Supercritical Space-Width Trade-offs for Resolution

Jakob Nordström

KTH Royal Institute of Technology

SAT and Interactions
Schloss Dagstuhl
September 19–23, 2016

Joint work with Christoph Berkholz

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\bot$ derived

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

1. $x \lor y$
2. $x \lor \overline{y} \lor z$
3. $\overline{x} \lor z$
4. $\overline{y} \lor \overline{z}$
5. $\overline{x} \lor \overline{z}$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$\boldsymbol{x \lor \overline{y} \lor z}$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\boldsymbol{\overline{y} \lor \overline{z}}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$\boldsymbol{x \lor \overline{y} \lor z}$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\boldsymbol{\overline{y} \lor \overline{z}}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$\boldsymbol{x \lor \overline{y}}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\bot$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$\boldsymbol{x \lor \overline{y}}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$\boldsymbol{x \vee y}$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$\boldsymbol{x \vee \overline{y}}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\bot$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$\boldsymbol{x \lor y}$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$\boldsymbol{x \lor \overline{y}}$	Res$(2,4)$
7.	$\boldsymbol{x}$	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\bot$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	$\boldsymbol{x}$	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\bot$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\boldsymbol{\overline{x} \lor z}$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\boldsymbol{\overline{x} \lor \overline{z}}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\boldsymbol{\overline{x} \vee z}$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\boldsymbol{\overline{x} \vee \overline{z}}$	Axiom
6.	$x \vee \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\boldsymbol{\overline{x}}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\boldsymbol{\overline{x}}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	$\boldsymbol{x}$	Res$(1,6)$
8.	$\boldsymbol{\overline{x}}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	$\boldsymbol{x}$	Res$(1, 6)$
8.	$\boldsymbol{\overline{x}}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$x \vee \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

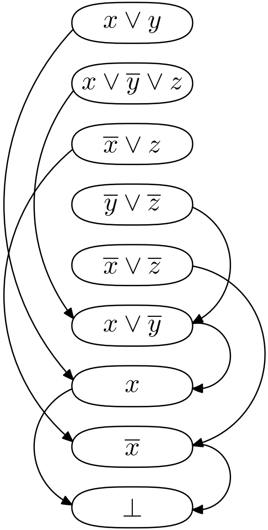
- Start with axiom clauses in formula
- Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)



The Resolution Proof System

Goal: refute **unsatisfiable** CNF

- Start with axiom clauses in formula
- Derive new clauses by resolution rule

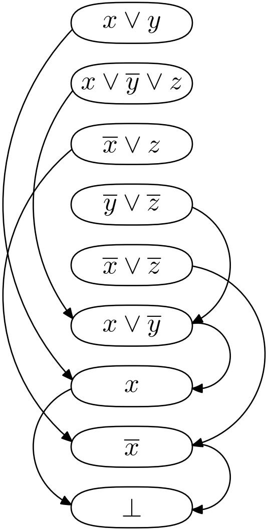
$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

- Done when empty clause $\perp$ derived

Can represent refutation/proof as

- annotated list or
- directed acyclic graph (DAG)

Tree-like resolution if DAG is tree



Resolution Size/Length and Width

Length of proof = # clauses (9 in our example)
Length of refuting F = min length over all proofs for F

Resolution Size/Length and Width

Length of proof = # clauses (9 in our example)
Length of refuting F = min length over all proofs for F

Size should strictly speaking measure # symbols
But for resolution don't care too much about linear factors here
Set size = length

Resolution Size/Length and Width

Length of proof = # clauses (9 in our example)
Length of refuting F = min length over all proofs for F

Size should strictly speaking measure # symbols
But for resolution don't care too much about linear factors here
Set size = length

Width of proof = # literals in largest clause (3 in our example)
Width of refuting F = min width over all proofs for F

Width at most linear, so here obviously care about linear factors

Resolution Space

Space = amount of memory needed
when performing refutation

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

Resolution Space

Space = amount of memory needed when performing refutation

Can be measured in different ways:

- clause space (our focus)
- total space

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

Resolution Space

Space = amount of memory needed when performing refutation

Can be measured in different ways:

- clause space (our focus)
- total space

Clause space at step t: # clauses at steps $\leq t$ used at steps $\geq t$
Total space at step t: Count also literals

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$x \vee \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

Resolution Space

Space = amount of memory needed when performing refutation

Can be measured in different ways:

- clause space (our focus)
- total space

Clause space at step t: # clauses at steps $\leq t$ used at steps $\geq t$
Total space at step t: Count also literals

Example: Clause space at step 7

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

Resolution Space

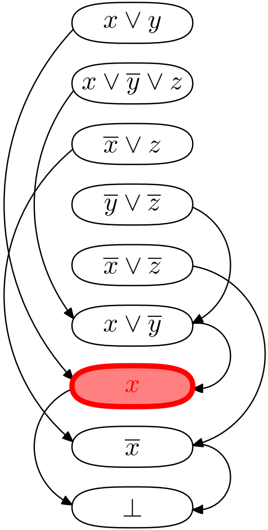
Space = amount of memory needed when performing refutation

Can be measured in different ways:

- clause space (our focus)
- total space

Clause space at step t: # clauses at steps $\leq t$ used at steps $\geq t$
Total space at step t: Count also literals

Example: Clause space at step 7



Resolution Space

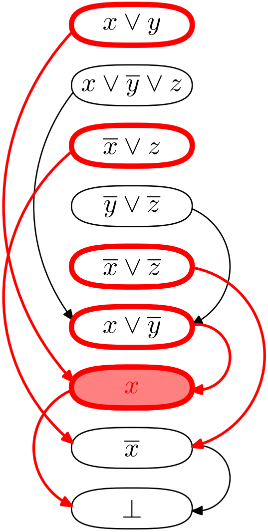
Space = amount of memory needed when performing refutation

Can be measured in different ways:

- clause space (our focus)
- total space

Clause space at step t: # clauses at steps $\leq t$ used at steps $\geq t$
Total space at step t: Count also literals

Example: Clause space at step 7 is 5



Resolution Space

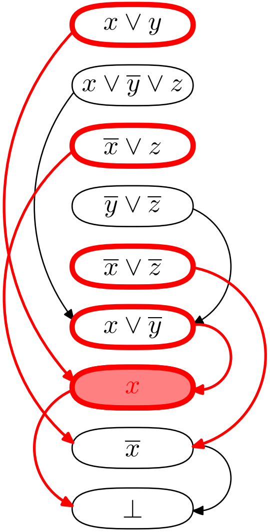
Space = amount of memory needed when performing refutation

Can be measured in different ways:

- clause space (our focus)
- total space

Clause space at step t: # clauses at steps $\leq t$ used at steps $\geq t$
Total space at step t: Count also literals

Example: Clause space at step 7 is 5
Total space at step 7 is 9



Resolution Space

Space = amount of memory needed
when performing refutation

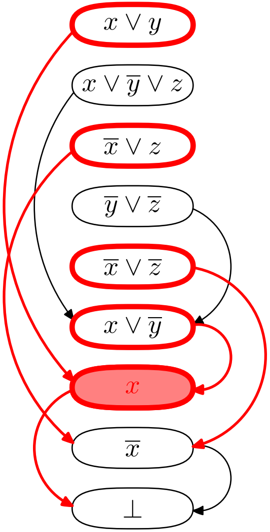
Can be measured in different ways:

- clause space (our focus)
- total space

Clause space at step t: # clauses at steps $\leq t$ used at steps $\geq t$
Total space at step t: Count also literals

Example: Clause space at step 7 is 5
Total space at step 7 is 9

Space of proof = max over all steps
Space of refuting F = min over all proofs



Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Size / length	# derivation steps	$\mathcal{O}(2^n)$

Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Size / length	# derivation steps	$\mathcal{O}(2^n)$
Width	max # literals in a clause	$\mathcal{O}(n)$

Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Size / length	# derivation steps	$\mathcal{O}(2^n)$
Width	max # literals in a clause	$\mathcal{O}(n)$
Clause space	max # clauses in memory	$\mathcal{O}(n)$

Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Size / length	# derivation steps	$\mathcal{O}(2^n)$
Width	max # literals in a clause	$\mathcal{O}(n)$
Clause space	max # clauses in memory	$\mathcal{O}(n)$
Total space	total size of memory	$\mathcal{O}(n^2)$

Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Size / length	# derivation steps	$\mathcal{O}(2^n)$
Width	max # literals in a clause	$\mathcal{O}(n)$
Clause space	max # clauses in memory	$\mathcal{O}(n)$
Total space	total size of memory	$\mathcal{O}(n^2)$

This talk: focus on width and clause space

Upper Bounds on Resolution Complexity Measures

Worst-case upper bounds for resolution refutations of formula (from now on assume $n = \#\text{variables}$):

Size / length	# derivation steps	$\mathcal{O}(2^n)$
Width	max # literals in a clause	$\mathcal{O}(n)$
Clause space	max # clauses in memory	$\mathcal{O}(n)$
Total space	total size of memory	$\mathcal{O}(n^2)$

This talk: focus on width and clause space
But results translate to total space by:

$$\text{clause space} \leq \text{total space} \leq \text{clause space} \cdot \text{width}$$

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

width $\leq$ Ω(clause space) [Atserias & Dalmau '03]

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

$$\begin{array}{lll} \text{width} & \leq & \Omega\big(\text{clause space}\big) \\ \text{width}^2 & \leq & \Omega\big(\text{total space}\big) \end{array}$$

[Atserias & Dalmau '03]
[Bonacina '16]

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

width	$\leq$	$\Omega\big(\text{clause space}\big)$	[Atserias & Dalmau '03]
width^2	$\leq$	$\Omega\big(\text{total space}\big)$	[Bonacina '16]
width^2	$\leq$	$\Omega\big(n \log(\text{size})\big)$	[Ben-Sasson & Widgerson '99]

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

width	$\leq$	$\Omega\big(\text{clause space}\big)$	[Atserias & Dalmau '03]
width^2	$\leq$	$\Omega\big(\text{total space}\big)$	[Bonacina '16]
width^2	$\leq$	$\Omega\big(n \log(\text{size})\big)$	[Ben-Sasson & Widgerson '99]

In particular, width $= \Omega(n) \Longrightarrow$ size $= \Omega\big(2^n\big)$

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

width	$\leq$	$\Omega\big(\text{clause space}\big)$	[Atserias & Dalmau '03]
width^2	$\leq$	$\Omega\big(\text{total space}\big)$	[Bonacina '16]
width^2	$\leq$	$\Omega\big(n \log(\text{size})\big)$	[Ben-Sasson & Widgerson '99]

In particular, width $= \Omega(n) \Longrightarrow$ size $= \Omega\big(2^n\big)$

So clearly width key measure—but not the answer to every question

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

width	$\leq$	Ω(clause space)	[Atserias & Dalmau '03]
width^2	$\leq$	Ω(total space)	[Bonacina '16]
width^2	$\leq$	$\Omega(n \log(\text{size}))$	[Ben-Sasson & Widgerson '99]

In particular, width $= \Omega(n) \Longrightarrow$ size $= \Omega(2^n)$

So clearly width key measure—but not the answer to every question

- Can have width $\Theta(\sqrt{n})$ and still size $\mathrm{poly}(n)$ [Bonet & Galesi '99]

Lower Bounds via Resolution Width

For n-variable k-CNFs (k constant) it holds that:

width	$\leq$	$\Omega(\text{clause space})$	[Atserias & Dalmau '03]
width^2	$\leq$	$\Omega(\text{total space})$	[Bonacina '16]
width^2	$\leq$	$\Omega(n \log(\text{size}))$	[Ben-Sasson & Widgerson '99]

In particular, width $= \Omega(n) \Longrightarrow$ size $= \Omega(2^n)$

So clearly width key measure—but not the answer to every question

- Can have width $\Theta(\sqrt{n})$ and still size $\mathrm{poly}(n)$ [Bonet & Galesi '99]
- Can have width $\mathcal{O}(1)$ and still clause space $\Omega(n/\log n)$ [Ben-Sasson & Nordström '08]

Upper Bounds via Resolution Width

$$\textsf{size} \quad \leq \quad n^{\mathcal{O}(\textsf{width})}$$

Upper Bounds via Resolution Width

$$\begin{aligned} \text{size} &\leq n^{\mathcal{O}(\text{width})} \\ \text{time to find a refutation} &\leq n^{\mathcal{O}(\text{width})} \end{aligned}$$

for $w \leftarrow 3 \ldots n$ **do**
  Resolve all clauses & keep resolvents with at most w literals
  If $\perp$ has been derived, then output `UNSAT`
end for
Output `SAT`

Upper Bounds via Resolution Width

$$\begin{aligned} \text{size} &\leq n^{\mathcal{O}(\text{width})} \\ \text{time to find a refutation} &\leq n^{\mathcal{O}(\text{width})} \end{aligned}$$

for $w \leftarrow 3 \ldots n$ **do**
  Resolve all clauses & keep resolvents with at most w literals
  If $\perp$ has been derived, then output `UNSAT`
end for
Output `SAT`

Algorithm (and resolution proof) requires $n^{\mathcal{O}(\text{width})}$ space!

Upper Bounds via Resolution Width

$$\begin{aligned} \text{size} &\leq n^{\mathcal{O}(\text{width})} \\ \text{time to find a refutation} &\leq n^{\mathcal{O}(\text{width})} \end{aligned}$$

for $w \leftarrow 3 \ldots n$ **do**
  Resolve all clauses & keep resolvents with at most w literals
  If $\perp$ has been derived, then output `UNSAT`
end for
Output `SAT`

Algorithm (and resolution proof) requires $n^{\mathcal{O}(\text{width})}$ space!

[Ben-Sasson '02] exhibited formulas

- refutable in width $\mathcal{O}(1)$ and clause space $\mathcal{O}(1)$
- width $\mathcal{O}(1) \implies$ clause space $\Omega(n/\log n)$

Upper Bounds via Resolution Width

$$\begin{aligned} \text{size} &\leq n^{\mathcal{O}(\text{width})} \\ \text{time to find a refutation} &\leq n^{\mathcal{O}(\text{width})} \end{aligned}$$

for $w \leftarrow 3 \ldots n$ **do**
  Resolve all clauses & keep resolvents with at most w literals
  If $\perp$ has been derived, then output `UNSAT`
end for
Output `SAT`

Algorithm (and resolution proof) requires $n^{\mathcal{O}(\text{width})}$ space!

[Ben-Sasson '02] exhibited formulas

- refutable in width $\mathcal{O}(1)$ and clause space $\mathcal{O}(1)$
- width $\mathcal{O}(1) \implies$ clause space $\Omega(n/\log n)$

Which bound is closer to the truth?

Upper Bounds via Resolution Width

$$\begin{aligned} \text{size} &\leq n^{\mathcal{O}(\text{width})} \\ \text{time to find a refutation} &\leq n^{\mathcal{O}(\text{width})} \end{aligned}$$

for $w \leftarrow 3 \dots n$ **do**
  Resolve all clauses & keep resolvents with at most w literals
  If $\perp$ has been derived, then output `UNSAT`
end for
Output `SAT`

Algorithm (and resolution proof) requires $n^{\mathcal{O}(\text{width})}$ space!

[Ben-Sasson '02] exhibited formulas

- refutable in width $\mathcal{O}(1)$ and clause space $\mathcal{O}(1)$
- width $\mathcal{O}(1) \Longrightarrow$ clause space $\Omega(n/\log n)$

Which bound is closer to the truth?
Recall: can always do clause space $\mathcal{O}(n)$

A Supercritical Space-Width Tradeoff

Theorem

For any $\varepsilon > 0$ *and* $6 \leq w \leq n^{\frac{1}{2}-\varepsilon}$ *exist* n*-variable* w-CNF*s* F_n *s.t.*

1. *Resolution can refute* F_n *in width* w
2. *Any width-*w *refutation of* F_n *requires clause space* $n^{\Omega(w)}$

A Supercritical Space-Width Tradeoff

Theorem

For any $\varepsilon > 0$ *and* $6 \leq w \leq n^{\frac{1}{2}-\varepsilon}$ *exist* n*-variable* w-CNF*s* F_n *s.t.*

1. *Resolution can refute* F_n *in width* w
2. *Any width-*w *refutation of* F_n *requires clause space* $n^{\Omega(w)}$

Space lower bound $n^{\Omega(w)}$ holds for all proofs up to width $o(w \log n)$

A Supercritical Space-Width Tradeoff

Theorem

For any $\varepsilon > 0$ and $6 \leq w \leq n^{\frac{1}{2}-\varepsilon}$ exist n-variable w-CNFs F_n s.t.

1. *Resolution can refute F_n in width w*
2. *Any width-w refutation of F_n requires clause space $n^{\Omega(w)}$*

Space lower bound $n^{\Omega(w)}$ holds for all proofs up to width $o(w \log n)$

Proof outline

Use hardness condensation approach in [Razborov '16]:

1. Start with formula that requires nearly linear clause space
2. Reduce the number of variables from n to $n^{1/w}$
3. But maintain space lower bound for small-width proofs

A Supercritical Space-Width Tradeoff

Theorem

For any $\varepsilon > 0$ and $6 \leq w \leq n^{\frac{1}{2}-\varepsilon}$ exist n-variable w-CNFs F_n s.t.

1. *Resolution can refute F_n in width w*
2. *Any width-w refutation of F_n requires clause space $n^{\Omega(w)}$*

Space lower bound $n^{\Omega(w)}$ holds for all proofs up to width $o(w \log n)$

Proof outline

Use hardness condensation approach in [Razborov '16]:

1. Start with formula that requires nearly linear clause space
2. Reduce the number of variables from n to $n^{1/w}$
3. But maintain space lower bound for small-width proofs

Key components:

- Expander graphs
- XORification (substitution with exclusive or)

What Do You Mean "Supercritical"?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ

What Do You Mean “Supercritical”?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}

What Do You Mean "Supercritical"?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$

What Do You Mean "Supercritical"?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$
  - Any solution S with $\varphi(S)$ even medium-small must have $\psi(S)$ approach critical value ψ_{crit}

What Do You Mean "Supercritical"?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$
  - Any solution S with $\varphi(S)$ even medium-small must have $\psi(S)$ approach critical value ψ_{crit}
  - Conversely, $\psi(S)$ medium-small $\Longrightarrow \varphi(S) \approx \varphi_{\mathrm{crit}}$

What Do You Mean "Supercritical"?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$
  - Any solution S with $\varphi(S)$ even medium-small must have $\psi(S)$ approach critical value ψ_{crit}
  - Conversely, $\psi(S)$ medium-small $\Longrightarrow \varphi(S) \approx \varphi_{\mathrm{crit}}$

Supercritical setting for trade-offs:

- Any S with $\varphi(S)$ medium-small must have $\psi(S) \gg \psi_{\mathrm{crit}}$

What Do You Mean “Supercritical”?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$
  - Any solution S with $\varphi(S)$ even medium-small must have $\psi(S)$ approach critical value ψ_{crit}
  - Conversely, $\psi(S)$ medium-small $\Longrightarrow \varphi(S) \approx \varphi_{\mathrm{crit}}$

Supercritical setting for trade-offs:

- Any S with $\varphi(S)$ medium-small must have $\psi(S) \gg \psi_{\mathrm{crit}}$
- Optimizing φ pushes ψ up into supercritical regime above worst case!

What Do You Mean “Supercritical”?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$
  - Any solution S with $\varphi(S)$ even medium-small must have $\psi(S)$ approach critical value ψ_{crit}
  - Conversely, $\psi(S)$ medium-small $\Longrightarrow \varphi(S) \approx \varphi_{\mathrm{crit}}$

Supercritical setting for trade-offs:

- Any S with $\varphi(S)$ medium-small must have $\psi(S) \gg \psi_{\mathrm{crit}}$
- Optimizing φ pushes ψ up into supercritical regime above worst case!
- **Very** strong trade-offs—Razborov refers to them as “ultimate”

What Do You Mean "Supercritical"?!

Typical setting for trade-off results:

- Have two complexity measures φ and ψ
- Worst-case (usually trivial) upper bounds φ_{crit} and ψ_{crit}
- There are instances I_n such that:
  - $\exists$ solutions S_1, S_2 with $\varphi(S_1) = \text{small}'$ and $\psi(S_2) = \text{small}''$
  - Any solution S with $\varphi(S)$ even medium-small must have $\psi(S)$ approach critical value ψ_{crit}
  - Conversely, $\psi(S)$ medium-small $\Longrightarrow \varphi(S) \approx \varphi_{\mathrm{crit}}$

Supercritical setting for trade-offs:

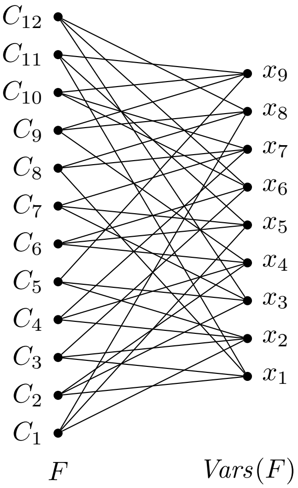
- Any S with $\varphi(S)$ medium-small must have $\psi(S) \gg \psi_{\mathrm{crit}}$
- Optimizing φ pushes ψ up into supercritical regime above worst case!
- **Very** strong trade-offs—Razborov refers to them as "ultimate"
- We feel "supercritical" is more descriptive

Expanders

Very well-connected so-called expander graphs play leading role in many proof complexity lower bounds

Expanders

Very well-connected so-called expander graphs play leading role in many proof complexity lower bounds

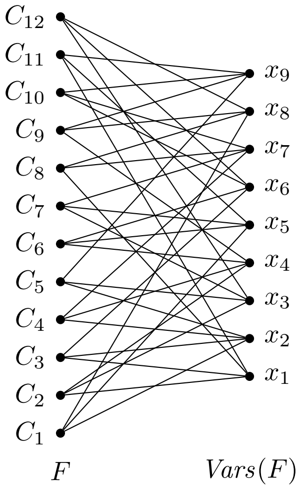


Clause-variable incidence graph (CVIG)

- Clauses on the left
- Variables on the right
- Edge if variable $\in$ clause (ignore signs)

Expanders

Very well-connected so-called expander graphs play leading role in many proof complexity lower bounds



Clause-variable incidence graph (CVIG)

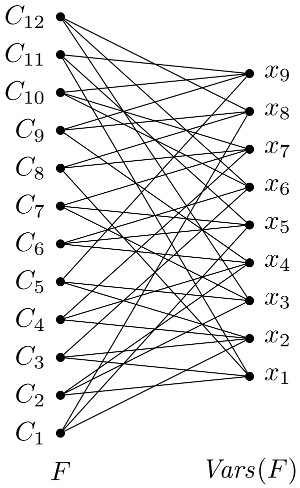
- Clauses on the left
- Variables on the right
- Edge if variable $\in$ clause (ignore signs)

If CVIG well-connected, then lower bounds for

- width, size, and space in resolution
  [Ben-Sasson & Wigderson '99, Ben-Sasson & Galesi '03]
- degree and size in polynomial calculus
  [Impagliazzo et al. '99, Alekhnovich & Razborov '01]

Expanders

Very well-connected so-called expander graphs play leading role in many proof complexity lower bounds



Clause-variable incidence graph (CVIG)

- Clauses on the left
- Variables on the right
- Edge if variable $\in$ clause (ignore signs)

If CVIG well-connected, then lower bounds for

- width, size, and space in resolution [Ben-Sasson & Wigderson '99, Ben-Sasson & Galesi '03]
- degree and size in polynomial calculus [Impagliazzo et al. '99, Alekhnovich & Razborov '01]

Can also define more general graphs that capture "underlying combinatorial structure" and extend results [Mikša & Nordström '15]

XORification

Modify F to $F[\oplus_2]$ by substituting $x_1 \oplus x_2$ for every variable x

XORification

Modify F to $F[\oplus_2]$ by substituting $x_1 \oplus x_2$ for every variable x

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

$$\Downarrow$$

$$\neg\,(x_1 \oplus x_2) \vee (y_1 \oplus y_2)$$

$$\Downarrow$$

$$\begin{aligned}
&(x_1 \vee \overline{x}_2 \vee y_1 \vee y_2)\\
\wedge\ &(x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee \overline{y}_2)\\
\wedge\ &(\overline{x}_1 \vee x_2 \vee y_1 \vee y_2)\\
\wedge\ &(\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee \overline{y}_2)
\end{aligned}$$

XORification

Modify F to $F[\oplus_2]$ by substituting $x_1 \oplus x_2$ for every variable x

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

$$\Downarrow$$

$$\neg\,(x_1 \oplus x_2) \vee (y_1 \oplus y_2)$$

$$\Downarrow$$

$$\begin{aligned} &(x_1 \vee \overline{x}_2 \vee y_1 \vee y_2) \\ \wedge\ &(x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee \overline{y}_2) \\ \wedge\ &(\overline{x}_1 \vee x_2 \vee y_1 \vee y_2) \\ \wedge\ &(\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee \overline{y}_2) \end{aligned}$$

Used to prove, e.g.:

- width $\geq w$ for $F \Longrightarrow$ size $\geq \exp(\Omega(w))$ for $F[\oplus_2]$ [Ben-Sasson '02] (credited to [Alekhnovich & Razborov])

XORification

Modify F to $F[\oplus_2]$ by substituting $x_1 \oplus x_2$ for every variable x

$$\overline{x} \vee y$$
$$\Downarrow$$
$$\neg\,(x_1 \oplus x_2) \vee (y_1 \oplus y_2)$$
$$\Downarrow$$
$$\begin{aligned} & (x_1 \vee \overline{x}_2 \vee y_1 \vee y_2) \\ \wedge\ & (x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee \overline{y}_2) \\ \wedge\ & (\overline{x}_1 \vee x_2 \vee y_1 \vee y_2) \\ \wedge\ & (\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee \overline{y}_2) \end{aligned}$$

Used to prove, e.g.:

- width $\geq w$ for $F \Longrightarrow$ size $\geq \exp(\Omega(w))$ for $F[\oplus_2]$ [Ben-Sasson '02] (credited to [Alekhnovich & Razborov])
- # vars in memory $\geq s$ for $F \Longrightarrow$ clause space $\geq \Omega(s)$ for $F[\oplus_2]$ [Ben-Sasson & Nordström '08]

Intuition for XORification Lower Bounds

How to construct resolution refutation π of $F[\oplus_2]$?

Intuition for XORification Lower Bounds

How to construct resolution refutation π of $F[\oplus_2]$?

Naive idea: Simulate resolution refutation π' of F
(using substitution on previous slide)

Intuition for XORification Lower Bounds

How to construct resolution refutation π of $F[\oplus_2]$?

Naive idea: Simulate resolution refutation π' of F
(using substitution on previous slide)

Seems like a bad idea—XORification causes bad blow-up

- linear in # variables in memory
- exponential in width

Intuition for XORification Lower Bounds

How to construct resolution refutation π of $F[\oplus_2]$?

Naive idea: Simulate resolution refutation π' of F
(using substitution on previous slide)

Seems like a bad idea—XORification causes bad blow-up

- linear in # variables in memory
- exponential in width

Nevertheless, can prove (sort of) this is the best resolution can do

Intuition for XORification Lower Bounds

How to construct resolution refutation π of $F[\oplus_2]$?

Naive idea: Simulate resolution refutation π' of F
(using substitution on previous slide)

Seems like a bad idea—XORification causes bad blow-up

- linear in # variables in memory
- exponential in width

Nevertheless, can prove (sort of) this is the best resolution can do

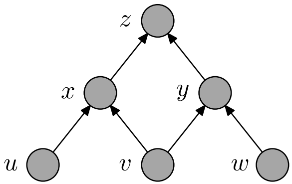
Intuition behind proof

- Given resolution refutation π of $F[\oplus_2]$
- Extract the refutation π' of F that π is simulating
- Prove that extraction preserves complexity measures of interest

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \land (v_1 \oplus v_2) \to (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \land (w_1 \oplus w_2) \to (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \land (y_1 \oplus y_2) \to (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$

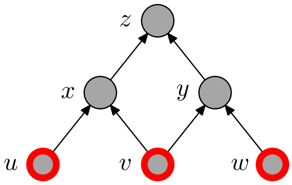


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \wedge (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \wedge (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \wedge (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$

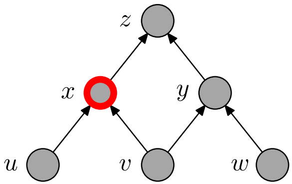


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \wedge (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \wedge (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \wedge (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$

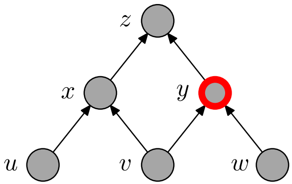


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \wedge (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \wedge (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \wedge (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$

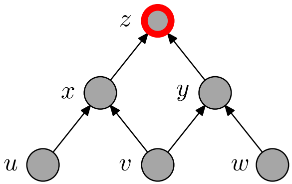


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \wedge (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \wedge (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \wedge (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$

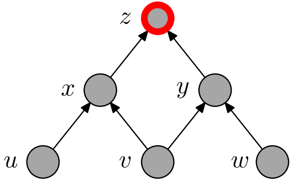


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \wedge (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \wedge (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \wedge (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$

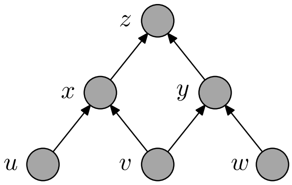


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \wedge (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \wedge (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \wedge (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$



- sources are true
- truth propagates upwards
- but sink is false

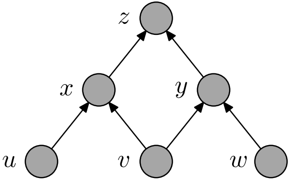
Written in CNF as explained before, e.g.

$$
\begin{aligned}
u_1 \oplus u_2 &= (u_1 \vee u_2) \wedge (\overline{u}_1 \vee \overline{u}_2) \\
\neg(z_1 \oplus z_2) &= (z_1 \vee \overline{z}_2) \wedge (\overline{z}_1 \vee z_2)
\end{aligned}
$$

Pebbling Formulas

Encode pebble games on DAGs
[Ben-Sasson & Wigderson '99]

1. $u_1 \oplus u_2$
2. $v_1 \oplus v_2$
3. $w_1 \oplus w_2$
4. $(u_1 \oplus u_2) \land (v_1 \oplus v_2) \rightarrow (x_1 \oplus x_2)$
5. $(v_1 \oplus v_2) \land (w_1 \oplus w_2) \rightarrow (y_1 \oplus y_2)$
6. $(x_1 \oplus x_2) \land (y_1 \oplus y_2) \rightarrow (z_1 \oplus z_2)$
7. $\neg(z_1 \oplus z_2)$



- sources are true
- truth propagates upwards
- but sink is false

Written in CNF as explained before, e.g.

$$
\begin{aligned}
u_1 \oplus u_2 &= (u_1 \lor u_2) \land (\overline{u}_1 \lor \overline{u}_2) \\
\neg(z_1 \oplus z_2) &= (z_1 \lor \overline{z}_2) \land (\overline{z}_1 \lor z_2)
\end{aligned}
$$

Easy to refute pebbling formulas in size $\mathcal{O}(n)$ and width $\mathcal{O}(1)$
Pebbling space lower bounds $\Rightarrow$ clause space lower bounds
[Ben-Sasson & Nordström '08, '11]

XOR Substitution with Recycling (1/2)

Suppose

- F CNF formula over variables U
- $\mathcal{G} = (U \mathbin{\dot\cup} V, E)$ bipartite graph

Substituted formula $F[\mathcal{G}]$ over variables V:

- replace every $u \in U$ by $\bigoplus_{v \in N(u)} v$

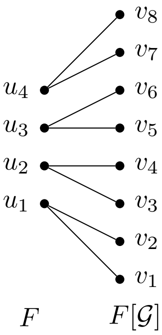
XOR Substitution with Recycling (1/2)

Suppose

- F CNF formula over variables U
- $\mathcal{G} = (U \,\dot\cup\, V, E)$ bipartite graph

Substituted formula $F[\mathcal{G}]$ over variables V:

- replace every $u \in U$ by $\bigoplus_{v \in N(u)} v$



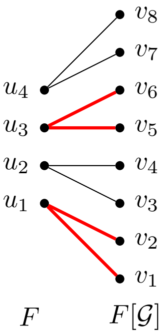
XOR Substitution with Recycling (1/2)

Suppose

- F CNF formula over variables U
- $\mathcal{G} = (U \,\dot\cup\, V, E)$ bipartite graph

Substituted formula $F[\mathcal{G}]$ over variables V:

- replace every $u \in U$ by $\bigoplus_{v \in N(u)} v$



$$\overline{u}_1 \vee u_3 \quad \longrightarrow \quad \neg\,(v_1 \oplus v_2) \vee (v_5 \oplus v_6)$$

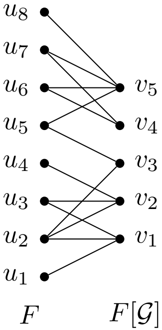
XOR Substitution with Recycling (1/2)

Suppose

- F CNF formula over variables U
- $\mathcal{G} = (U \dot{\cup} V, E)$ bipartite graph

Substituted formula $F[\mathcal{G}]$ over variables V:

- replace every $u \in U$ by $\bigoplus_{v \in N(u)} v$



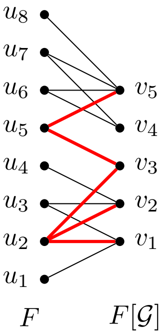
XOR Substitution with Recycling (1/2)

Suppose

- F CNF formula over variables U
- $\mathcal{G} = (U \mathbin{\dot{\cup}} V, E)$ bipartite graph

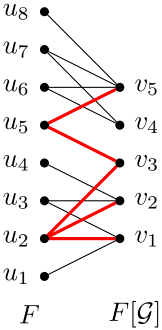
Substituted formula $F[\mathcal{G}]$ over variables V:

- replace every $u \in U$ by $\bigoplus_{v \in N(u)} v$



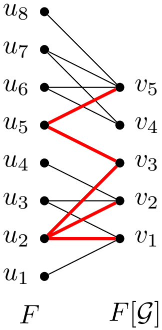
$$\overline{u}_2 \lor u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \lor (v_4 \oplus v_6)$$

XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \lor u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \lor (v_4 \oplus v_6)$$

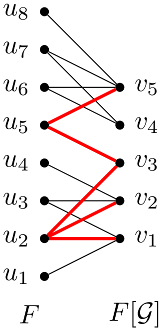
XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \vee u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \vee (v_4 \oplus v_6)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$

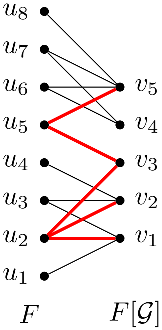
XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \vee u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \vee (v_4 \oplus v_6)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$

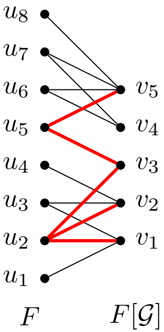
XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \lor u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \lor (v_4 \oplus v_6)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$

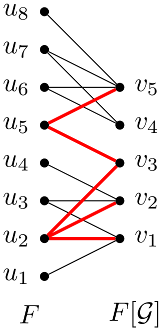
XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \vee u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \vee (v_4 \oplus v_6)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$ ✓

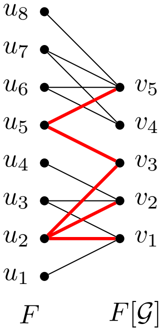
XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \lor u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \lor (v_4 \oplus v_6)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$ ✓
  - space of width-w refutation of $F[\mathcal{G}] \gtrapprox$ space of refutation of $F = \Omega(n/\log n) = |V|^{\Omega(w)}$

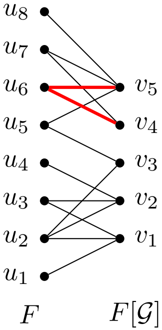
XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \lor u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \lor (v_4 \oplus v_6)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$ ✓
  - space of width-w refutation of $F[\mathcal{G}] \gtrapprox$ space of refutation of $F = \Omega(n/\log n) = |V|^{\Omega(w)}$?

XOR Substitution with Recycling (2/2)

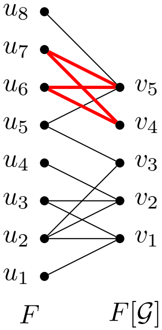


$$\overline{u}_2 \vee u_5 \quad \longrightarrow \quad \neg\,(v_2 \oplus v_3 \oplus v_4) \vee (v_4 \oplus v_6)$$

$$u_6 \quad \longrightarrow \quad (v_4 \oplus v_5)$$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$ ✓
  - space of width-w refutation of $F[\mathcal{G}] \gtrapprox$ space of refutation of $F = \Omega(n/\log n) = |V|^{\Omega(w)}$ **?**

XOR Substitution with Recycling (2/2)

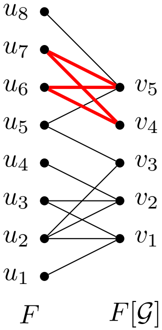


$$\begin{aligned}
\overline{u}_2 \vee u_5 &\longrightarrow \neg\,(v_2 \oplus v_3 \oplus v_4) \vee (v_4 \oplus v_6) \\
u_6 &\longrightarrow (v_4 \oplus v_5) \\
\overline{u}_7 &\longrightarrow \neg\,(v_4 \oplus v_5)
\end{aligned}$$

$\lightning$

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$ ✓
  - space of width-w refutation of $F[\mathcal{G}] \gtrsim$ space of refutation of $F = \Omega(n/\log n) = |V|^{\Omega(w)}$ **?**

XOR Substitution with Recycling (2/2)



$$\overline{u}_2 \vee u_5 \longrightarrow \neg(v_2 \oplus v_3 \oplus v_4) \vee (v_4 \oplus v_6)$$

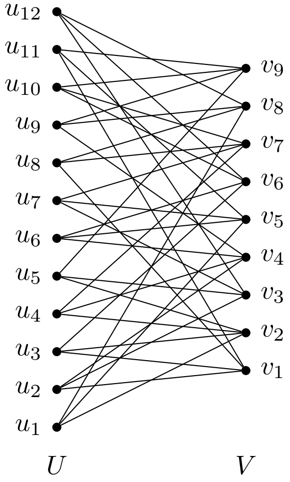
$$u_6 \longrightarrow (v_4 \oplus v_5)$$

$$\overline{u}_7 \longrightarrow \neg(v_4 \oplus v_5)$$

Solution: Use expander graphs!

- Apply to pebbling formulas F in [Ben-Sasson & Nordström '08]
  - refutable in width 6
  - require space $\Omega(n/\log n)$
- $\mathcal{G}$ expander with left-degree $\leq w/6$, $|U| = n$, and $|V| = n^{\mathcal{O}(1/w)}$
  - $F[\mathcal{G}]$ refutable in width $\leq w$ ✓
  - space of width-w refutation of $F[\mathcal{G}] \gtrapprox$ space of refutation of $F = \Omega(n/\log n) = |V|^{\Omega(w)}$ ✓

Bipartite Boundary Expander

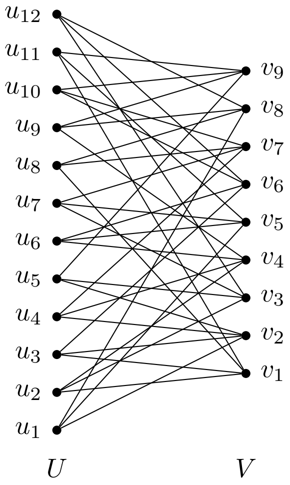


$\mathcal{G} = (U \dot\cup V, E)$ is (d, r, c)-boundary expander if

- left-degree $\leq d$
- for every $U' \subseteq U$, $|U'| \leq r$ it holds that $|\partial(U')| \geq c|U'|$

$\partial(U') := \{v \in N(U') : |N(v) \cap U'| = 1\}$

Bipartite Boundary Expander



$\mathcal{G} = (U \dot\cup V, E)$ is (d, r, c)-boundary expander if

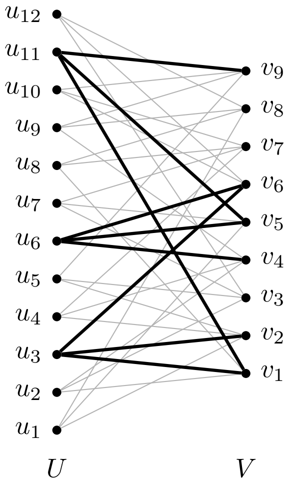
- left-degree $\leq d$
- for every $U' \subseteq U$, $|U'| \leq r$ it holds that $|\partial(U')| \geq c|U'|$

$\partial(U') := \{v \in N(U') : |N(v) \cap U'| = 1\}$

Example

- left-degree $d = 3$
- expanding set size $r = 3$
- expansion factor $c = 1$

Bipartite Boundary Expander



$\mathcal{G} = (U \mathbin{\dot\cup} V, E)$ is (d, r, c)-boundary expander if

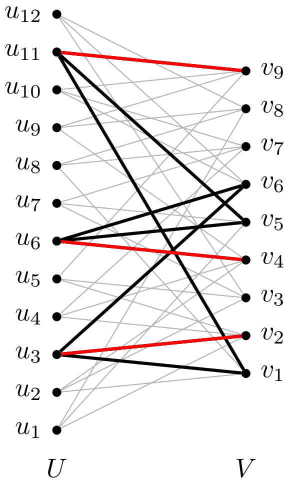
- left-degree $\leq d$
- for every $U' \subseteq U$, $|U'| \leq r$ it holds that $|\partial(U')| \geq c|U'|$

$$\partial(U') := \{v \in N(U') : |N(v) \cap U'| = 1\}$$

Example

- left-degree $d = 3$
- expanding set size $r = 3$
- expansion factor $c = 1$

Bipartite Boundary Expander



$\mathcal{G} = (U \dot\cup V, E)$ is (d, r, c)-boundary expander if

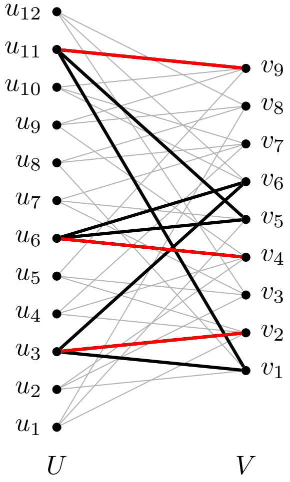
- left-degree $\leq d$
- for every $U' \subseteq U$, $|U'| \leq r$ it holds that $|\partial(U')| \geq c|U'|$

$\partial(U') := \{v \in N(U') : |N(v) \cap U'| = 1\}$

Example

- left-degree $d = 3$
- expanding set size $r = 3$
- expansion factor $c = 1$

Bipartite Boundary Expander



$\mathcal{G} = (U \dot\cup V, E)$ is (d, r, c)-boundary expander if

- left-degree $\leq d$
- for every $U' \subseteq U$, $|U'| \leq r$ it holds that $|\partial(U')| \geq c|U'|$

$\partial(U') := \{v \in N(U') : |N(v) \cap U'| = 1\}$

Example

- left-degree $d = 3$
- expanding set size $r = 3$
- expansion factor $c = 1$

Lemma ([Razborov '16])

For $\varepsilon > 0$ and n, d with $d \leq |V|^{\frac{1}{2}-\varepsilon}$, $|U| = n$, $|V| = n^{\mathcal{O}(1/d)}$ there are $(d, r, 2)$-boundary expanders $\mathcal{G}$ with $r = d \log n$

Sketch of Proof Sketch

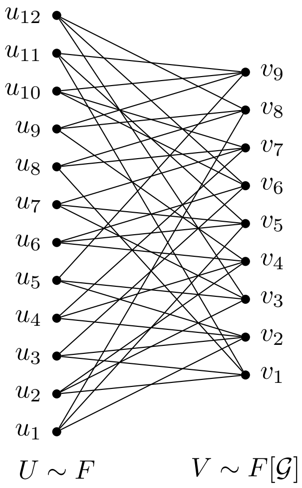
Look at clauses $\mathcal{D}$ in memory in width-w refutation of $F[\mathcal{G}]$

Recover clauses $\mathcal{C}$ in memory in “simulated refutation” of F

Sketch of Proof Sketch

Look at clauses $\mathcal{D}$ in memory in width-w refutation of $F[\mathcal{G}]$

Recover clauses $\mathcal{C}$ in memory in “simulated refutation” of F



Must have $N(\mathit{Vars}(\mathcal{C})) \subseteq \mathit{Vars}(\mathcal{D})$

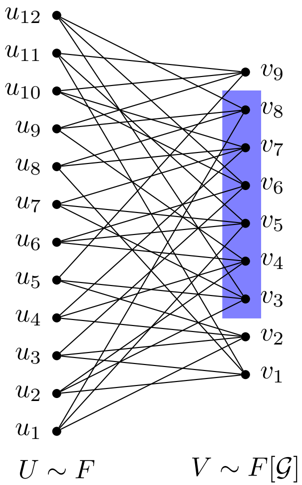
$\mathrm{Ker}(V') := \{u \in U : N(u) \subseteq V'\}$

$|V'| \leq r \implies |\mathrm{Ker}(V')| \leq |V'|$
(since left vertex sets expand a lot)

Sketch of Proof Sketch

Look at clauses $\mathcal{D}$ in memory in width-w refutation of $F[\mathcal{G}]$

Recover clauses $\mathcal{C}$ in memory in "simulated refutation" of F



Must have $N(\mathit{Vars}(\mathcal{C})) \subseteq \mathit{Vars}(\mathcal{D})$

$\mathrm{Ker}(V') := \{u \in U : N(u) \subseteq V'\}$

$|V'| \leq r \implies |\mathrm{Ker}(V')| \leq |V'|$
(since left vertex sets expand a lot)

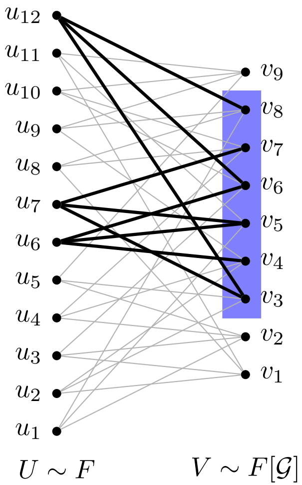
Example

$V' = \{v_3, \ldots, v_8\}$, $\mathrm{Ker}(V') = \{u_6, u_7, u_{12}\}$

Sketch of Proof Sketch

Look at clauses $\mathcal{D}$ in memory in width-w refutation of $F[\mathcal{G}]$

Recover clauses $\mathcal{C}$ in memory in "simulated refutation" of F



Must have $N(\mathit{Vars}(\mathcal{C})) \subseteq \mathit{Vars}(\mathcal{D})$

$\mathrm{Ker}(V') := \{u \in U \,:\, N(u) \subseteq V'\}$

$|V'| \leq r \quad \Longrightarrow \quad |\mathrm{Ker}(V')| \leq |V'|$
(since left vertex sets expand a lot)

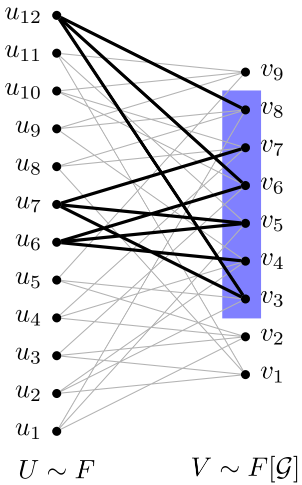
Example

$V' = \{v_3, \ldots, v_8\}$, $\mathrm{Ker}(V') = \{u_6, u_7, u_{12}\}$

Sketch of Proof Sketch

Look at clauses $\mathcal{D}$ in memory in width-w refutation of $F[\mathcal{G}]$

Recover clauses $\mathcal{C}$ in memory in "simulated refutation" of F



Must have $N(\mathit{Vars}(\mathcal{C})) \subseteq \mathit{Vars}(\mathcal{D})$

$\mathrm{Ker}(V') := \{u \in U \ : \ N(u) \subseteq V'\}$

$|V'| \leq r \implies |\mathrm{Ker}(V')| \leq |V'|$
(since left vertex sets expand a lot)

Example

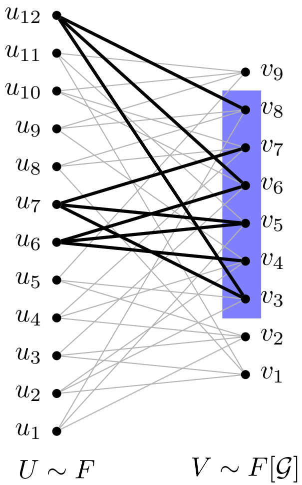
$V' = \{v_3, \ldots, v_8\}$, $\mathrm{Ker}(V') = \{u_6, u_7, u_{12}\}$

Locally looks almost like XORification without recycling, so previous proof might work...
And give bound in terms of $|U| \gg |V|$

Sketch of Proof Sketch

Look at clauses $\mathcal{D}$ in memory in width-w refutation of $F[\mathcal{G}]$

Recover clauses $\mathcal{C}$ in memory in "simulated refutation" of F



Must have $N(\mathit{Vars}(\mathcal{C})) \subseteq \mathit{Vars}(\mathcal{D})$

$\mathrm{Ker}(V') := \{u \in U \,:\, N(u) \subseteq V'\}$

$|V'| \leq r \quad \Longrightarrow \quad |\mathrm{Ker}(V')| \leq |V'|$
(since left vertex sets expand a lot)

Example

$V' = \{v_3, \ldots, v_8\}$, $\mathrm{Ker}(V') = \{u_6, u_7, u_{12}\}$

Locally looks almost like XORification without recycling, so previous proof might work. . .
And give bound in terms of $|U| \gg |V|$

Actual details very different

On the Method of Hardness Condensation

Introduced in [Razborov JACM '16] to show that treelike resolution refutations of width w can require doubly exponential size $2^{n^{\Omega(w)}}$

On the Method of Hardness Condensation

Introduced in [Razborov JACM '16] to show that treelike resolution refutations of width w can require doubly exponential size $2^{n^{\Omega(w)}}$

Has also been used to establish

- Tradeoffs between width and rank for Lovász-Schrijver linear programming hierarchy [Razborov ECCC '16]

On the Method of Hardness Condensation

Introduced in [Razborov JACM '16] to show that treelike resolution refutations of width w can require doubly exponential size $2^{n^{\Omega(w)}}$

Has also been used to establish

- Tradeoffs between width and rank for Lovász-Schrijver linear programming hierarchy [Razborov ECCC '16]
- Quantifier depth lower bounds for finite variable fragments of first-order logic [Berkholz & Nordström LICS '16]

On the Method of Hardness Condensation

Introduced in [Razborov JACM '16] to show that treelike resolution refutations of width w can require doubly exponential size $2^{n^{\Omega(w)}}$

Has also been used to establish

- Tradeoffs between width and rank for Lovász-Schrijver linear programming hierarchy [Razborov ECCC '16]
- Quantifier depth lower bounds for finite variable fragments of first-order logic [Berkholz & Nordström LICS '16]

Where else can this technique be useful?

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

- Weaknesses: non-constant width and **huge** size blow-up

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

- Weaknesses: non-constant width and **huge** size blow-up
- Inherent for XORification with large arity

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case "critical" bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

- Weaknesses: non-constant width and **huge** size blow-up
- Inherent for XORification with large arity

Open question 2

Are there supercritical tradeoffs for 3-CNFs?

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

- Weaknesses: non-constant width and **huge** size blow-up
- Inherent for XORification with large arity

Open question 2

Are there supercritical tradeoffs for 3-CNFs?

- Probably yes, unless PSPACE = EXPTIME

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

- Weaknesses: non-constant width and **huge** size blow-up
- Inherent for XORification with large arity

Open question 2

Are there supercritical tradeoffs for 3-CNFs?

- Probably yes, unless PSPACE = EXPTIME
- Can search for small-space refutations in PSPACE, but finding refutations in given width EXPTIME-complete [Berkholz '12]

Concluding Remarks

- We exhibit supercritical space-width trade-offs for resolution
- Minimizing width can make space go way above linear (worst-case “critical” bound)

Open question 1

Similar tradeoffs for degree vs. space in polynomial calculus?

- Weaknesses: non-constant width and **huge** size blow-up
- Inherent for XORification with large arity

Open question 2

Are there supercritical tradeoffs for 3-CNFs?

- Probably yes, unless PSPACE = EXPTIME
- Can search for small-space refutations in PSPACE, but finding refutations in given width EXPTIME-complete [Berkholz '12]

Thank you for your attention!