such that

- for all $x \in L$ there is a string $\pi$ (a proof) such that $P(x, \pi) = 1$,
- for all $x \notin L$ it holds for all strings $\pi$ that $P(x, \pi) = 0$.

Think of $P$ as "proof checker"
Note that proof $\pi$ can be very large compared to $x$
Only have to achieve polynomial time in $|x| + |\pi|$

Propositional proof system: proof system for the language TAUT of all valid propositional logic formulas (or tautologies)

# Propositional Logic: Syntax

Set $Vars$ of Boolean variables ranging over $\{0, 1\}$ (false and true)

Logical connectives:

- negation $\neg$,
- conjunction $\wedge$,
- disjunction $\vee$,
- implication $\rightarrow$,
- equivalence $\leftrightarrow$.

Set $\textsc{prop}$ of propositional logic formulas is smallest set $X$ such that

- $x \in X$ for all propositional logic variables $x \in Vars$,
- if $F, G \in X$ then $\big(F \wedge G\big), \big(F \vee G\big), \big(F \rightarrow G\big), \big(F \leftrightarrow G\big) \in X$,
- if $F \in X$ then $\big(\neg F\big) \in X$.

# Propositional Logic: Semantics

Let $\alpha$ denote a truth value assignment, i.e., $\alpha : Vars \mapsto \{0, 1\}$

Extend $\alpha$ from variables to formulas by:

- $\alpha(\neg F) = 1$ if $\alpha(F) = 0$
- $\alpha(F \vee G) = 1$ unless $\alpha(F) = \alpha(G) = 0$
- $\alpha(F \wedge G) = 1$ if $\alpha(F) = \alpha(G) = 1$
- $\alpha(F \rightarrow G) = 1$ unless $\alpha(F) = 1$ and $\alpha(G) = 0$
- $\alpha(F \leftrightarrow G) = 1$ if $\alpha(F) = \alpha(G)$

We say that $F$ is

- satisfiable if there is an assignment $\alpha$ with $\alpha(F) = 1$
- valid or tautological if all assignments satisfy $F$
- falsifiable if there is an assignment $\alpha$ with $\alpha(F) = 0$
- unsatisfiable or contradictory if all assignments falsify $F$

# Example Propositional Proof System

## Example (Truth table)

| $p$ | $q$ | $r$ | $(p \wedge (q \vee r)) \leftrightarrow ((p \wedge q) \vee (p \wedge r))$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Certainly polynomial-time checkable measured in "proof" size
Why does this not make us happy?

# Example Propositional Proof System

## Example (Truth table)

| $p$ | $q$ | $r$ | $(p \wedge (q \vee r)) \leftrightarrow ((p \wedge q) \vee (p \wedge r))$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Certainly polynomial-time checkable measured in "proof" size
Why does this not make us happy?

# Proof System Complexity

Complexity $cplx(P)$ of a proof system $P$:

Smallest $g : \mathbb{N} \mapsto \mathbb{N}$ such that $x \in L$ if and only if there is a proof $\pi$ of size $|\pi| \leq g(|x|)$ such that $P(x, \pi) = 1$.

If a proof system is of polynomial complexity, it is said to be polynomially bounded or $p$-bounded.

**Example (Truth table continued)**

Truth table is a propositional proof system, but of exponential complexity!

# Proof System Complexity

Complexity $cplx(P)$ of a proof system $P$:

Smallest $g : \mathbb{N} \mapsto \mathbb{N}$ such that $x \in L$ if and only if there is a proof $\pi$ of size $|\pi| \leq g(|x|)$ such that $P(x, \pi) = 1$.

If a proof system is of polynomial complexity, it is said to be polynomially bounded or $p$-bounded.

**Example (Truth table continued)**

Truth table is a propositional proof system, but of exponential complexity!

# Proof systems and P vs. NP

**Theorem (Cook & Reckhow '79)**

NP = co-NP *if and only if there exists a polynomially bounded propositional proof system.*

**Proof.**

NP *exactly* the set of languages with $p$-bounded proof systems

($\Rightarrow$) $\textsc{taut} \in$ co-NP since $F$ is *not* a tautology iff $\neg F \in \textsc{sat}$.
If NP = co-NP, then $\textsc{taut} \in$ NP has a $p$-bounded proof system by definition.

($\Leftarrow$) Suppose there exists a $p$-bounded proof system. Then $\textsc{taut} \in$ NP, and since $\textsc{taut}$ is complete for co-NP it follows that NP = co-NP. $\square$

# Proof systems and P vs. NP

**Theorem (Cook & Reckhow '79)**

NP = co-NP *if and only if there exists a polynomially bounded propositional proof system.*

**Proof.**

NP *exactly* the set of languages with $p$-bounded proof systems

($\Rightarrow$) $\text{TAUT} \in$ co-NP since $F$ is *not* a tautology iff $\neg F \in \text{SAT}$.
If NP = co-NP, then $\text{TAUT} \in$ NP has a $p$-bounded proof system by definition.

($\Leftarrow$) Suppose there exists a $p$-bounded proof system. Then $\text{TAUT} \in$ NP, and since TAUT is complete for co-NP it follows that NP = co-NP. $\square$

# Proof systems and P vs. NP

**Theorem (Cook & Reckhow '79)**

NP = co-NP *if and only if there exists a polynomially bounded propositional proof system.*

**Proof.**

NP *exactly* the set of languages with $p$-bounded proof systems

($\Rightarrow$) $\text{TAUT} \in$ co-NP since $F$ is *not* a tautology iff $\neg F \in \text{SAT}$.
If NP = co-NP, then $\text{TAUT} \in$ NP has a $p$-bounded proof system by definition.

($\Leftarrow$) Suppose there exists a $p$-bounded proof system. Then $\text{TAUT} \in$ NP, and since TAUT is complete for co-NP it follows that NP = co-NP. □

# Proof systems and P vs. NP

**Theorem (Cook & Reckhow '79)**

NP = co-NP *if and only if there exists a polynomially bounded propositional proof system.*

**Proof.**

NP *exactly* the set of languages with $p$-bounded proof systems

($\Rightarrow$) $\text{TAUT} \in$ co-NP since $F$ is *not* a tautology iff $\neg F \in \text{SAT}$.
If NP = co-NP, then $\text{TAUT} \in$ NP has a $p$-bounded proof system by definition.

($\Leftarrow$) Suppose there exists a $p$-bounded proof system. Then $\text{TAUT} \in$ NP, and since TAUT is complete for co-NP it follows that NP = co-NP. $\square$

# Polynomial Simulation

The convential wisdom is that NP $\neq$ co-NP
Seems that proof of this is lightyears away
(Would imply P $\neq$ NP as a corollary)

**Reason 1 for proof complexity:** approach this distant goal by studying successively stronger proof systems and relating their strengths

**Definition ($p$-simulation)**

$P_1$ polynomially simulates, or $p$-simulates, $P_2$ if there exists a polynomial-time computable function $f$ such that for all $F \in \text{TAUT}$ it holds that $P_2(F, \pi) = 1$ iff $P_1(F, f(\pi)) = 1$.

Weak $p$-simulation: $cplx(P_1) = (cplx(P_2))^{\mathcal{O}(1)}$ but we do not know explicit translation function $f$ from $P_2$-proofs to $P_1$-proofs

# Polynomial Simulation

The convential wisdom is that NP $\neq$ co-NP
Seems that proof of this is lightyears away
(Would imply P $\neq$ NP as a corollary)

**Reason 1 for proof complexity:** approach this distant goal by studying successively stronger proof systems and relating their strengths

## Definition ($p$-simulation)

$P_1$ polynomially simulates, or $p$-simulates, $P_2$ if there exists a polynomial-time computable function $f$ such that for all $F \in \text{TAUT}$ it holds that $P_2(F, \pi) = 1$ iff $P_1(F, f(\pi)) = 1$.

Weak $p$-simulation: $cplx(P_1) = (cplx(P_2))^{\mathcal{O}(1)}$ but we do not know explicit translation function $f$ from $P_2$-proofs to $P_1$-proofs

# Polynomial Equivalence

**Definition ($p$-equivalence)**

Two propositional proof systems $P_1$ and $P_2$ are polynomially equivalent, or $p$-equivalent, if each proof system $p$-simulates the other.

If $P_1$ $p$-simulates $P_2$ but $P_2$ does not $p$-simulate $P_1$, then $P_1$ is strictly stronger than $P_2$

Lots of results proven relating strength of different propositional proof systems

But not the focus of this course (although we will see a few examples)

# A Fundamental Theoretical Problem...

The constructive version of the problem:

**Problem**

Given a propositional logic formula $F$, can we decide efficiently whether is it true no matter how we assign values to its variables?

TAUT: Fundamental problem in theoretical computer science ever since Stephen Cook's NP-completeness paper in 1971

(And significance realized much earlier — cf. Gödel's letter in 1956)

These days recognized as one of the main challenges for all of mathematics — one of the million dollar "Millennium Problems"

# A Fundamental Theoretical Problem...

The constructive version of the problem:

**Problem**

Given a propositional logic formula $F$, can we decide efficiently whether is it true no matter how we assign values to its variables?

TAUT: Fundamental problem in theoretical computer science ever since Stephen Cook's NP-completeness paper in 1971

(And significance realized much earlier — cf. Gödel's letter in 1956)

These days recognized as one of the main challenges for all of mathematics — one of the million dollar "Millennium Problems"

# . . . with Huge Practical Implications

- All known algorithms run in exponential time in worst case
- But enormous progress on applied computer programs last 10-15 years
- These so-called SAT solvers are routinely deployed to solve large-scale real-world problems with 100 000s or even 1 000 000s of variables
- Used in e.g. hardware verification, software testing, software package management, artificial intelligence, cryptography, bioinformatics, and more
- But we also know small example formulas with only hundreds of variables that trip up even state-of-the-art SAT solvers

# Automated Theorem Proving or SAT Solving

**Reason 2 for proof complexity:** understand proof systems used for solving formulas occurring in "real-world applications"

- Study proof systems used by SAT solvers
- Model actual methods of reasoning used by SAT solvers as "refinements" (subsystems) of these systems
- Prove upper and lower bounds in these systems
- Try to explain or predict theoretically what happens in practice

This course:

- Focus on proof systems used for SAT solving (resolution, polynomial calculus, cutting planes)
- Maybe not too much "low-level modelling"

# Automated Theorem Proving or SAT Solving

**Reason 2 for proof complexity:** understand proof systems used for solving formulas occurring in "real-world applications"

- Study proof systems used by SAT solvers
- Model actual methods of reasoning used by SAT solvers as "refinements" (subsystems) of these systems
- Prove upper and lower bounds in these systems
- Try to explain or predict theoretically what happens in practice

This course:

- Focus on proof systems used for SAT solving (resolution, polynomial calculus, cutting planes)
- Maybe not too much "low-level modelling"

# Proof Search Algorithms and Automatizability

Proof search algorithm $A_P$ for propositional proof system $P$:
Deterministic algorithm with

- input: formula $F$
- output: $P$-proof $\pi$ of $F$ or report that $F$ is falsifiable

**Definition (Automatizability)**

$P$ is automatizable if there exists a proof search algorithm $A_P$ such that if $F \in \text{TAUT}$ then $A_P$ on input $F$ outputs a $P$-proof of $F$ in time polynomial in *the size of a smallest $P$-proof of $F$*.

# Proof Search Algorithms and Automatizability

Proof search algorithm $A_P$ for propositional proof system $P$:
Deterministic algorithm with

- input: formula $F$
- output: $P$-proof $\pi$ of $F$ or report that $F$ is falsifiable

## Definition (Automatizability)

$P$ is automatizable if there exists a proof search algorithm $A_P$ such that if $F \in \text{TAUT}$ then $A_P$ on input $F$ outputs a $P$-proof of $F$ in time polynomial in *the size of a smallest $P$-proof of $F$*.

# Short Proofs Seem Hard to Find (at Least in Theory)

**Example (Truth table continued)**

Truth table is (trivially) an automatizable propositional proof system. (But the proofs we find are of exponential size, so this is not very exciting.)

We want proof systems that are *both*

- strong (i.e., have short proofs for all tautologies) and
- automatizable (i.e., we can find these short proofs)

Seems that this is not possible (under reasonable complexity assumptions)

But can find proof search algorithms that work really well "in practice"

# Short Proofs Seem Hard to Find (at Least in Theory)

**Example (Truth table continued)**

Truth table is (trivially) an automatizable propositional proof system. (But the proofs we find are of exponential size, so this is not very exciting.)

We want proof systems that are *both*

- strong (i.e., have short proofs for all tautologies) and
- automatizable (i.e., we can find these short proofs)

Seems that this is not possible (under reasonable complexity assumptions)

But can find proof search algorithms that work really well "in practice"

# Potential and Limitations of Mathematical Reasoning

**Reason 3 for proof complexity:** understand how deep / hard various mathematical truths are

- Look at logic encoding of various mathematical truths (e.g. combinatorial principles)
- Determine how strong proof systems are needed to provide efficient proofs
- Tells us how powerful mathematical tools are needed for establishing such statements

Fascinating area, but this course will not go into this at all

# Transforming Tautologies to Unsatisfiable CNFs

Any propositional logic formula $F$ can be converted to formula $F'$ in conjunctive normal form (CNF) such that

- $F'$ only linearly larger than $F$
- $F'$ unsatisfiable iff $F$ tautology

Idea [Tseitin '68]:

- Introduce new variable $x_G$ for each subformula $G \doteq H_1 \circ H_2$ in $F$, $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$
- Translate $G$ to set of disjunctive clauses $Cl(G)$ which enforces that truth value of $x_G$ is computed correctly given $x_{H_1}$ and $x_{H_2}$

# Transforming Tautologies to Unsatisfiable CNFs

Any propositional logic formula $F$ can be converted to formula $F'$ in conjunctive normal form (CNF) such that

- $F'$ only linearly larger than $F$
- $F'$ unsatisfiable iff $F$ tautology

Idea [Tseitin '68]:

- Introduce new variable $x_G$ for each subformula $G \doteq H_1 \circ H_2$ in $F$, $\circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$
- Translate $G$ to set of disjunctive clauses $Cl(G)$ which enforces that truth value of $x_G$ is computed correctly given $x_{H_1}$ and $x_{H_2}$

# Sketch of Transformation

Two examples for $\vee$ and $\rightarrow$ ($\wedge$ and $\leftrightarrow$ are analogous):

$$G \equiv H_1 \vee H_2 : \qquad Cl(G) := \left(\neg x_G \vee x_{H_1} \vee x_{H_2}\right) \wedge \left(x_G \vee \neg x_{H_1}\right) \wedge \left(x_G \vee \neg x_{H_2}\right)$$

$$G \equiv H_1 \rightarrow H_2 : \qquad Cl(G) := \left(\neg x_G \vee \neg x_{H_1} \vee x_{H_2}\right) \wedge \left(x_G \vee x_{H_1}\right) \wedge \left(x_G \vee \neg x_{H_2}\right)$$

- Finally, add clause $\neg x_F$

# Proof Systems for Refuting Unsatisfiable CNFs

- Easy to verify that constructed CNF formula $F'$ is unsatisfiable iff $F$ is a tautology
- So any sound and complete proof system which produces refutations of formulas in conjunctive normal form can be used as a propositional proof system
- From now on and for the rest of this course, we will discuss only such proof systems

# Some Notation and Terminology

- Literal $a$: variable $x$ or its negation $\overline{x}$ (rather than $\neg x$)
- Let $\overline{\overline{x}} = x$
- Sometimes write $x^1 = x$ and $x^0 = \overline{x}$
- Clause $C = a_1 \vee \ldots \vee a_k$: set of literals
  At most $k$ literals: $k$-clause
- CNF formula $F = C_1 \wedge \ldots \wedge C_m$: set of clauses
  $k$-CNF formula: CNF formula consisting of $k$-clauses
- $Vars(\cdot)$: set of variables in clause or formula
  $Lit(\cdot)$: set of literals in clause or formula
- $F \models D$: semantical implication, $\alpha(F)$ true $\Rightarrow \alpha(D)$ true
  for all truth value assignments $\alpha$
- $[n] = \{1, 2, \ldots, n\}$

# Sequential Proof Systems

A proof system $P$ is sequential if a proof $\pi$ in $P$ is a

- sequence of lines $\pi = \{L_1, \ldots, L_\tau\}$
- of some prescribed syntactic form
  (depending on the proof system in question)
- where each line is derived from previous lines by one of a finite set of allowed inference rules

(This will become clearer when we get some examples)

# Complexity Measures (High-level Intuition)

View a proof as

- non-deterministic Turing machine computation,
- special read-only input tape from which the clauses of $F$ (the axioms) can be downloaded
- working memory where all derivation steps are made

Interested in measuring

- size of proofs
- "complexity" of proofs

Size of a proof $\approx$ time of the computation

(Space) Complexity $\approx$ memory consumption of proof (how many things needed to remember simultaneously)

# Formal Definition of Sequential Proof (1/2)

**Definition (Derivation (Inspired by [Alekhnovich et al. '02]))**

A $\mathcal{P}$-configuration $\mathbb{D}$ is a set of lines $\{L_i\}$ (of correct syntactic form for $\mathcal{P}$)

A sequence of configurations $\{\mathbb{D}_0, \ldots, \mathbb{D}_\tau\}$ is a a $\mathcal{P}$-derivation from a CNF formula $F$ if $\mathbb{D} = \emptyset$ and for all $t \in [\tau]$, the set $\mathbb{D}_t$ is obtained from $\mathbb{D}_{t-1}$ by one of the following derivation steps:

*Axiom Download* $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L_C\}$, where $L_C$ is the encoding of a clause $C \in F$ (an axiom clause)

*Inference* $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L\}$ for $L$ inferred by one of the inference rules for $\mathcal{P}$ from $L_1, \ldots, L_m \in \mathbb{D}_{t-1}$

*Erasure* $\mathbb{D}_t = \mathbb{D}_{t-1} \setminus \{L\}$ for some $L \in \mathbb{D}_{t-1}$

# Formal Definition of Sequential Proof (2/2)

## Definition (Refutation)

A $\mathcal{P}$-refutation $\pi : F \vdash \bot$ of a CNF formula $F$ is

- a derivation $\pi = \{\mathbb{D}_0, \ldots, \mathbb{D}_\tau\}$ such that
- $\mathbb{D}_0 = \emptyset$ and
- $\bot \in \mathbb{D}_\tau$, where $\bot$ denotes (explicit) contradiction

## Definition (Tree-like refutation)

If every line $L$ in a refutation is used at most once before being erased (though it can possibly be rederived later), the refutation is tree-like

Looking at DAG $G_\pi$ with lines in $\pi$ as vertices and edges from the assumptions to the consequence for inferences, $G_\pi$ will be a tree

# Length and Space (Generic Definitions)

## Definition (Length)

Length $L(\pi)$ of refutation $\pi = \#$ derivation steps
($\approx \#$ lines counted with repetitions)

Length of refuting $F$ in $\mathcal{P}$
$L_{\mathcal{P}}(F \vdash \bot) =$ minimal length of any refutation

## Definition (Space)

Space $Sp(\pi)$ of refutation $\pi =$ "size" of largest configuration in $\pi$

Space of refuting $F$ in $\mathcal{P}$
$Sp_{\mathcal{P}}(F \vdash \bot) =$ minimal space of any refutation

These definitions to be made more precise for specific proof systems

# Resolution [Blake '37]

Lines in refutation are disjunctive clauses

Just one inference rule, the resolution rule:

$$\frac{B \vee x \quad C \vee \overline{x}}{B \vee C}$$

$B \vee C$ is the resolvent of $B \vee x$ and $C \vee \overline{x}$

**Observation**

*If $F$ is a satisfiable CNF formula and $D$ is derived from clauses $C_1, C_2 \in F$ by the resolution rule, then $F \wedge D$ is satisfiable.*

Prove $F$ unsatisfiable by deriving the unsatisfiable empty clause $\perp$ (the clause with no literals) from $F$ by resolution

# Resolution Sound and Complete

Resolution is sound and implicationally complete.

**Sound** If there is a resolution derivation $\pi : F \vdash A$ then $F \models A$

**Complete** If $F \models A$ then there is a resolution derivation $\pi : F \vdash A'$ for some $A' \subseteq A$.

In particular:

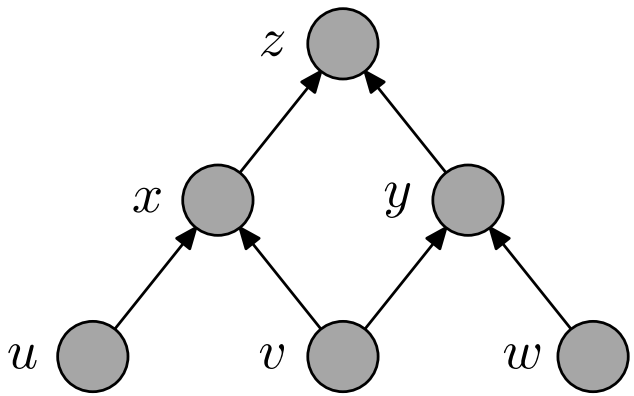
$$F \text{ is unsatisfiable} \Leftrightarrow \exists \text{ resolution refutation of } F$$

# Resolution as a Sequential Proof System

- Goal: Refute given CNF formula (i.e., prove it is unsatisfiable)
- Proof system operates with disjunctive clauses
- Proof/refutation is "presented on blackboard"
- Derivation steps:
  - Write down clauses of CNF formula being refuted (axiom clauses)
  - Infer new clauses by resolution rule
  - Erase clauses that are not currently needed (to save space on blackboard)
- Refutation ends when empty clause $\bot$ is derived

# Example CNF Formula

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

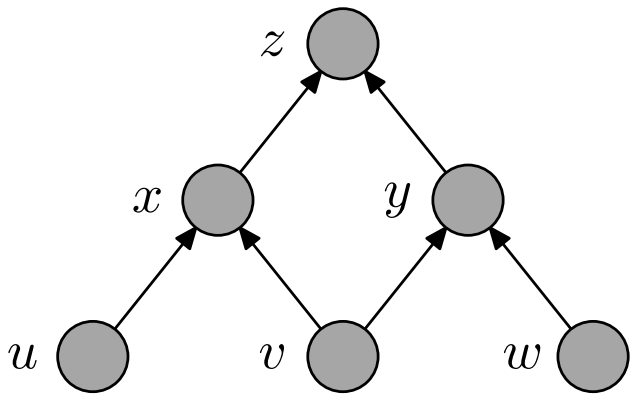



Defined in terms of directed acyclic graph (DAG):

- source vertices true
- truth propagates upwards
- but sink vertex is false

# Example CNF Formula

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

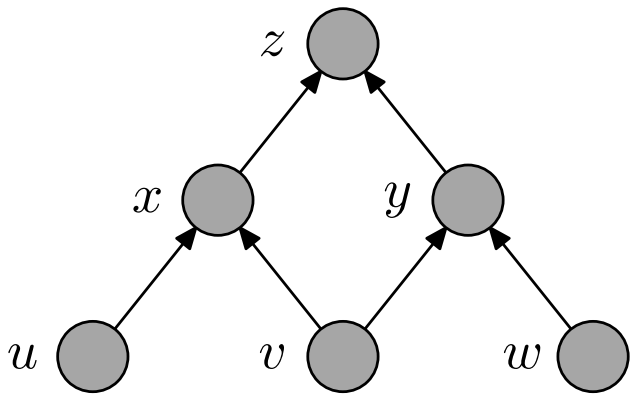



Defined in terms of directed acyclic graph (DAG):

- source vertices true
- truth propagates upwards
- but sink vertex is false

# Example CNF Formula

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

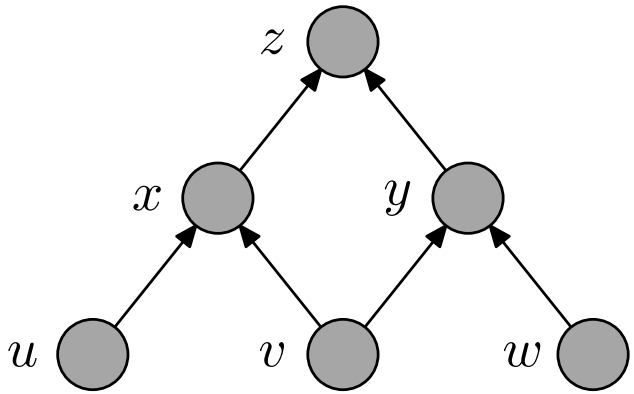



Defined in terms of directed acyclic graph (DAG):

- source vertices true
- truth propagates upwards
- but sink vertex is false

# Example CNF Formula

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

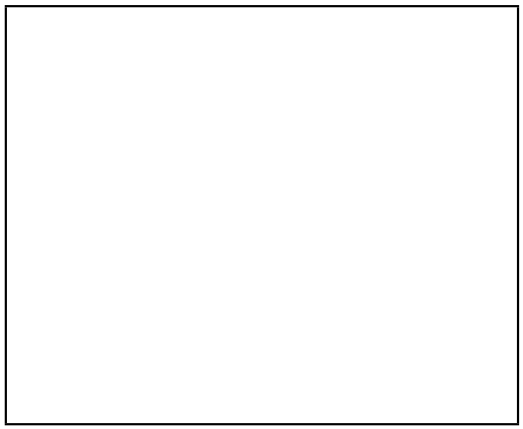



Defined in terms of directed acyclic graph (DAG):

- source vertices true
- truth propagates upwards
- but sink vertex is false

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$



| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 0 |
| max # lines on board | 0 |
| max # literals on board | 0 |

Can download axioms,
erase used clauses or
infer new clauses by resolution rule

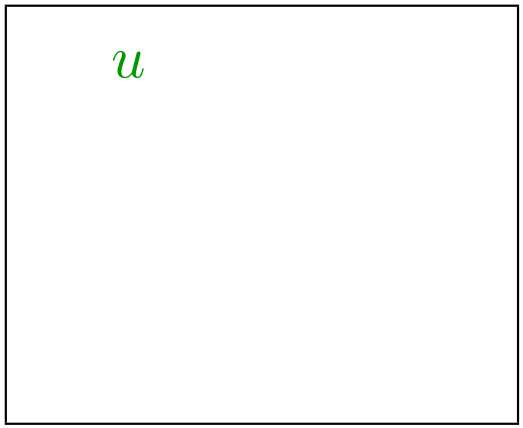
$$\frac{B \vee x \quad C \vee \overline{x}}{B \vee C}$$

(but only from clauses currently on the board!)

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 1 |
| max # lines on board | 1 |
| max # literals on board | 1 |

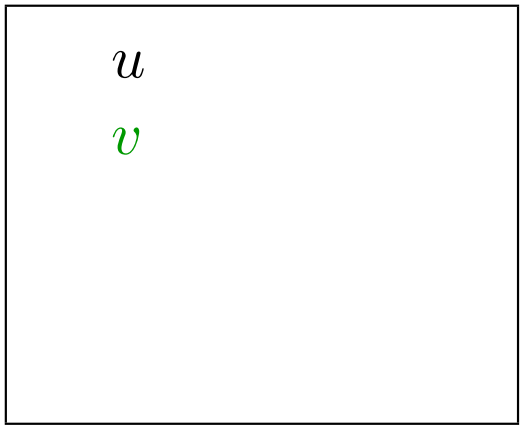



Download axiom 1: $u$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 2 |
| max # lines on board | 2 |
| max # literals on board | 2 |




Download axiom 1: $u$

Download axiom 2: $v$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 3 |
| max # lines on board | 3 |
| max # literals on board | 5 |

$u$

$v$

$\overline{u} \vee \overline{v} \vee x$

Download axiom 1: $u$

Download axiom 2: $v$

Download axiom 4: $\overline{u} \vee \overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 3 |
| max # lines on board | 3 |
| max # literals on board | 5 |

$u$

$v$

$\overline{u} \vee \overline{v} \vee x$

Download axiom 1: $u$
Download axiom 2: $v$
Download axiom 4: $\overline{u} \vee \overline{v} \vee x$
Infer $\overline{v} \vee x$ from
$u$ and $\overline{u} \vee \overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 4 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$u$

$v$

$\overline{u} \vee \overline{v} \vee x$

$\overline{v} \vee x$

Download axiom 1: $u$
Download axiom 2: $v$
Download axiom 4: $\overline{u} \vee \overline{v} \vee x$
Infer $\overline{v} \vee x$ from
$u$ and $\overline{u} \vee \overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 4 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$u$

$v$

$\overline{u} \vee \overline{v} \vee x$

$\overline{v} \vee x$

Download axiom 2: $v$
Download axiom 4: $\overline{u} \vee \overline{v} \vee x$
Infer $\overline{v} \vee x$ from
$u$ and $\overline{u} \vee \overline{v} \vee x$
Erase the clause $\overline{u} \vee \overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 4 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$u$

$v$

$\overline{v} \vee x$

Download axiom 2: $v$

Download axiom 4: $\overline{u} \vee \overline{v} \vee x$

Infer $\overline{v} \vee x$ from

$u$ and $\overline{u} \vee \overline{v} \vee x$

Erase the clause $\overline{u} \vee \overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 4 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$u$

$v$

$\overline{v} \vee x$

Download axiom 4: $\overline{u} \vee \overline{v} \vee x$

Infer $\overline{v} \vee x$ from

$u$ and $\overline{u} \vee \overline{v} \vee x$

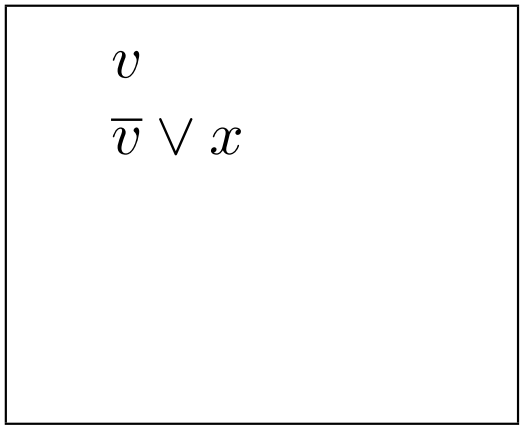
Erase the clause $\overline{u} \vee \overline{v} \vee x$

Erase the clause $u$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 4 |
| max # lines on board | 4 |
| max # literals on board | 7 |

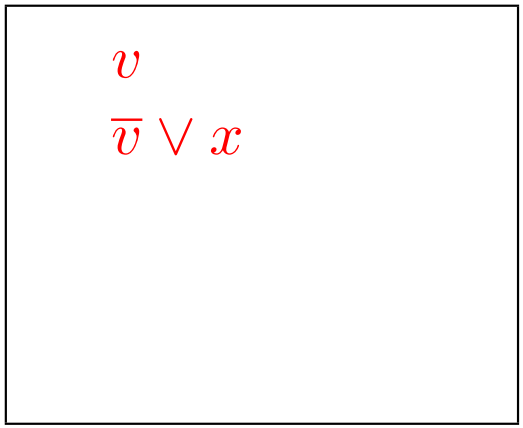



Download axiom 4: $\overline{u} \vee \overline{v} \vee x$
Infer $\overline{v} \vee x$ from
$u$ and $\overline{u} \vee \overline{v} \vee x$
Erase the clause $\overline{u} \vee \overline{v} \vee x$
Erase the clause $u$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 4 |
| max # lines on board | 4 |
| max # literals on board | 7 |




$u$ and $\overline{u} \vee \overline{v} \vee x$
Erase the clause $\overline{u} \vee \overline{v} \vee x$
Erase the clause $u$
Infer $x$ from
$v$ and $\overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 5 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$v$

$\overline{v} \vee x$

$x$

$u$ and $\overline{u} \vee \overline{v} \vee x$

Erase the clause $\overline{u} \vee \overline{v} \vee x$

Erase the clause $u$

Infer $x$ from

$v$ and $\overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 5 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$v$

$\overline{v} \vee x$

$x$

Erase the clause $\overline{u} \vee \overline{v} \vee x$

Erase the clause $u$

Infer $x$ from

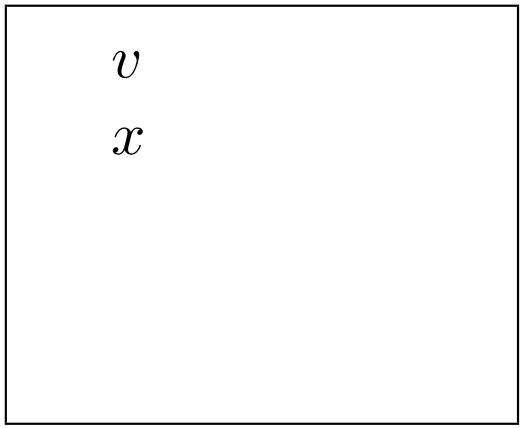
$v$ and $\overline{v} \vee x$

Erase the clause $\overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 5 |
| max # lines on board | 4 |
| max # literals on board | 7 |

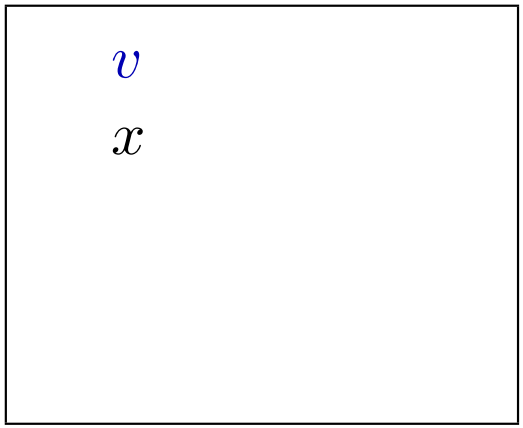



Erase the clause $\overline{u} \vee \overline{v} \vee x$
Erase the clause $u$
Infer $x$ from
$v$ and $\overline{v} \vee x$
Erase the clause $\overline{v} \vee x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 5 |
| max # lines on board | 4 |
| max # literals on board | 7 |

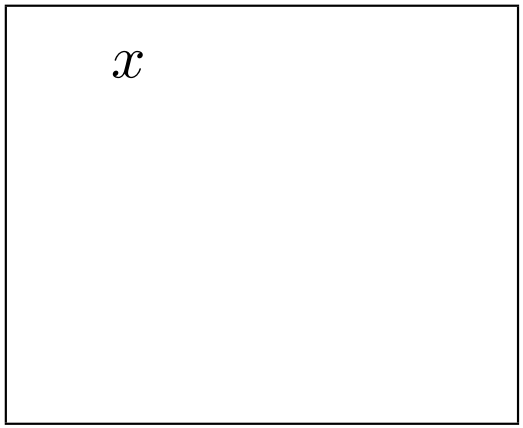



Erase the clause $u$
Infer $x$ from
$v$ and $\overline{v} \vee x$
Erase the clause $\overline{v} \vee x$
Erase the clause $v$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 5 |
| max # lines on board | 4 |
| max # literals on board | 7 |

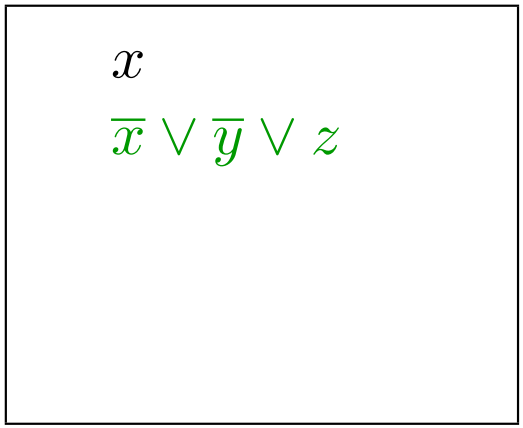



Erase the clause $u$
Infer $x$ from
$v$ and $\overline{v} \vee x$
Erase the clause $\overline{v} \vee x$
Erase the clause $v$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 6 |
| max # lines on board | 4 |
| max # literals on board | 7 |




Infer $x$ from

$v$ and $\overline{v} \vee x$

Erase the clause $\overline{v} \vee x$

Erase the clause $v$

Download axiom 6: $\overline{x} \vee \overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 6 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$x$

$\overline{x} \vee \overline{y} \vee z$

Erase the clause $\overline{v} \vee x$

Erase the clause $v$

Download axiom 6: $\overline{x} \vee \overline{y} \vee z$

Infer $\overline{y} \vee z$ from

$x$ and $\overline{x} \vee \overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 7 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$x$

$\overline{x} \vee \overline{y} \vee z$

$\overline{y} \vee z$

Erase the clause $\overline{v} \vee x$

Erase the clause $v$

Download axiom 6: $\overline{x} \vee \overline{y} \vee z$

Infer $\overline{y} \vee z$ from

$x$ and $\overline{x} \vee \overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 7 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$x$

$\overline{x} \vee \overline{y} \vee z$

$\overline{y} \vee z$

Erase the clause $v$

Download axiom 6: $\overline{x} \vee \overline{y} \vee z$

Infer $\overline{y} \vee z$ from

$x$ and $\overline{x} \vee \overline{y} \vee z$

Erase the clause $\overline{x} \vee \overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 7 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$x$

$\overline{y} \vee z$

Erase the clause $v$

Download axiom 6: $\overline{x} \vee \overline{y} \vee z$

Infer $\overline{y} \vee z$ from

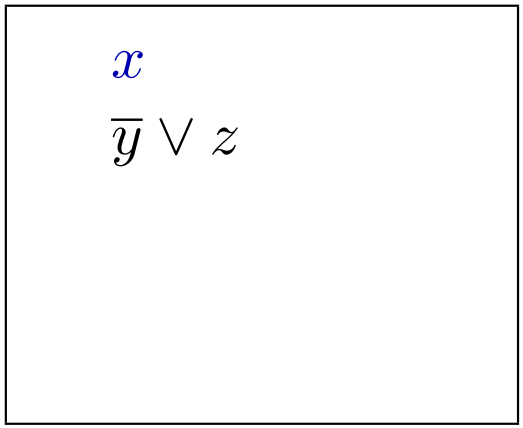
$x$ and $\overline{x} \vee \overline{y} \vee z$

Erase the clause $\overline{x} \vee \overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 7 |
| max # lines on board | 4 |
| max # literals on board | 7 |

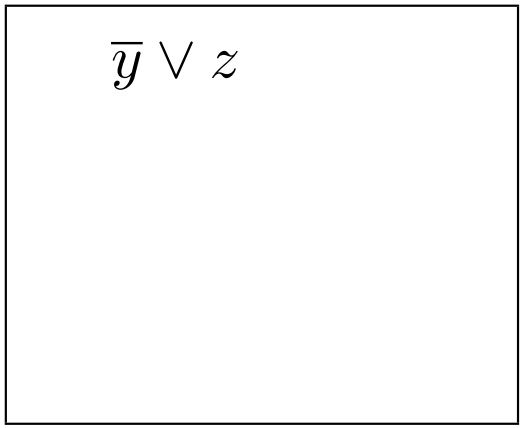



Download axiom 6: $\overline{x} \vee \overline{y} \vee z$
Infer $\overline{y} \vee z$ from
$x$ and $\overline{x} \vee \overline{y} \vee z$
Erase the clause $\overline{x} \vee \overline{y} \vee z$
Erase the clause $x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 7 |
| max # lines on board | 4 |
| max # literals on board | 7 |




Download axiom 6: $\overline{x} \vee \overline{y} \vee z$
Infer $\overline{y} \vee z$ from
$x$ and $\overline{x} \vee \overline{y} \vee z$
Erase the clause $\overline{x} \vee \overline{y} \vee z$
Erase the clause $x$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 8 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$\overline{y} \vee z$

$\overline{v} \vee \overline{w} \vee y$

Infer $\overline{y} \vee z$ from

$x$ and $\overline{x} \vee \overline{y} \vee z$

Erase the clause $\overline{x} \vee \overline{y} \vee z$

Erase the clause $x$

Download axiom 5: $\overline{v} \vee \overline{w} \vee y$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 8 |
| max # lines on board | 4 |
| max # literals on board | 7 |

$\overline{y} \vee z$

$\overline{v} \vee \overline{w} \vee y$

Erase the clause $\overline{x} \vee \overline{y} \vee z$

Erase the clause $x$

Download axiom 5: $\overline{v} \vee \overline{w} \vee y$

Infer $\overline{v} \vee \overline{w} \vee z$ from

$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 9 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{y} \vee z$
$\overline{v} \vee \overline{w} \vee y$
$\overline{v} \vee \overline{w} \vee z$

Erase the clause $\overline{x} \vee \overline{y} \vee z$
Erase the clause $x$
Download axiom 5: $\overline{v} \vee \overline{w} \vee y$
Infer $\overline{v} \vee \overline{w} \vee z$ from
$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 9 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{y} \vee z$

$\overline{v} \vee \overline{w} \vee y$

$\overline{v} \vee \overline{w} \vee z$

Erase the clause $x$

Download axiom 5: $\overline{v} \vee \overline{w} \vee y$

Infer $\overline{v} \vee \overline{w} \vee z$ from

$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$

Erase the clause $\overline{v} \vee \overline{w} \vee y$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 9 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{y} \vee z$

$\overline{v} \vee \overline{w} \vee z$

Erase the clause $x$

Download axiom 5: $\overline{v} \vee \overline{w} \vee y$

Infer $\overline{v} \vee \overline{w} \vee z$ from

$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$

Erase the clause $\overline{v} \vee \overline{w} \vee y$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 9 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{y} \vee z$

$\overline{v} \vee \overline{w} \vee z$

Download axiom 5: $\overline{v} \vee \overline{w} \vee y$

Infer $\overline{v} \vee \overline{w} \vee z$ from

$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$

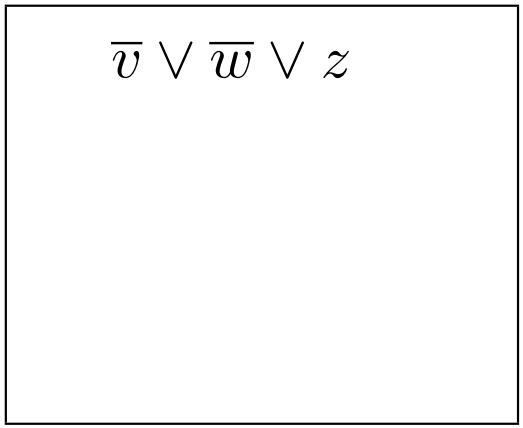
Erase the clause $\overline{v} \vee \overline{w} \vee y$

Erase the clause $\overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 9 |
| max # lines on board | 4 |
| max # literals on board | 8 |




Download axiom 5: $\overline{v} \vee \overline{w} \vee y$
Infer $\overline{v} \vee \overline{w} \vee z$ from
$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$
Erase the clause $\overline{v} \vee \overline{w} \vee y$
Erase the clause $\overline{y} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 10 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$v$

Infer $\overline{v} \vee \overline{w} \vee z$ from
$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$
Erase the clause $\overline{v} \vee \overline{w} \vee y$
Erase the clause $\overline{y} \vee z$
Download axiom 2: $v$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 11 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$v$

$w$

$\overline{y} \vee z$ and $\overline{v} \vee \overline{w} \vee y$

Erase the clause $\overline{v} \vee \overline{w} \vee y$

Erase the clause $\overline{y} \vee z$

Download axiom 2: $v$

Download axiom 3: $w$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 12 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$v$

$w$

$\overline{z}$

Erase the clause $\overline{v} \vee \overline{w} \vee y$
Erase the clause $\overline{y} \vee z$
Download axiom 2: $v$
Download axiom 3: $w$
Download axiom 7: $\overline{z}$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 12 |
| max # lines on board | 4 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$v$

$w$

$\overline{z}$

Download axiom 2: $v$
Download axiom 3: $w$
Download axiom 7: $\overline{z}$
Infer $\overline{w} \vee z$ from
$v$ and $\overline{v} \vee \overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 13 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$v$

$w$

$\overline{z}$

$\overline{w} \vee z$

Download axiom 2: $v$
Download axiom 3: $w$
Download axiom 7: $\overline{z}$
Infer $\overline{w} \vee z$ from
$v$ and $\overline{v} \vee \overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 13 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$v$

$w$

$\overline{z}$

$\overline{w} \vee z$

Download axiom 3: $w$
Download axiom 7: $\overline{z}$
Infer $\overline{w} \vee z$ from
$v$ and $\overline{v} \vee \overline{w} \vee z$
Erase the clause $v$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 13 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$w$

$\overline{z}$

$\overline{w} \vee z$

Download axiom 3: $w$
Download axiom 7: $\overline{z}$
Infer $\overline{w} \vee z$ from
$v$ and $\overline{v} \vee \overline{w} \vee z$
Erase the clause $v$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 13 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{v} \vee \overline{w} \vee z$

$w$

$\overline{z}$

$\overline{w} \vee z$

Download axiom 7: $\overline{z}$

Infer $\overline{w} \vee z$ from

$v$ and $\overline{v} \vee \overline{w} \vee z$

Erase the clause $v$

Erase the clause $\overline{v} \vee \overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 13 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$w$

$\overline{z}$

$\overline{w} \vee z$

Download axiom 7: $\overline{z}$

Infer $\overline{w} \vee z$ from

$v$ and $\overline{v} \vee \overline{w} \vee z$

Erase the clause $v$

Erase the clause $\overline{v} \vee \overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 13 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$w$

$\overline{z}$

$\overline{w} \vee z$

$v$ and $\overline{v} \vee \overline{w} \vee z$

Erase the clause $v$

Erase the clause $\overline{v} \vee \overline{w} \vee z$

Infer $z$ from

$w$ and $\overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 14 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$w$

$\overline{z}$

$\overline{w} \vee z$

$z$

$v$ and $\overline{v} \vee \overline{w} \vee z$

Erase the clause $v$

Erase the clause $\overline{v} \vee \overline{w} \vee z$

Infer $z$ from

$w$ and $\overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 14 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$w$
$\overline{z}$
$\overline{w} \vee z$
$z$

Erase the clause $v$
Erase the clause $\overline{v} \vee \overline{w} \vee z$
Infer $z$ from
$w$ and $\overline{w} \vee z$
Erase the clause $w$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 14 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{z}$

$\overline{w} \vee z$

$z$

Erase the clause $v$

Erase the clause $\overline{v} \vee \overline{w} \vee z$

Infer $z$ from

$w$ and $\overline{w} \vee z$

Erase the clause $w$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 14 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{z}$

$\overline{w} \vee z$

$z$

Erase the clause $\overline{v} \vee \overline{w} \vee z$

Infer $z$ from

$w$ and $\overline{w} \vee z$

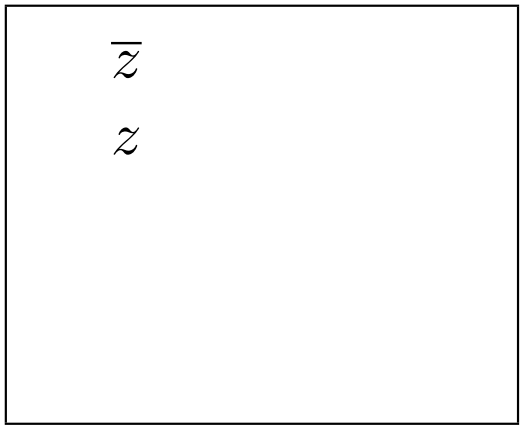
Erase the clause $w$

Erase the clause $\overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 14 |
| max # lines on board | 5 |
| max # literals on board | 8 |

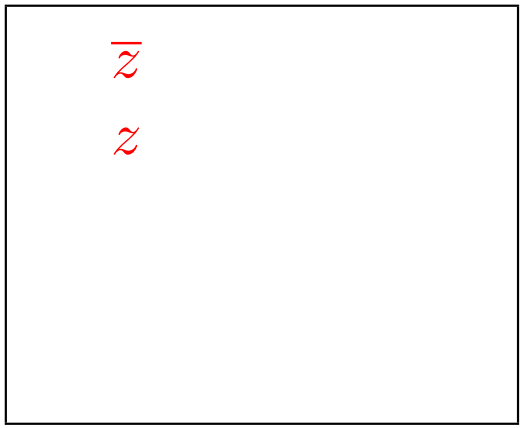



Erase the clause $\overline{v} \vee \overline{w} \vee z$
Infer $z$ from
$w$ and $\overline{w} \vee z$
Erase the clause $w$
Erase the clause $\overline{w} \vee z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 14 |
| max # lines on board | 5 |
| max # literals on board | 8 |




$w$ and $\overline{w} \vee z$
Erase the clause $w$
Erase the clause $\overline{w} \vee z$
Infer 0 from
$\overline{z}$ and $z$

# Example Resolution Refutation

1. $u$
2. $v$
3. $w$
4. $\overline{u} \vee \overline{v} \vee x$
5. $\overline{v} \vee \overline{w} \vee y$
6. $\overline{x} \vee \overline{y} \vee z$
7. $\overline{z}$

| **Blackboard bookkeeping** | |
|---|---|
| total # clauses on board | 15 |
| max # lines on board | 5 |
| max # literals on board | 8 |

$\overline{z}$

$z$

0

$w$ and $\overline{w} \vee z$

Erase the clause $w$

Erase the clause $\overline{w} \vee z$

Infer 0 from

$\overline{z}$ and $z$

# Complexity Measures of Interest: Length and Space

- Length: Lower bound on time for proof search algorithm
- Space: Lower bound on memory for proof search algorithm

**Length**

# clauses written on blackboard counted with repetitions

**Space**

Somewhat less straightforward — several ways of measuring




Clause space: 3

Total space: 6

# Complexity Measures of Interest: Length and Space

- Length: Lower bound on time for proof search algorithm
- Space: Lower bound on memory for proof search algorithm

**Length**

# clauses written on blackboard counted with repetitions

**Space**

Somewhat less straightforward — several ways of measuring




Clause space: 3

Total space: 6

# Complexity Measures of Interest: Length and Space

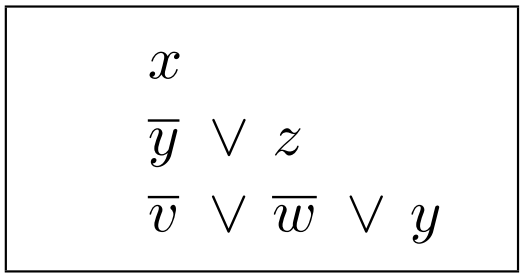
- Length: Lower bound on time for proof search algorithm
- Space: Lower bound on memory for proof search algorithm

**Length**

# clauses written on blackboard counted with repetitions

**Space**

Somewhat less straightforward — several ways of measuring




Clause space: 3

Total space: 6

# Complexity Measures of Interest: Length and Space

- Length: Lower bound on time for proof search algorithm
- Space: Lower bound on memory for proof search algorithm

**Length**

# clauses written on blackboard counted with repetitions

**Space**

Somewhat less straightforward — several ways of measuring

1. $x$
2. $\overline{y} \vee z$
3. $\overline{v} \vee \overline{w} \vee y$

Clause space: 3

Total space: 6

# Complexity Measures of Interest: Length and Space

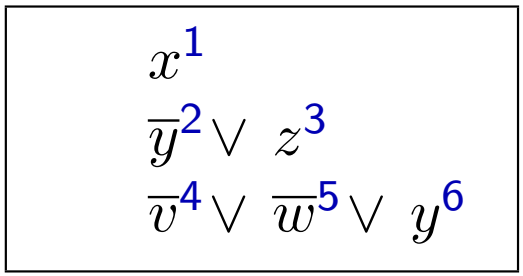
- Length: Lower bound on time for proof search algorithm
- Space: Lower bound on memory for proof search algorithm

**Length**

# clauses written on blackboard counted with repetitions

**Space**

Somewhat less straightforward — several ways of measuring




Clause space: 3

Total space: 6

# Complexity Measures of Interest: Length and Space

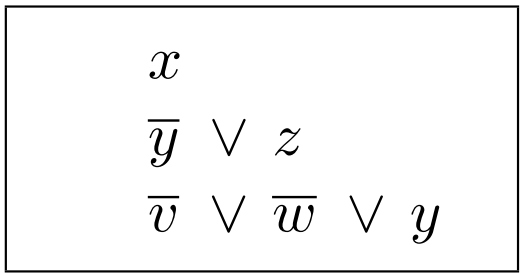
- Length: Lower bound on time for proof search algorithm
- Space: Lower bound on memory for proof search algorithm

**Length**

# clauses written on blackboard counted with repetitions
(in our example resolution refutation 15)

**Space**

Somewhat less straightforward — several ways of measuring




Clause space: 3
(in our refutation 5)
Total space: 6
(in our refutation 8)

# $k$-DNF Resolution

Family of proof systems $\mathcal{R}(k)$ parameterized by $k \in \mathbb{N}^+$
($\mathcal{R}(1)$ is resolution)

Lines in $k$-DNF-resolution refutation are $k$-DNF formulas
i.e., disjunctions of conjunctions (terms) of size $\leq k$

Inference rules as follows (where $G, H$ denote $k$-DNF formulas, $T, T'$ denote $k$-terms, and $a_1, \ldots, a_k$ denote literals):

*$k$-cut* $\dfrac{(a_1 \wedge \cdots \wedge a_{k'}) \vee G \qquad \overline{a}_1 \vee \cdots \vee \overline{a}_{k'} \vee H}{G \vee H}$, $(k' \leq k)$

*$\wedge$-introduction* $\dfrac{G \vee T \qquad G \vee T'}{G \vee (T \wedge T')}$, as long as $|T \cup T'| \leq k$.

*$\wedge$-elimination* $\dfrac{G \vee T}{G \vee T'}$ for any $T' \subseteq T$.

*Weakening* $\dfrac{G}{G \vee H}$ for any $k$-DNF formula $H$.

# $k$-DNF Resolution Measures

**Length**

# derivation steps

($\approx$ # $k$-DNF formulas counted with repetitions)

**Formula space**

# $k$-DNF formulas in any configuration

(Analogue of clause space)

**Total space**

Total # literals in configuration counted with repetitions

# Cutting Planes: Informal Description

- Geometric proof system introduced by [Cook, Coullard & Turán '87]
- Translate clauses to linear inequalities for real variables in $[0, 1]$
- For instance, $x \vee y \vee \overline{z}$ gets translated to $x + y + (1 - z) \geq 1$, i.e., $x + y - z \geq 0$
- Manipulate linear inequalities to derive contradiction $0 \geq 1$

# Cutting Planes: Inference Rules

Lines in cutting planes (CP) refutation: are linear inequalities with integer coefficients.

Derivation rules:

*Variable axioms* $\dfrac{}{x \geq 0}$ and $\dfrac{}{-x \geq -1}$ for all variables $x$

*Addition* $\dfrac{\sum a_i x_i \geq A \qquad \sum b_i x_i \geq B}{\sum (a_i + b_i) x_i \geq A + B}$

*Multiplication* $\dfrac{\sum a_i x_i \geq A}{\sum c a_i x_i \geq cA}$ for a positive integer $c$

*Division* $\dfrac{\sum c a_i x_i \geq A}{\sum a_i x_i \geq \lceil A/c \rceil}$ for a positive integer $c$

A CP-refutation ends when the inequality $0 \geq 1$ has been derived

# Cutting Planes Measures

**Length**
# derivation steps

**Size**
# symbols needed to represent proof (coefficients can be huge)

**Line space**
# Linear inequalities in any configuration
(Analogue of clause space)

**Total space**
Total # variables in configuration counted with repetitions
+ log of coefficients

# Polynomial Calculus

- Algrebraic system introduced by [Clegg, Edmonds & Impagliazzo '96] under the name of "Gröbner proof system"
- Clauses are interpreted as multilinear polynomial equations
- Here, natural to flip convention and think of 0 as true and 1 as false
- For instance, clause $x \vee y \vee \overline{z}$ gets translated to $xy(1-z) = 0$ or $xy - xyz = 0$
- Derive contradiction by showing that there is no common root for the polynomial equations corresponding to all the clauses

# Polynomial Calculus: Inference Rules

Lines in polynomial calculus (PC) refutation: multivariate polynomial equations $p = 0$, where $p \in \mathbb{F}[x, y, z, \ldots]$ for some (fixed) field $\mathbb{F}$, typically finite

Customary to omit "$= 0$" and only write $p$

The derivation rules are as follows, where $\alpha, \beta \in \mathbb{F}$, $p, q \in \mathbb{F}[x, y, z, \ldots]$, and $x$ is any variable:

*Variable axioms* $\dfrac{}{x^2 - x}$ for all variables $x$ (forcing 0/1-solutions)

*Linear combination* $\dfrac{p \qquad q}{\alpha p + \beta q}$

*Multiplication* $\dfrac{p}{xp}$

A PC-refutation ends when 1 has been derived (i.e., $1 = 0$)

(Note that multilinearity follows w.l.o.g. from $x^2 = x$ for all variables $x$)

# Polynomial Calculus: Alternate View

Can also (equivalently) consider a PC-refutation to be a calculation in the ideal generated by polynomials corresponding to clauses

Then a refutation concludes by proving that 1 is in this ideal, i.e., that the ideal is everything

Clearly implies that there is no common root

Less obvious: if there is no common root, then 1 is always in the ideal (requires some algebra)

# Polynomial Calculus Measures

**Size**
Total # monomials in the refutation counted with repetitions

**Length**
# derivation steps
($\approx$ # polynomial equations counted with repetitions)

**(Monomial) space**
Maximal # monomials in any configuration counted with repetitions
(Again an analogue of clause space)

**Total space**
Total # variables in any configuration counted with repetitions

# Main Focus of Course

Look at ($k$-DNF) resolution, cutting planes and polynomial calculus

- Relatively weak proof systems, so there is chance to understand them
- Also, because of this they can be (and are) used for SAT solving (as opposed to stronger systems)

Want to understand these systems and prove upper and lower bounds on

- length
- space
- length-space trade-offs

Use this understanding to say something about potential and limitations of SAT solving

Turns out that other measure are very helpful to increase understanding

- width for resolution
- degree for polynomial calculus

# State of the Art

- Resolution (and $k$-DNF resolution): much known
- Polynomial calculus: some known
- Cutting planes: very poorly understood

Lots of good open questions for all three systems

# Practicalities

- Official course code is DD3501 (FDD3501 for PhD students)
- Check out `www.csc.kth.se/~jakobn/teaching/proofcplx11` regularly for info
- Note irregular schedule (due to travelling) and changing seminar rooms
- Aiming at a total of 10 + 10 lectures (autumn + spring)
- Examination: problem sets + possibly scribed lecture notes (to be decided jointly)
- Possibility to substitute a reading or research project for parts of the other requirements
- Office hours by appointment only (or whenever I am in the fika room)
- Course intended to be fun and interesting (and challenging)
  Need feed-back to make that happen — let me know what you think

# Questions to Discuss

- Decide on the lecture schedule for period 3 right away or wait?
- Scribing lecture notes or not?
- How much connections to actual, practical SAT solving do we want?
- Would we want SAT-related guest lectures even if they don't fit our regular schedule?
- Any other questions or concerns?