NULLSTELLENSATZ-PROOFS FOR MULTIPLIER VERIFICATION

Daniela Kaufmann and Armin Biere
Johannes Kepler University
Linz, Austria

CASC 2020

September 14, 2020
Online

JOHANNES KEPLER
UNIVERSITY LINZ

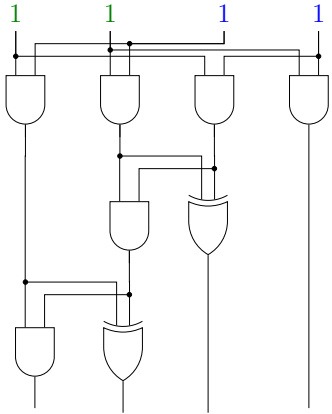
Multiplier Circuits

$$
\begin{array}{cccc}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & {\scriptstyle 1} & & {\scriptstyle 1} & {\scriptstyle 0} \\
\hline
 & 1 & 0 & 0 & 1
\end{array}
$$

$3 \cdot 3 = 9$

Multiplier Circuits

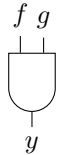
$$
\begin{array}{cccc}
1 & 1 & \cdot & 1 & 1 \\
\hline
 & & & 1 & 1 \\
 & & 1 & 1 & \\
 & 1 & 1 & 0 & \\
\hline
 & 1 & 0 & 0 & 1
\end{array}
$$



$3 \cdot 3 = 9$

Multiplier Circuits

AND-Gate



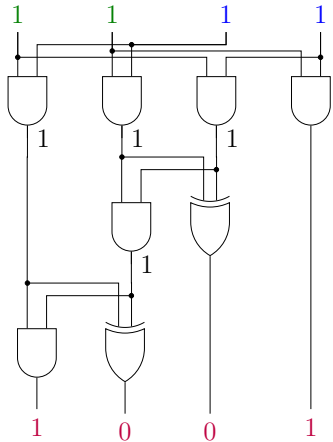
$f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate

$f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0

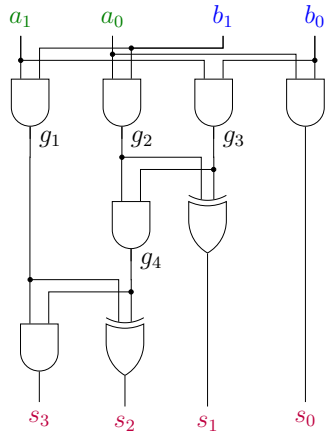


Multiplier Circuits

Given: Gate-level multiplier for fixed bit-width.

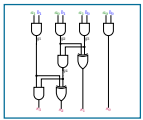
Question: For all possible $a_i, b_i \in \mathbb{B}$:

$$(2a_1 + a_0) * (2b_1 + b_0) = 8s_3 + 4s_2 + 2s_1 + s_0?$$



Motivation

Multiplier



Polynomials

$$
\begin{aligned}
B = \{ & \\
& x - a_0 * b_0, \\
& y - a_1 * b_1, \\
& s_0 - x * y, \\
& \dots \\
\} &
\end{aligned}
$$

Specification

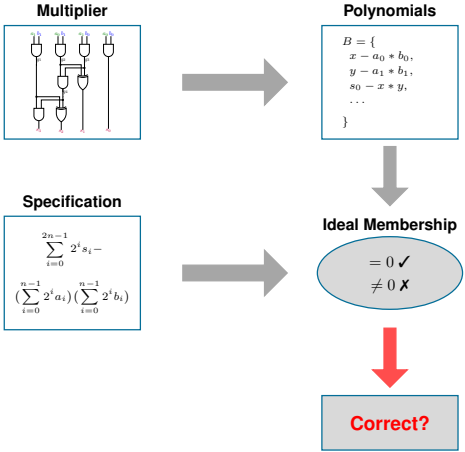
$$\sum_{i=0}^{2n-1} 2^i s_i - (\sum_{i=0}^{n-1} 2^i a_i)(\sum_{i=0}^{n-1} 2^i b_i)$$

Ideal Membership

$= 0$ ✓

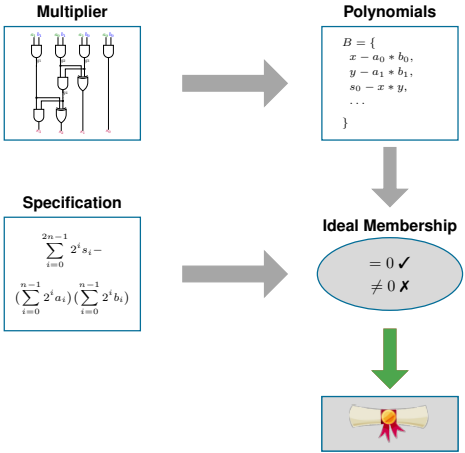
$\neq 0$ ✗

Motivation



Problem: Verification might not be error free

Motivation

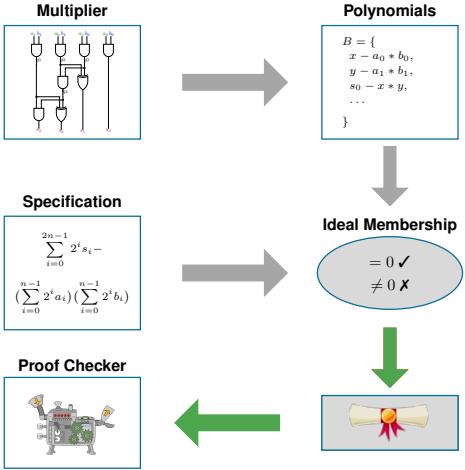


Problem: Verification might not be error free

Goal: Validate result of verification process

- Generate machine-checkable proofs
- Check by independent proof checkers

Motivation



Problem: Verification might not be error free

Algebraic Proof Systems:

- Polynomial Calculus [Clegg et al., STOC'96]
  - D. Ritirc, A. Biere, and M. Kauers. A Practical Algebraic Calculus for Arithmetic Circuit Verification. In SC2'2018.
  - D. Kaufmann, M.Fleury, and A. Biere. The Proof Checkers Pacheck and Pastèque for the Practical Algebraic Calculus. In FMCAD'20.
- Nullstellensatz Proofs [Beame et al., Proc. LMS'96]
  - Topic of this paper.

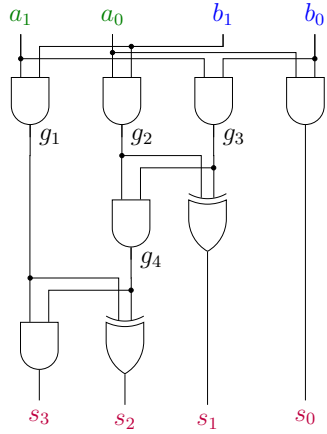
From Circuits to Polynomials

AND-Gate $f \wedge g = y$

f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

XOR-Gate $f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0



From Circuits to Polynomials

AND-Gate $f \wedge g = y$

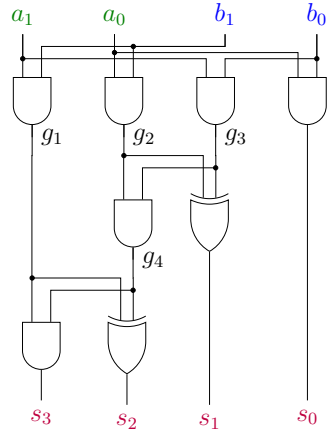
f	g	y
0	0	0
0	1	0
1	0	0
1	1	1

$$-y + fg = 0$$

XOR-Gate $f \oplus g = y$

f	g	y
0	0	0
0	1	1
1	0	1
1	1	0

$$-y + f + g - 2fg = 0$$



From Circuits to Polynomials

Gate polynomials $G(C) \subseteq \mathbb{Z}[X]$.

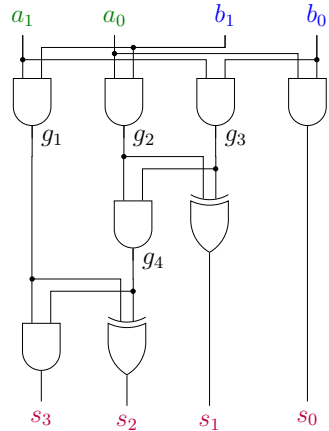
$$
\begin{array}{ll}
s_3 = g_1 \wedge g_4 & -s_3 + g_4 g_1, \\
s_2 = g_1 \oplus g_4 & -s_2 - 2g_4 g_1 + g_4 + g_1, \\
g_4 = g_2 \wedge g_3 & -g_4 + g_2 g_3, \\
s_1 = g_2 \oplus g_3 & -s_1 - 2g_2 g_3 + g_2 + g_3, \\
g_1 = a_1 \wedge b_1 & -g_1 + a_1 b_1, \\
g_2 = a_0 \wedge b_1 & -g_2 + a_0 b_1, \\
g_3 = a_1 \wedge b_0 & -g_3 + a_1 b_0, \\
s_0 = a_0 \wedge b_0 & -s_0 + a_0 b_0
\end{array}
$$

Boolean value constraints $B(C) \subseteq \mathbb{Z}[X]$.

$$
\begin{array}{ll}
a_1, a_0 \in \mathbb{B} & a_1(1-a_1),\ a_0(1-a_0), \\
b_1, b_0 \in \mathbb{B} & b_1(1-b_1),\ b_0(1-b_0)
\end{array}
$$

Specification $\mathcal{S}_n \in \mathbb{Z}[X]$.

$$8s_3 + 4s_2 + 2s_1 + s_0 - (2a_1 + a_0)(2b_1 + b_0)$$



Verification

[Kaufmann et al., FMCAD'19]

- Let $J(C) = \langle G(C) \cup B(C) \rangle \subseteq \mathbb{Z}[X]$.
- Circuit fulfills specification $\Leftrightarrow \mathcal{S}_n \in J(C)$.

Verification

[Kaufmann et al., FMCAD'19]

- Let $J(C) = \langle G(C) \cup B(C) \rangle \subseteq \mathbb{Z}[X]$.
- Circuit fulfills specification $\Leftrightarrow \mathcal{S}_n \in J(C)$.

- Reverse topological term ordering $\Rightarrow G(C) \cup B(C)$ is a D-Gröbner basis for $J(C)$.
- To show correctness we reduce $\mathcal{S}_n$ by $G(C) \cup B(C)$.

Verification

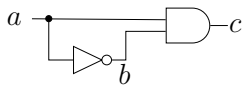
[Kaufmann et al., FMCAD'19]

- Let $J(C) = \langle G(C) \cup B(C) \rangle \subseteq \mathbb{Z}[X]$.
- Circuit fulfills specification $\Leftrightarrow \mathcal{S}_n \in J(C)$.

- Reverse topological term ordering $\Rightarrow G(C) \cup B(C)$ is a D-Gröbner basis for $J(C)$.
- To show correctness we reduce $\mathcal{S}_n$ by $G(C) \cup B(C)$.

- Simply reducing $\mathcal{S}_n$ leads to large intermediate reduction results.
  $\Rightarrow$ Preprocessing techniques based on variable elimination.
- Transform $G(C)$ to a rewritten D-Gröbner basis $G'(C)$.
- To show correctness we reduce $\mathcal{S}_n$ by $G'(C) \cup B(C)$.

Algebraic Proof Systems




```
G(C) =   -b+1-a,      b = ¬a
         -c+a*b,      c = a ∧ b = a ∧ ¬a
Spec =   c            c = ⊥
```

Polynomial Calculus [Clegg et al., STOC'96]

```
* : -b+1-a,   a,       -a*b;
+ : -a*b,     -c+a*b,  -c;
* : -c,       -1,      c;
```

- We modified format to support proof checking (PAC). [SC2'18, FMCAD'20]
- Captures reduction process on-the-fly.
- Allows to detect position of bugs.

Nullstellensatz Proofs [Beame et al., Proc. LMS'96]

```
a;
-1;
```

- Provides list of co-factors.
- Correctness is checked by expanding linear combination.
- Smaller and shorter than PAC.

Conjecture

In [Kaufmann, PhD Thesis]:

> *"In a correct NSS proof we would also need to express the rewritten polynomials $G'(C)$ as a linear combination of the given set of polynomials $G(C)$ and thus lose the optimized representation, which will most likely lead to an exponential blow-up of the co-factors in the NSS proof."*

Proof Generation

Update basis representation during preprocessing and reduction.

Basis representation.

Let $\mathrm{base}(r) = \{(p_i, q_i) \mid p_i \in G(C),\, q_i \in \mathbb{Z}[X]/\langle B(C)\rangle\}$ be the basis representation of $r \in \mathbb{Z}[X]/\langle B(C)\rangle$, such that $r = \sum_{(p_i,q_i)\,\in \mathrm{base}(r)} q_i p_i$.

Algorithm 1: Spec-Reduction($\mathcal{S}_n, G'(C)$)

Input : Circuit specification $\mathcal{S}_n \in \mathbb{Z}[X]$, D-Gröbner basis $G'(C)$
Output: Remainder r, Basis representation $\mathrm{base}(\mathcal{S}_n)$

$r \leftarrow \mathcal{S}_n, \mathrm{base}(\mathcal{S}_n) \leftarrow \{\}$;
foreach $g \in G'(C)$ **do**
    $r, h \leftarrow$ Reduction(r, g);
    $\mathrm{base}(\mathcal{S}_n) \leftarrow$ Add-to-basis-representation($g, h, \mathrm{base}(\mathcal{S}_n)$);
return $r, \mathrm{base}(\mathcal{S}_n)$

NUSS-CHECKER

- Implemented in C
- 1800 lines of code
- Available: `fmv.jku.at/nussproofs`

- NUSS-CHECKER reads three input files `<input>`, `<cofact>`, and `<target>`.
- Verifies that the polynomial in `<target>` can be represented as a linear combination of the polynomials in `<input>` using the co-factors provided in `<cofact>`.

- We always calculate modulo $\langle B(C) \rangle$.

Evaluation

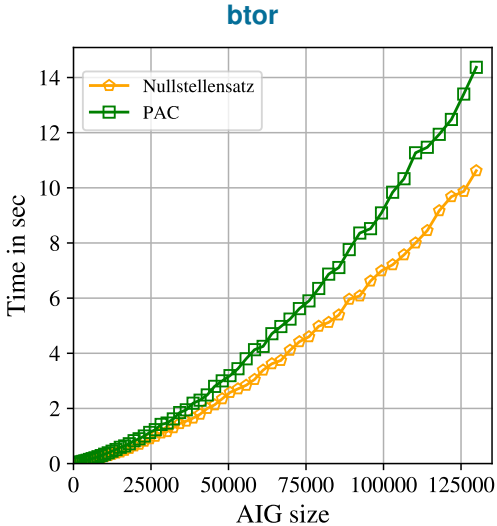
btor – Multipliers

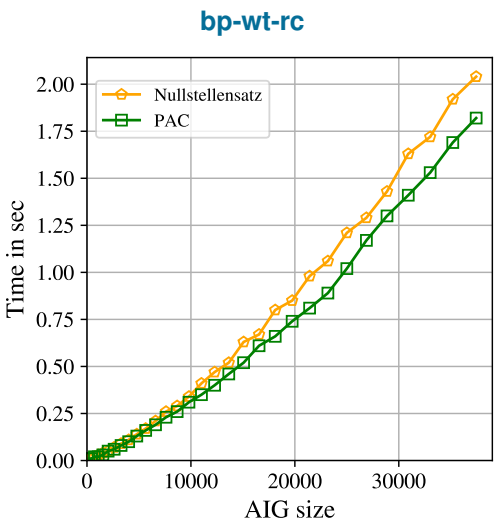
- Simple multiplier architecture
  - Partial product gen.: AND - gates
  - Partial product acc.: Array structure
  - Final stage adder: Ripple Carry
- Generated using Boolector (btor)
- Input bit-width $n \in [4, 128]$

bp-wt-rc – Multipliers

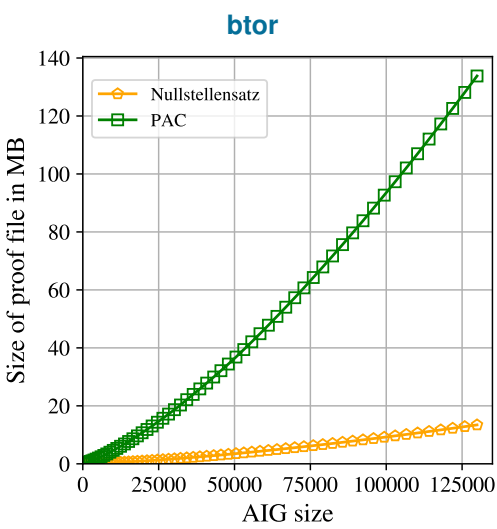
- Complex multiplier architecture
  - Partial product gen.: Booth (bp)
  - Partial product acc.: Wallace tree (wt)
  - Final stage adder: Ripple Carry (rc)
- Part of the aoki benchmarks
- Input bit-width $n \in [4, 64]$

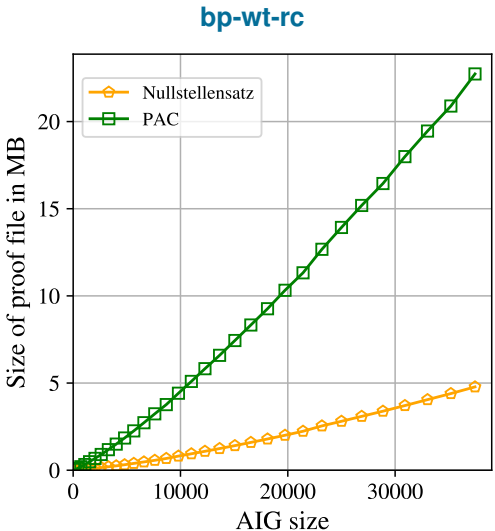
Proof Generation Time



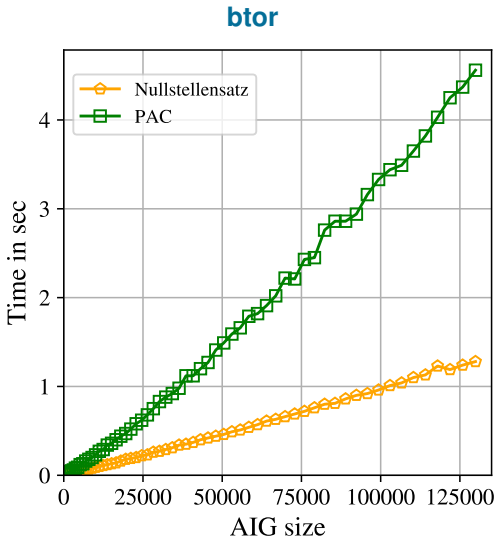


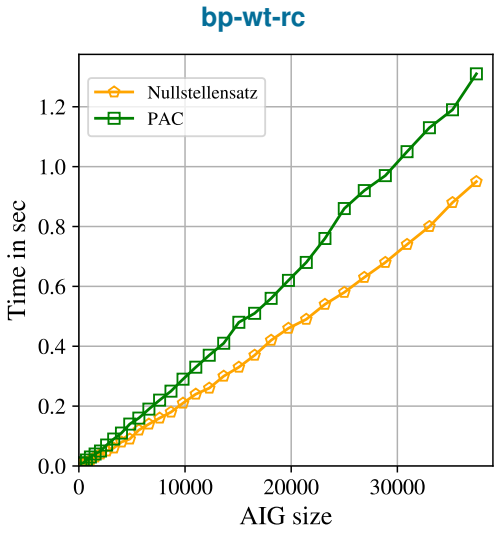
Size of the proof files





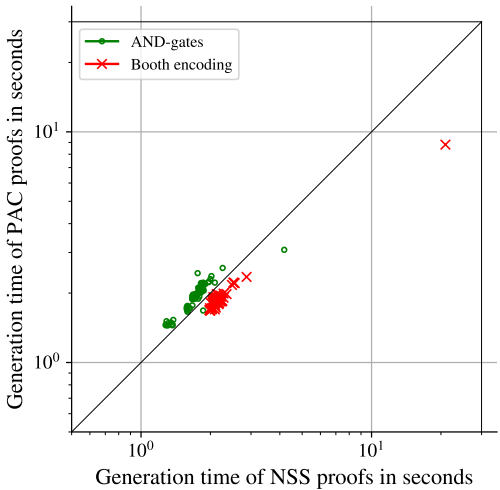
Proof Checking Time



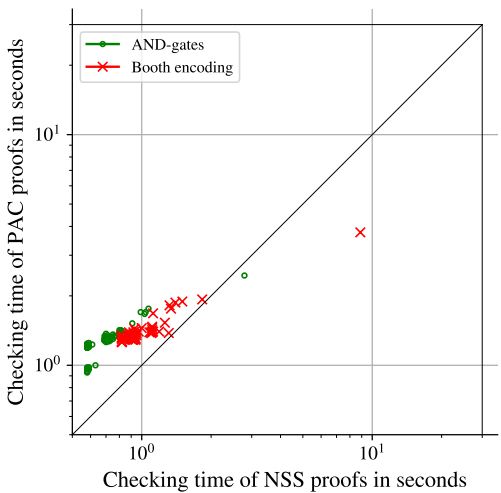


168 different multiplier architectures

Proof Generation Time



Proof Checking Time



Conclusion

Further contents

- Details for proof generation.
- Implementation details of NUSS-CHECKER.
- Formal proof of proof size in $\mathcal{O}(n^2)$ for simple multipliers.
- Empirical proof of proof size in $\mathcal{O}(n^2)$ for complex multipliers.

Conclusion

- NSS proofs are shorter and faster to check, but not always faster to generate.
- However, NSS proofs do not allow localization of errors.

NULLSTELLENSATZ-PROOFS FOR MULTIPLIER VERIFICATION

Daniela Kaufmann and Armin Biere
Johannes Kepler University
Linz, Austria

CASC 2020

September 14, 2020
Online

JOHANNES KEPLER
UNIVERSITY LINZ