Partial Regularization of First-Order Resolution Proofs

J. Gorzny[1] E. Postan[2] B. Woltzenlogel Paleo[3,4]

[1]University of Waterloo

[2]Universidad Nacional de Rosario

[3]Australian National University

[4]Vienna University of Technology

26 Oct 2020

The Quest for Simple Proofs

"The 24th problem in my Paris lecture was to be: Criteria of simplicity, or proof of the greatest simplicity of certain proofs. Develop a theory of the method of proof in mathematics in general. Under a given set of conditions there can be but one simplest proof. Quite generally, if there are two proofs for a theorem, you must keep going until you have derived each from the other, or until it becomes quite evident what variant conditions (and aids) have been used in the two proofs. "

–David Hilbert [Thi03]

First-Order Proof Compression Motivation

- The best, most efficient provers, do not generate the best, least redundant proofs.
- Many compression algorithms for propositional proofs; few for first-order proofs.
- Finding a minimal proof is NP-hard, so use heuristics to find smaller proofs (see [FMP11])

The ‘Real World’

nature.com : Sitemap

nature International weekly journal of science

Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video | For

Archive > Volume 534 > Issue 7605 > News > Article

NATURE | NEWS

Two-hundred-terabyte maths proof is largest ever

A computer cracks the Boolean Pythagorean triples problem — but is it really maths?

Evelyn Lamb

26 May 2016

(See [HKM16])

Proofs as Interfaces

- Larger proofs harder/longer to check; use more resources
- Proofs that are too large may mean solutions can't be written (SAT 2014)
- May use a strict subset of original hypothesis: better proofs!

Our Goal

Lifting propositional proof compression algorithms to first-order logic.

Previous work: `LowerUnits` [FMP11].

This work: `RecyclePivotWithIntersection` [FMP11, BIFH+08]

Recycling Pivots

Removes *irregularities*: inferences η where the pivot occurs as a pivot of another inference below η on the path to the root

- Store a set of *safe* $\mathcal{S}(\eta)$ literals for each node η
- If there are multiple paths, take *intersection* of safe literals
- Bottom-up: compute safe literals; mark deletions
- Top-down: regularize

Recycling Pivots

Removes *irregularities*: inferences η where the pivot occurs as a pivot of another inference below η on the path to the root

- Store a set of *safe* $\mathcal{S}(\eta)$ literals for each node η
- If there are multiple paths, take *intersection* of safe literals
- Bottom-up: compute safe literals; mark deletions
- Top-down: regularize

Recycling Pivots

Removes *irregularities*: inferences η where the pivot occurs as a pivot of another inference below η on the path to the root

- Store a set of *safe* $\mathcal{S}(\eta)$ literals for each node η
- If there are multiple paths, take *intersection* of safe literals
- Bottom-up: compute safe literals; mark deletions
- Top-down: regularize

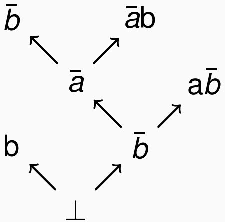
Regularization Can Be Bad

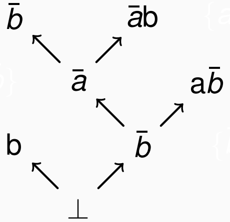
Resolution without irregularities is still complete. But:

Theorem ([Tse70])

There are unsatisfiable formulas whose shortest regular resolution refutations are exponentially longer than their shortest unrestricted resolution refutations.

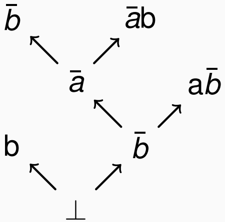
Propositional Example

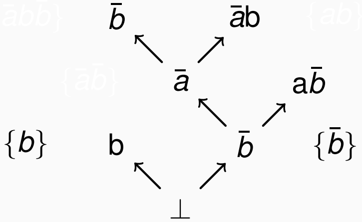




$\uparrow$ safe literal collection

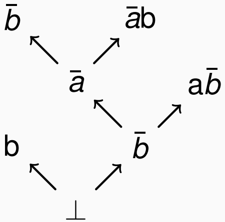
Propositional Example

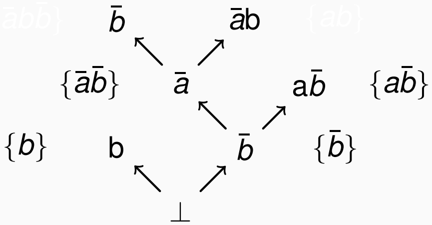




$\uparrow$ safe literal collection

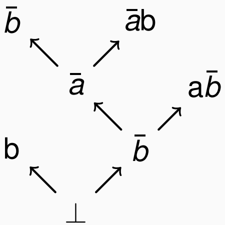
Propositional Example

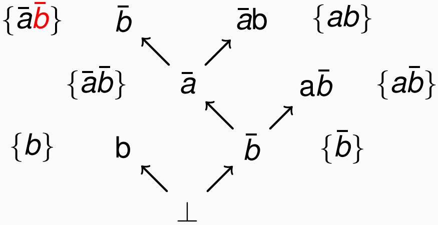




$\uparrow$ safe literal collection

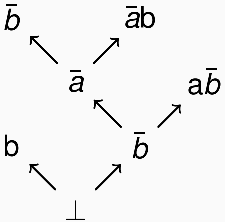
Propositional Example

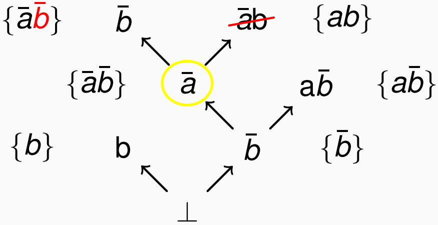




↑ safe literal collection

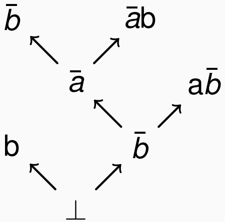
Propositional Example

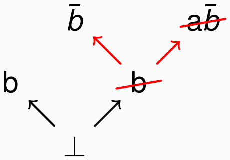




↑ safe literal collection

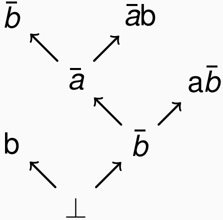
Propositional Example

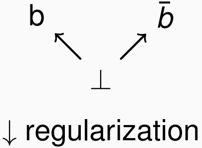




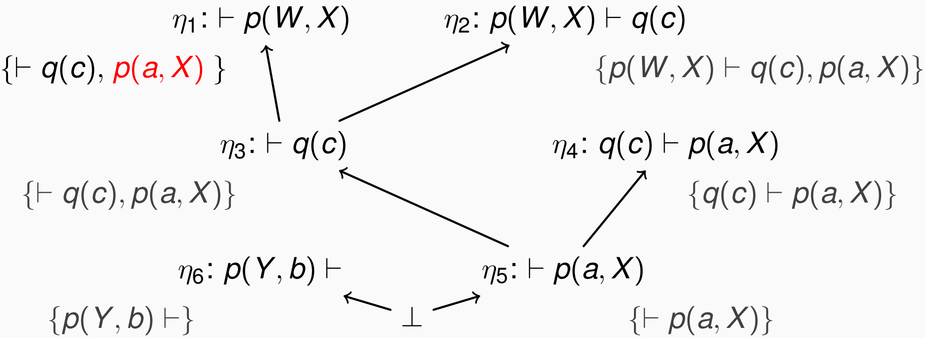
$\downarrow$ regularization

Propositional Example





Pre-Regularization Checks I



$$\sigma = \{W \to a\} \implies \sigma\eta_1 \in \mathcal{S}(\eta_1)$$

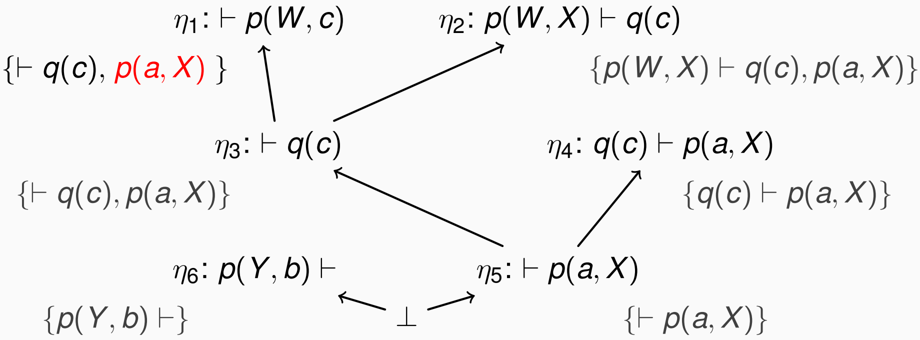
Pre-Regularization Checks I

$\eta_6 \colon p(Y, b) \vdash$ $\qquad\qquad$ $\eta_1 \colon \vdash p(W, X)$

$$\eta_6 \leftarrow \bot \rightarrow \eta_1$$

$$\sigma = \{W \rightarrow Y, X \rightarrow b\}$$

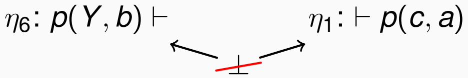
Pre-Regularization Checks II



$$\sigma = \{W \to a, X \to c\} \implies \sigma\eta_1 \in \mathcal{S}(\eta_1)$$

but...

Pre-Regularization Checks II



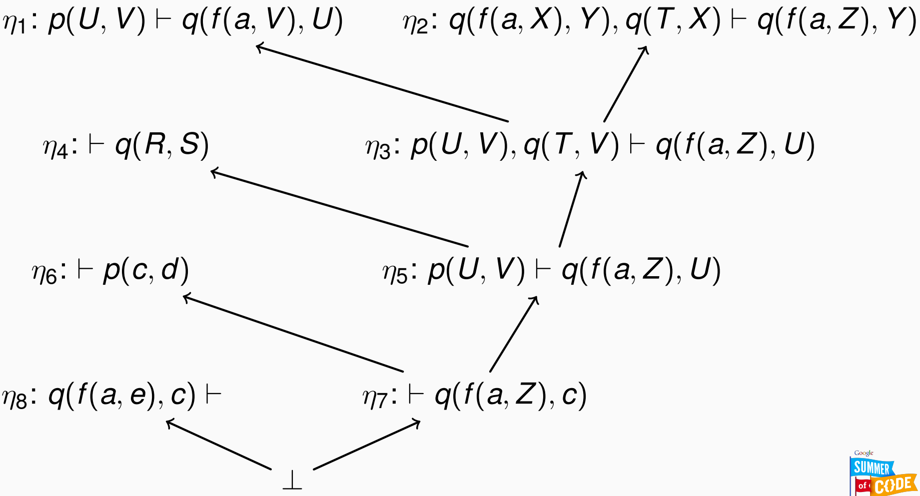
no σ!

Pre-Regularization Unifiability

Definition

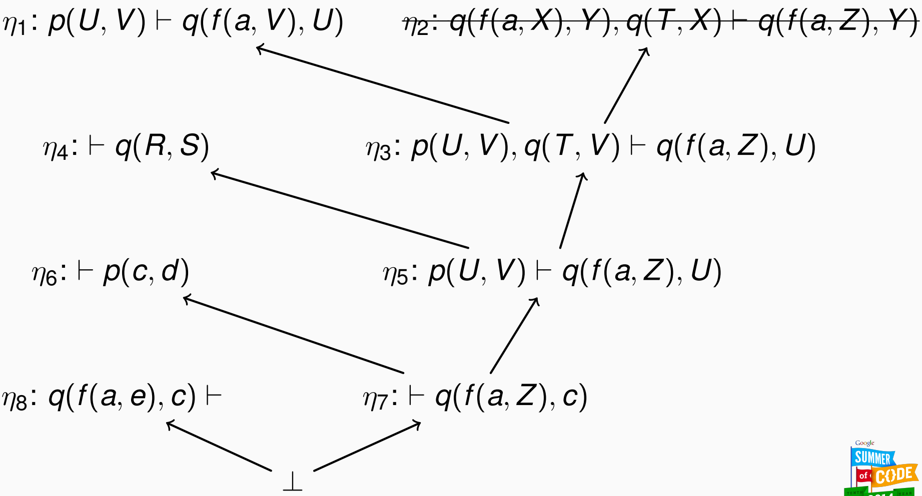
Let η be a node with pivot ℓ' unifiable with safe literal ℓ which is resolved against literals $\ell_1, \ldots, \ell_n$ in a proof ψ. η is said to satisfy the *pre-regularization unifiability property* in ψ if $\ell_1,\ldots,\ell_n$, and $\overline{\ell'}$ are unifiable.

Post-Regularization Checks

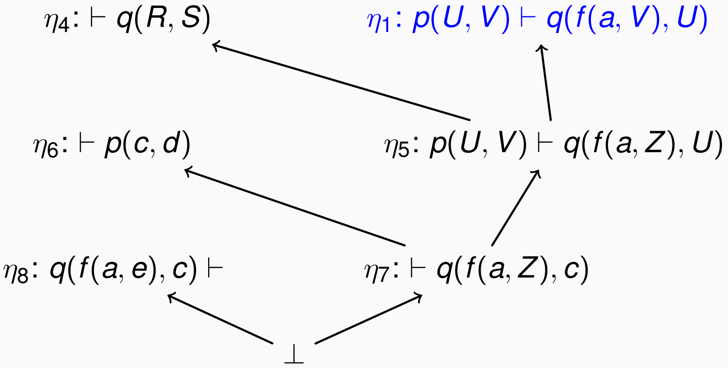


$$\mathcal{S}(\eta_3) = \{q(T, V), p(c, d) \vdash q(f(a, e), c)\}$$

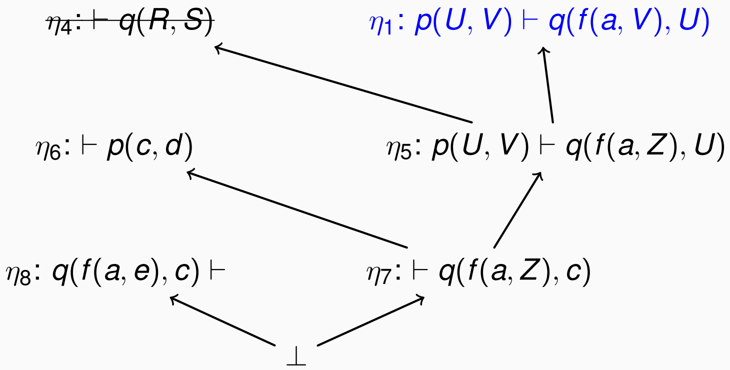
Post-Regularization Checks



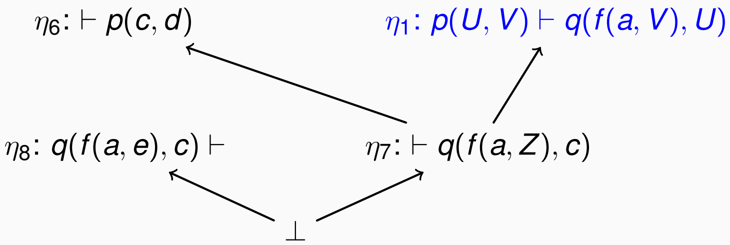
Post-Regularization Checks



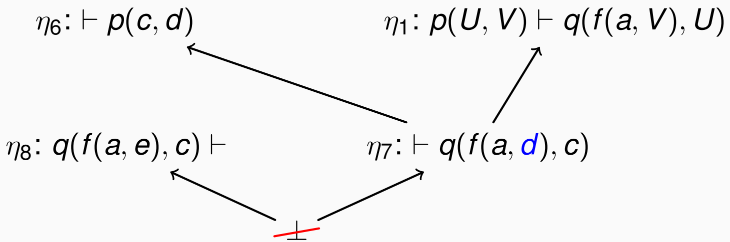
Post-Regularization Checks



Post-Regularization Checks



Post-Regularization Checks



Regularization Unifiability

Definition

Let η be a node with safe literals $\mathcal{S}(\eta) = \phi$ that is marked for regularization with parents η_1 and η_2, where η_2 is marked as a `deletedNode` in a proof ψ. η is said to satisfy the *regularization unifiability property* in ψ if there exists a substitution σ such that $\eta_1\sigma \subseteq \phi$.

First-Order RPI

- Traverse bottom up, collect safe literals (apply unifiers to pivots), check pre-regularization property
- Traverse top-down, check regularization property

Experiment Setup

- Greedy First-Order Lower Units, Recycle Pivots With Intersection implemented as part of Skeptik (in Scala)

- $>$ 2400 randomly generated resolution proofs
- minutes to generate, seconds to compress

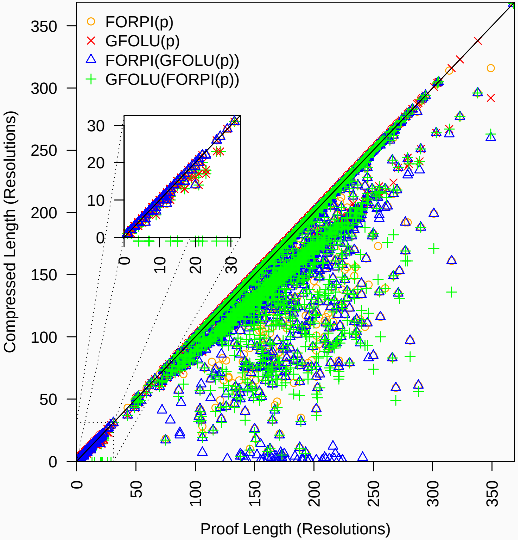
Results

Algorithm	# of Proofs Compressed			# of Removed Nodes		
	TPTP	Random	Both	TPTP	Random	Both
GFOLU(p)	55 (17.9%)	817 (35.9%)	872 (33.7%)	107 (4.8%)	17,769 (4.5%)	17,876 (4.
FORPI(p)	23 (7.5%)	666 (29.2%)	689 (26.2%)	36 (1.6%)	28,904 (7.3%)	28,940 (7.
GFOLU(FORPI(p))	55 (17.9%)	1303 (57.1%)	1358 (52.5%)	120 (5.4%)	48,126 (12.2%)	48,246 (12.
FORPI(GFOLU(p))	23 (7.5%)	1302 (57.1%)	1325 (51.2%)	120 (5.4%)	48,434 (12.3%)	48,554 (12.
Best	59 (19.2%)	1303 (57.1%)	1362 (52.5%)	120 (5.4%)	55,530 (14.1%)	55,650 (14.

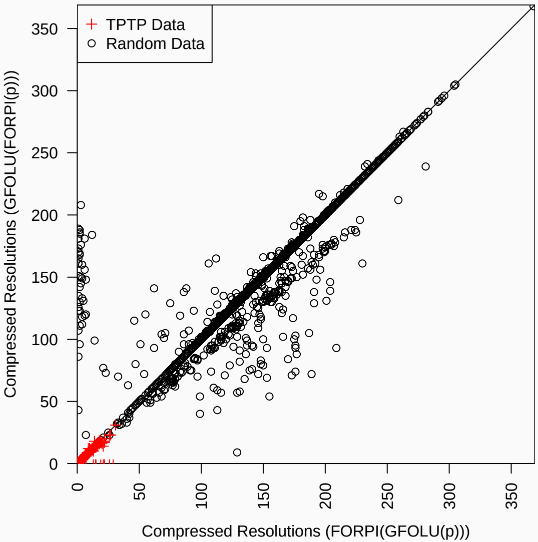
Results

Algorithm	First-Order Compression		Algorithm	Propositional Compression [3]
	All	Compressed Only		
GFOLU(p)	4.5%	13.5%	LU(p)	7.5%
FORPI(p)	6.2%	23.2%	RPI(p)	17.8%
GFOLU(FORPI(p))	10.6%	23.0%	(LU(RPI(p))	21.7%
FORPI(GFOLU(p))	11.1%	21.5%	(RPI(LU(p))	22.0%
Best	12.6%	24.4%	Best	22.0%

Results

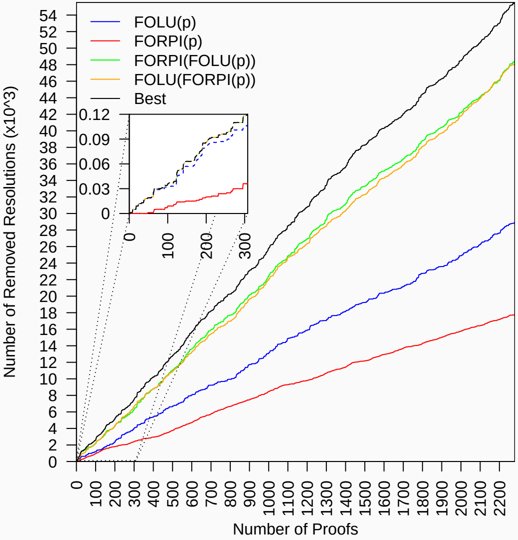

FORPI(p)
GFOLU(p)
FORPI(GFOLU(p))
GFOLU(FORPI(p))
Compressed Length (Resolutions)
Proof Length (Resolutions)
0
50
100
150
200
250
300
350
0
10
20
30

Results



TPTP Data
Random Data
Compressed Resolutions (GFOLU(FORPI(p)))
Compressed Resolutions (FORPI(GFOLU(p)))
0
50
100
150
200
250
300
350

Results


FOLU(p)
FORPI(p)
FORPI(FOLU(p))
FOLU(FORPI(p))
Best
Number of Removed Resolutions (x10^3)
Number of Proofs
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54
0 100 200 300 400 500 600 700 800 900 1000 1100 1200 1300 1400 1500 1600 1700 1800 1900 2000 2100 2200
0 0.03 0.06 0.09 0.12
0 100 200 300

Conclusion

- Another simple, quick algorithm lifted from propositional to first-order logic for proof compression. Use both!
  - LowerUnits compresses more often
  - RPI compresses more
- Future work:
  - Explore other proof compression algorithms?
  - Explore ways of dealing with the post-deletion property quickly

Thank you for your attention.

Any questions?

- Source code: `https://github.com/jgorzny/Skeptik`
- Data: `https://cs.uwaterloo.ca/~jgorzny/data/`
- Expanded paper on Arxiv!

References i

- Omer Bar-Ilan, Oded Fuhrmann, Shlomo Hoory, Ohad Shacham, and Ofer Strichman, *Linear-time reductions of resolution proofs*, Haifa Verification Conference, Springer, 2008, pp. 114–128.
- Pascal Fontaine, Stephan Merz, and Bruno Woltzenlogel Paleo, *Compression of propositional resolution proofs via partial regularization*, International Conference on Automated Deduction, Springer, 2011, pp. 237–251.
- Marijn J. H. Heule, Oliver Kullmann, and Victor W. Marek, *Solving and verifying the boolean pythagorean triples problem via cube-and-conquer*, CoRR **abs/1605.00723** (2016).
- Rüdger Thiele, *Hilbert's twenty-fourth problem*, The American mathematical monthly **110** (2003), no. 1, 1–24.

Gregory Tseitin, *On the complexity of proofs in propositional logics*, Seminars in Mathematics, vol. 8, 1970, pp. 466–483.

To-do

$$
\cfrac{\cfrac{\cfrac{\cfrac{\eta_8\colon p(X), q(X), r(X) \vdash \qquad \eta_7\colon \vdash p(Y)}{\eta_6\colon q(Y), r(Y) \vdash} \qquad \eta_5\colon p(Z) \vdash q(Z)}{\eta_4\colon p(Z), r(Z) \vdash} \qquad \eta_3\colon \vdash r(W)}{\eta_2\colon p(W) \vdash} \qquad \eta_1\colon \vdash p(U)}{\psi\colon \bot}
$$