# In Between Resolution and Cutting Planes
# Proof Systems for Pseudo-Boolean SAT Solving

Marc Vinyals

KTH Royal Institute of Technology
Stockholm, Sweden

Joint work with Jan Elffers, Jesús Giráldez-Cru, Stephan Gocht, and Jakob Nordström

Pragmatics of Constraint Reasoning Workshop
August 28 2017, Melbourne, Australia

# What do we do

Study pseudo-Boolean solvers from proof complexity point of view

**Question**

How powerful are pseudo-Boolean solvers?

Build two kinds of formulas

- solvers can perform well with good heuristics
- solvers do not exploit power of pseudo-Boolean constraints

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z} \quad \overline{x} \lor y \quad \overline{x} \lor \overline{y}$

**Database**

**Assignment**

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

## Database

## Assignment

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

**Database**

**Assignment**

$x \overset{\mathsf{d}}{=} 0$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

## Database

## Assignment

$x \stackrel{\mathsf{d}}{=} 0$ $y \stackrel{x \vee y}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z} \quad \overline{x} \lor y \quad \overline{x} \lor \overline{y}$

## Database

## Assignment

$x \overset{\mathsf{d}}{=} 0 \quad y \overset{x \lor y}{=} 1 \quad z \overset{x \lor \overline{y} \lor z}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

**Database**

**Assignment**

$x \overset{\mathsf{d}}{=} 0$ $y \overset{x \vee y}{=} 1$ $z \overset{x \vee \overline{y} \vee z}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z} \quad \overline{x} \lor y \quad \overline{x} \lor \overline{y}$

## Database

$x$

## Assignment

$x \stackrel{\mathsf{d}}{=} 0 \quad y \stackrel{x \lor y}{=} 1 \quad z \stackrel{x \lor \overline{y} \lor z}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

**Database**

$x$

**Assignment**

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z} \quad \overline{x} \lor y \quad \overline{x} \lor \overline{y}$

## Database

$x$

## Assignment

$x \stackrel{x}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z} \quad \overline{x} \lor y \quad \overline{x} \lor \overline{y}$

## Database

$x$

## Assignment

$x \overset{x}{=} 1 \quad y \overset{\overline{x} \lor y}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

**Database**

$x$

**Assignment**

$x \stackrel{x}{=} 1 \quad y \stackrel{\overline{x} \vee y}{=} 1$

# The CDCL Algorithm

```
while not solved :
  unit propagate
  if conflict :
    learn
    backtrack
  else :
    decide variable
```

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

## Database

$x \quad \perp$

## Assignment

$x \stackrel{x}{=} 1 \quad y \stackrel{\overline{x} \vee y}{=} 1$

# Conflict Analysis

- Say there is a conflict with variable $z$

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z}$

**Assignment $\rho$**

$x \overset{\mathsf{d}}{=} 0 \quad y \overset{x \lor y}{=} 1 \quad z \overset{x \lor \overline{y} \lor z}{=} 1$

# Conflict Analysis

- Say there is a conflict with variable $z$
- Some clause $C \vee \overline{z}$ caused the conflict

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

**Assignment $\rho$**

$x \overset{\mathsf{d}}{=} 0 \quad y \overset{x \vee y}{=} 1 \quad z \overset{x \vee \overline{y} \vee z}{=} 1$

# Conflict Analysis

- Say there is a conflict with variable $z$
- Some clause $C \vee \overline{z}$ caused the conflict
- Another clause $D \vee z$ propagated $z$

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

**Assignment $\rho$**

$x \overset{\mathsf{d}}{=} 0 \quad y \overset{x \vee y}{=} 1 \quad z \overset{x \vee \overline{y} \vee z}{=} 1$

# Conflict Analysis

- Say there is a conflict with variable $z$
- Some clause $C \vee \overline{z}$ caused the conflict
- Another clause $D \vee z$ propagated $z$
- Use resolution rule to derive $C \vee D$.

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

**Assignment $\rho$**

$x \overset{\mathsf{d}}{=} 0 \quad y \overset{x \vee y}{=} 1 \quad z \overset{x \vee \overline{y} \vee z}{=} 1$

**Resolution**

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

# Conflict Analysis

- Say there is a conflict with variable $z$
- Some clause $C \vee \overline{z}$ caused the conflict
- Another clause $D \vee z$ propagated $z$
- Use resolution rule to derive $C \vee D$.
- Remove $z$ from assignment.

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

**Assignment $\rho \setminus \{z\}$**

$x \stackrel{\mathsf{d}}{=} 0 \quad y \stackrel{x \vee y}{=} 1$

**Resolution**

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

# Conflict Analysis

- Say there is a conflict with variable $z$
- Some clause $C \lor \overline{z}$ caused the conflict
- Another clause $D \lor z$ propagated $z$
- Use resolution rule to derive $C \lor D$.
- Remove $z$ from assignment.
- $\rho$ falsifies $C$, $\rho$ falsifies $D \Rightarrow$ $\rho \setminus \{z\}$ falsifies $C \lor D$.

$x \lor y \quad x \lor \overline{y} \lor z \quad x \lor \overline{y} \lor \overline{z}$

**Assignment $\rho \setminus \{z\}$**

$x \stackrel{\text{d}}{=} 0 \quad y \stackrel{x \lor y}{=} 1$

**Resolution**

$$\frac{x \lor \overline{y} \lor z \qquad x \lor \overline{y} \lor \overline{z}}{x \lor \overline{y}}$$

# Conflict Analysis

- Say there is a conflict with variable $z$
- Some clause $C \vee \overline{z}$ caused the conflict
- Another clause $D \vee z$ propagated $z$
- Use resolution rule to derive $C \vee D$.
- Remove $z$ from assignment.
- $\rho$ falsifies $C$, $\rho$ falsifies $D \Rightarrow$ $\rho \setminus \{z\}$ falsifies $C \vee D$.
- Repeat until there is no reason for propagation.

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

**Assignment $\rho \setminus \{z\}$**

$x \overset{\mathsf{d}}{=} 0 \quad y \overset{x \vee y}{=} 1$

**Resolution**

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

# The Power of CDCL Solvers

All CDCL proofs are resolution proofs

Lower bound for resolution length $\Rightarrow$ lower bound for CDCL run time

*(Ignoring preprocessing)

# The Power of CDCL Solvers

All CDCL proofs are resolution proofs
Lower bound for resolution length $\Rightarrow$ lower bound for CDCL run time

*(Ignoring preprocessing)

And the opposite direction?

**Theorem** [Pipatsrisawat, Darwiche '09; Atserias, Fichte, Thurley '09]

CDCL $\equiv$ Resolution

- CDCL can simulate any resolution proof
- Assumes optimal decision and erasure heuristics

# More Powerful Solvers

Resolution is a weak proof system

- e.g. cannot count
- $x_1 + \cdots + x_n = n/2$ needs exponentially many clauses

# More Powerful Solvers

Resolution is a weak proof system

- e.g. cannot count
- $x_1 + \cdots + x_n = n/2$ needs exponentially many clauses

Pseudo-Boolean constraints more expressive

$$x_1 + \cdots + x_n \geq n/2$$
$$\overline{x_1} + \cdots + \overline{x_n} \geq n/2$$

Build solvers with pseudo-Boolean constraints?

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

- Several variables can propagate in one go

$2x + y + z \geq 2$

**Assignment**

$x \stackrel{\mathsf{d}}{=} 0$

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

- Several variables can propagate in one go

$$2x + y + z \geq 2$$

**Assignment**

$$x \stackrel{\mathsf{d}}{=} 0 \quad y \stackrel{2x+y+z\geq 2}{=} 1 \quad z \stackrel{2x+y+z\geq 2}{=} 1$$

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

- Several variables can propagate in one go
- Derived constraint not always falsified by assignment

$$x_1 + 2\overline{x_3} + x_4 + 2x_6 \geq 2 \qquad x_2 + x_5 + 2\overline{x_6} \geq 2$$

**Assignment**

$$x_1 \stackrel{\mathsf{d}}{=} 0 \quad x_2 \stackrel{\mathsf{d}}{=} 0 \quad x_3 \stackrel{\mathsf{d}}{=} 1$$

**Database**

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

- Several variables can propagate in one go
- Derived constraint not always falsified by assignment

$$x_1 + 2\overline{x_3} + x_4 + 2x_6 \geq 2 \qquad x_2 + x_5 + 2\overline{x_6} \geq 2$$

**Assignment**

$$x_1 \overset{\mathsf{d}}{=} 0 \quad x_2 \overset{\mathsf{d}}{=} 0 \quad x_3 \overset{\mathsf{d}}{=} 1 \quad x_6 \overset{x_1+2\overline{x_3}+x_4+2x_6 \geq 2}{=} 1$$

**Database**

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

- Several variables can propagate in one go
- Derived constraint not always falsified by assignment

$$x_1 + 2\overline{x_3} + x_4 + 2x_6 \geq 2 \qquad x_2 + x_5 + 2\overline{x_6} \geq 2$$

**Assignment**

$x_1 \overset{\mathsf{d}}{=} 0 \quad x_2 \overset{\mathsf{d}}{=} 0 \quad x_3 \overset{\mathsf{d}}{=} 1$

**Database**

$$x_1 + x_2 + 2\overline{x_3} + x_4 + x_5 \geq 2$$

# Pseudo-Boolean CDCL

CDCL with pseudo-Boolean constraints is tricky

- Several variables can propagate in one go
- Derived constraint not always falsified by assignment

Yet all of this can be fixed

# Cutting Planes

All pseudo-Boolean proofs are cutting planes proofs

# Cutting Planes

All pseudo-Boolean proofs are cutting planes proofs

Work with linear pseudo-Boolean inequalities

$$x \vee \overline{y} \quad \rightarrow \quad x + \overline{y} \geq 1 \quad \equiv \quad x + (1 - y) \geq 1$$

# Cutting Planes

All pseudo-Boolean proofs are cutting planes proofs

Work with linear pseudo-Boolean inequalities

$$x \vee \overline{y} \quad \rightarrow \quad x + \overline{y} \geq 1 \quad \equiv \quad x + (1 - y) \geq 1$$

Rules

Variable axioms

$$\frac{}{x \geq 0} \qquad \frac{}{-x \geq -1}$$

Addition

$$\frac{\sum a_i x_i \geq a \qquad \sum b_i x_i \geq b}{\sum(\alpha a_i + \beta b_i) x_i \geq \alpha a + \beta b}$$

Division

$$\frac{\sum a_i x_i \geq a}{\sum(a_i / k) x_i \geq \lceil a/k \rceil}$$

# Cutting Planes

All pseudo-Boolean proofs are cutting planes proofs

Work with linear pseudo-Boolean inequalities

$$x \vee \overline{y} \quad \rightarrow \quad x + \overline{y} \geq 1 \quad \equiv \quad x + (1 - y) \geq 1$$

Rules

Variable axioms

$$\frac{}{x \geq 0} \qquad \frac{}{-x \geq -1}$$

Addition

$$\frac{\sum a_i x_i \geq a \qquad \sum b_i x_i \geq b}{\sum (\alpha a_i + \beta b_i) x_i \geq \alpha a + \beta b}$$

Division

$$\frac{\sum a_i x_i \geq a}{\sum (a_i / k) x_i \geq \lceil a/k \rceil}$$

Goal: derive $0 \geq 1$

# Addition in Practice

## Addition

$$\frac{\sum a_i x_i \geq a \qquad \sum b_i x_i \geq b}{\sum (\alpha a_i + \beta b_i) x_i \geq \alpha a + \beta b}$$

- Unbounded choices
- Need a reason to add inequalities:
  - One conflicting variable
  - Conflict disappears after addition

# Addition in Practice

## Addition

$$\frac{\sum a_i x_i \geq a \qquad \sum b_i x_i \geq b}{\sum (\alpha a_i + \beta b_i) x_i \geq \alpha a + \beta b}$$

- Unbounded choices
- Need a reason to add inequalities:
  - One conflicting variable
  - Conflict disappears after addition

## Cancelling Addition

Some variable cancels: $\alpha a_i + \beta b_i = 0$

# Division in Practice

## Division

$$\frac{\sum a_i x_i \geq a}{\sum (a_i/k) x_i \geq \lceil a/k \rceil}$$

- Too expensive

# Division in Practice

## Division

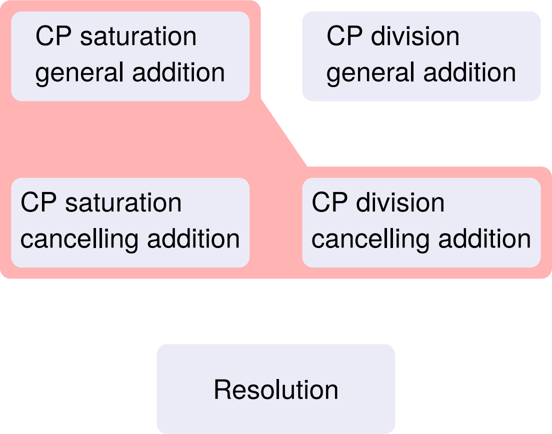
$$\frac{\sum a_i x_i \geq a}{\sum (a_i/k) x_i \geq \lceil a/k \rceil}$$

- Too expensive

## Saturation

$$\frac{\sum a_i x_i \geq a}{\sum \min(a, a_i) x_i \geq a}$$

# Proof Systems




Power of subsystems of CP?

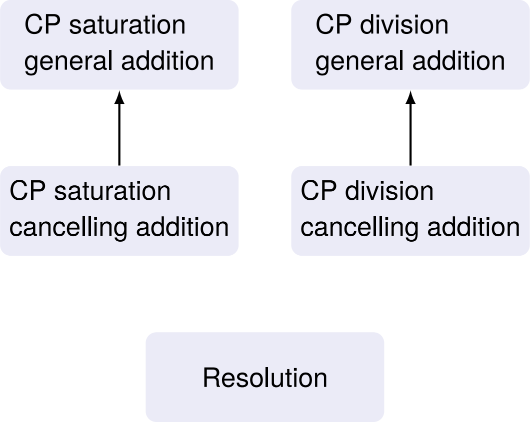
# Results

**Theorem**

On CNF inputs all subsystems as weak as resolution

- No subsystem is implicationally complete
- Solver becomes very sensitive to the encoding

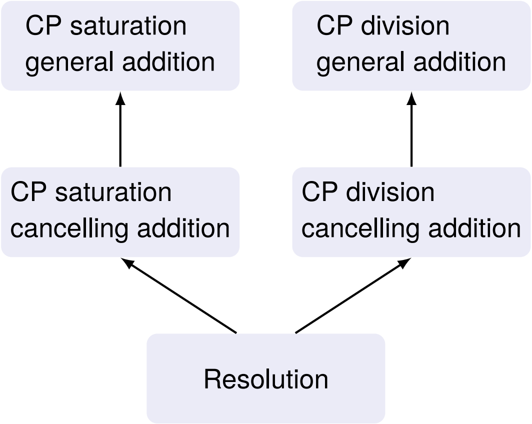
# Proof Systems




Cancelling addition is a particular case of addition

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)

# Proof Systems

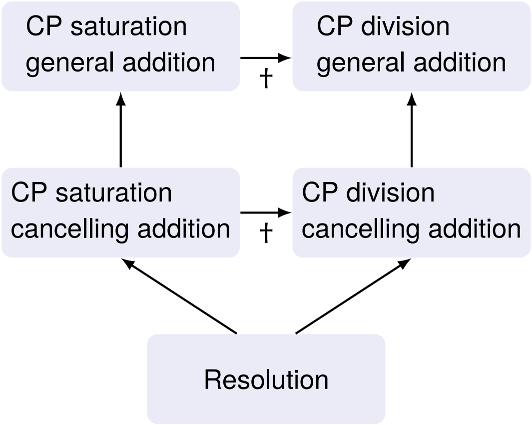



All subsystems simulate resolution

- Trivial over CNF inputs
- Also holds over linear pseudo-Boolean inputs

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)

# Proof Systems



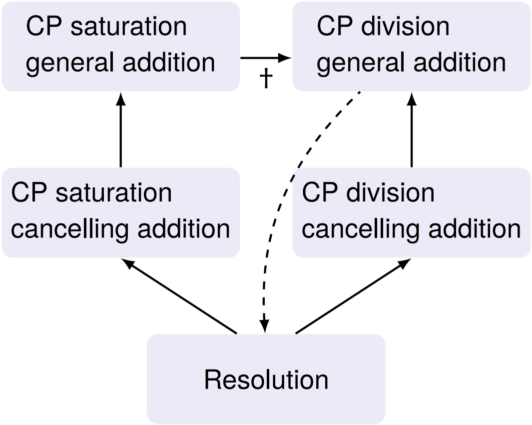

Repeated divisions simulate saturation

- Polynomial simulation only if polynomial coefficients

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)

†: known only for polynomial-size coefficients

# Proof Systems




CP stronger than resolution

- Pigeonhole principle
- Subset cardinality

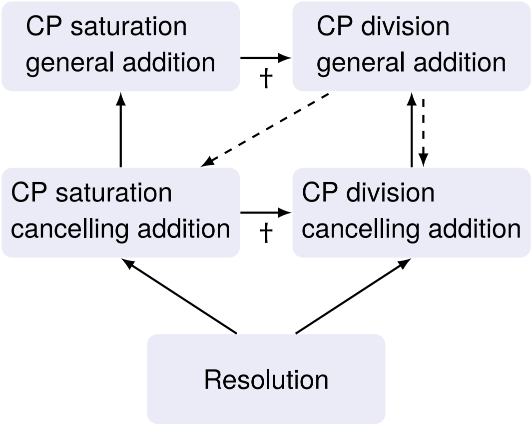
have proofs of size

- polynomial in PC
- exponential in resolution

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

# Proof Systems




Cancellation ≡ Resolution

- Over CNF inputs

[Hooker '88]

- Pigeonhole principle
- Subset cardinality

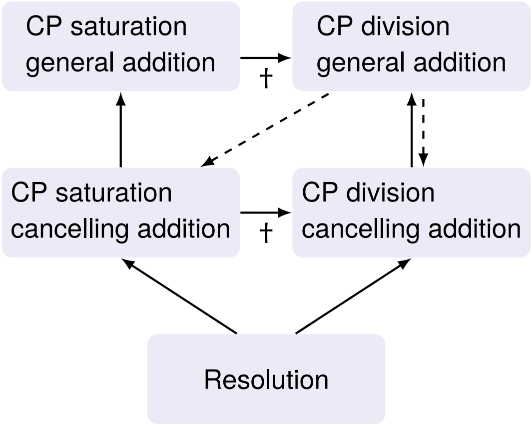
have proofs of size

- polynomial in PC
- exponential in resolution

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

# Proof Systems




$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

Cancellation ≡ Resolution

- Over CNF inputs

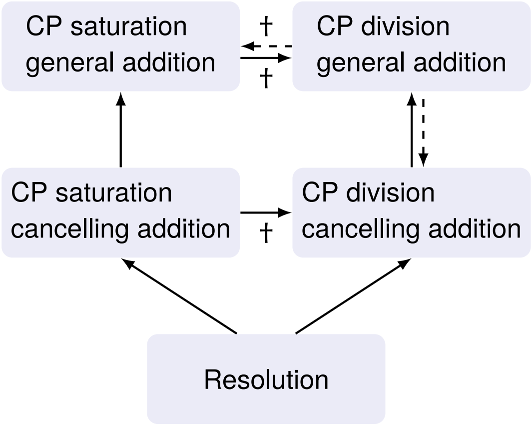
[Hooker '88]

- Pigeonhole principle
- Subset cardinality

have proofs of size

- polynomial in PC
- exponential in CP with cancelling addition and any rounding

# Proof Systems




Saturation ≡ Resolution

- Over CNF inputs

- Pigeonhole principle
- Subset cardinality

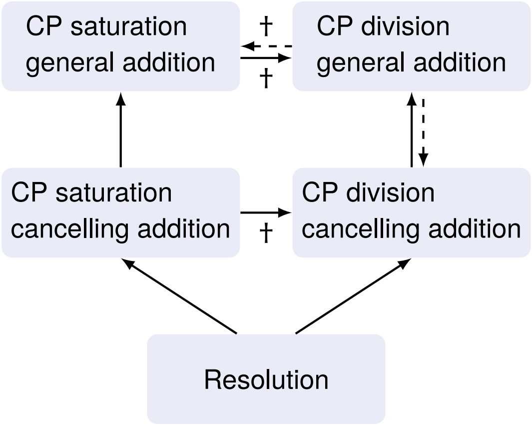
have proofs of size

- polynomial in PC
- exponential in resolution

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

# Proof Systems




Saturation ≡ Resolution

- Over CNF inputs

- Pigeonhole principle
- Subset cardinality

have proofs of size

- polynomial in PC
- exponential in CP with general addition and saturation

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

# Easy Formulas

Pseudo-Boolean solvers $\equiv$ CP? No

## Question

PB solvers $\equiv$ CP with cancelling addition and saturation?

# Easy Formulas

Pseudo-Boolean solvers $\equiv$ CP? No

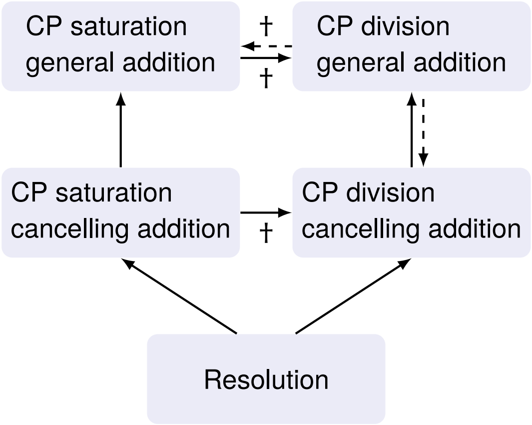
**Question**

PB solvers $\equiv$ CP with cancelling addition and saturation?

Craft combinatorial formulas easy for CP with cancelling addition and saturation

- All formulas without rational solutions
- Easy versions of NP-hard problems

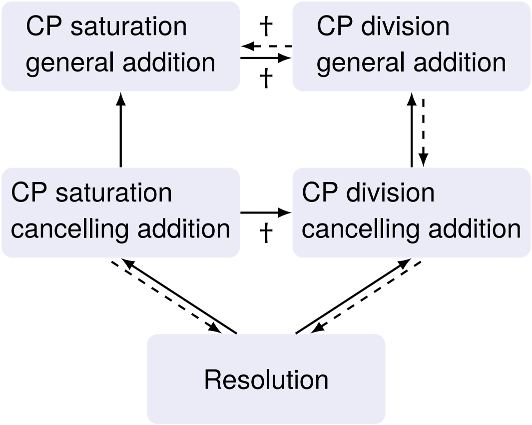
# Proof Systems




$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

# Proof Systems




Pseudo-Boolean versions of

- Pigeonhole principle
- Subset cardinality
- ...

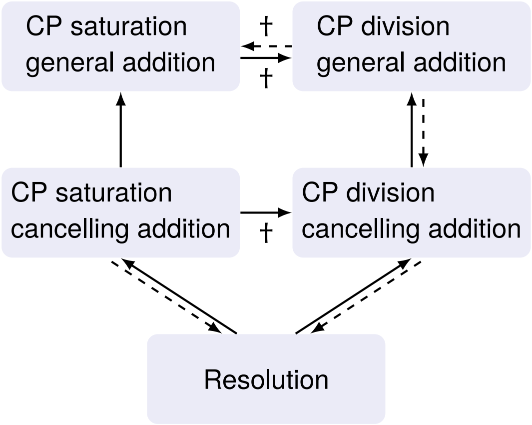
have proof of size

- polynomial in all CP subsystems
- exponential in resolution

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

# Proof Systems




$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)

†: known only for polynomial-size coefficients

Pseudo-Boolean versions of

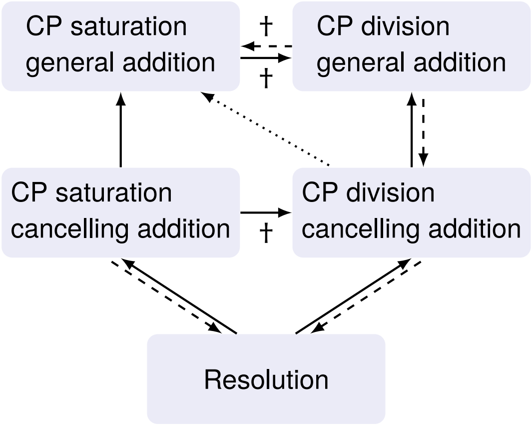
- Pigeonhole principle
- Subset cardinality
- ...

have proof of size

- polynomial in all CP subsystems
- exponential in resolution

CNF version exponential $\Rightarrow$
Cannot recover encoding $\Rightarrow$
Subsystems are incomplete

# Proof Systems




Separation candidates
Some formulas have proof of size

- polynomial in CP
  with cancelling addition
  and division
- unknown in CP
  with general addition
  and saturation

$A \longrightarrow B$: $B$ simulates $A$ (with only polynomial loss)
$A \dashrightarrow B$: $B$ cannot simulate $A$ (separation)
$A \cdots\to B$: candidate for a separation
†: known only for polynomial-size coefficients

# Take Home

## Bad News

- On CNF inputs subsystems of CP $\equiv$ resolution
- Subsystems of CP implicationally incomplete

# Take Home

## Bad News

- On CNF inputs subsystems of CP $\equiv$ resolution
- Subsystems of CP implicationally incomplete

## Good News

- Many formulas where PB solvers can shine
- Do PB solvers shine in practice?

# Take Home

## Bad News

- On CNF inputs subsystems of CP $\equiv$ resolution
- Subsystems of CP implicationally incomplete

## Good News

- Many formulas where PB solvers can shine
- Do PB solvers shine in practice? (Stay tuned...)

# Take Home

## Bad News

- On CNF inputs subsystems of CP $\equiv$ resolution
- Subsystems of CP implicationally incomplete

## Good News

- Many formulas where PB solvers can shine
- Do PB solvers shine in practice? (Stay tuned...)

Thanks!